

**The Application of System Dynamics to Project Management
An Integrated Methodology (SYDPIM)**

By

Alexandre G. Rodrigues

*Thesis submitted to the Department of Management Science in partial fulfilment of
the requirements for the degree of:*

Doctor of Philosophy

at the

UNIVERSITY OF STRATHCLYDE



July, 2000

Glasgow, United Kingdom

The copyright of this thesis belongs to the author under the terms of the United Kingdom Copyright Acts as qualified by University of Strathclyde Regulation 3.49. Due acknowledgement must always be made of the use of any material contained in, or derived from, this thesis.

July, 2000.

7. The SYDPIM Project Management Method

7.1 Introduction

The SYDPIM methodology has two main components: (1) a formal method to develop and validate a SD project model, and (2) a formal method to use a SD project model within the conventional project management framework, in a formally integrated manner with a PERT/CPM model. The first component was just described in the previous chapter.

This chapter describes the project management method of the SYDPIM methodology. This method is based on a process framework, which is an enhancement of the traditional project control cycle: a SD model is embedded within this process, enhancing the planning and monitoring functions. At the core of the SYDPIM Project Management Method are the analytical links established between the SD model and the PERT/CPM model. These links are essential to the formal integrative nature of the SYDPIM methodology.

7.2 Structure of this chapter

This chapter describes the SYDPIM Project Management Method as a formal model. This model is composed of four main elements: process logic, objects, analytical links and activities. The chapter can be considered as divided into four main parts, each part describing in detail one of these elements. In the end of this thesis, appendix A presents a definition of the key terms and expressions used throughout this chapter, appendix B presents various tables, appendix C presents the formal specification of all SYDPIM objects and appendix D presents the specification of all SYDPIM analytical links.

The next section 7.3 provides a brief overview of the formal model of the SYDPIM Project Management Method. The following section 7.4 describes each of its four elements in great detail. In the last section 7.5, some conclusions about the conceptual and practical aspects of the methodology are discussed.

The first element to be described in section 7.4 is the SYDPIM process logic (section 7.4.1). First, the general principles and assumptions underlying the SYDPIM process framework are presented. A simplified scenario based on the use of a single SD project model is then proposed as the basis to describe the SYDPIM Project Management Method throughout the chapter. The underlying process logic of the traditional PERT/CPM based framework is then described. The process logic of the SYDPIM framework is developed and is divided into two main sub-components: monitoring and planning, which are described separately. Overall, the SYDPIM process logic interconnects and articulates the use of objects, analytical links and activities.

The specification of the activities requires the use of the analytical links and of the objects. On the other hand, the formal specification of the analytical links requires the SD and PERT/CPM models to be specified as objects. Therefore, the next element of the SYDPIM formal model to be described are the SYDPIM objects.

Objects are defined in section 7.4.2 as data repositories and data processors. Since the SYDPIM objects need to be specified formally, a specification language is proposed. This language is simple, accessible to the non-expert, and provides the required level of formality. The objects are specified as comprising a hierarchical data-structure which stores data, operators (which provide information about the object's state), and conditions of validity (which must be verified for the object to be usable within SYDPIM). Throughout this detailed description of the SYDPIM objects, a rationale for the proposed concepts is presented whenever appropriate. While these rationale discussions can make the description of the objects longer, the aim is to provide a good conceptual foundation to justify the choices made, as well as to support the implementation of other alternatives.

In the following section 7.4.3 the analytical links of SYDPIM basic mode are described in detail, and are formally specified. This specification of the analytical links requires a full consideration and understanding of all aspects of integrating a SD project model within the conventional project management process. Therefore, some basic principles of integration are first discussed. The principles underlying each of the three types of links proposed (data links, structural links and data-structural links), are further discussed in more detail. The generic and elementary

operations within SYDPIM which require the analytical links are identified. This is followed by a discussion of the dynamic nature of the links, which emphasises that the links themselves will change over-time in the course of the project. In order to consolidate all these concepts and principles, some illustrative examples of SYDPIM links are then presented. In the remainder of this section, the analytical links proposed are presented, discussed, and specified formally. Like with the objects, a formal specification language is required. The same language used for the objects is extended to specify the analytical links, which are considered as possibly comprising three main elements: mathematical relationships, operators and validity conditions. Since the data and data-structural links are based upon the structural links, the latter are first specified. An overview of key assumptions is presented, followed by a separate discussion and presentation of each of the structural links (work breakdown, organisation breakdown and work dependencies links) – the specifications of all links can be found in appendix D.1. Where appropriate, formal algorithms that support the implementation of the links are presented. A summary of the structural links is presented and this section moves to the data-links. An overview is presented followed by a discussion of the various types of links that can be established. This is followed by a discussion of a critical issue for the establishment of data links: the dis-aggregation of data and the possible automation of this process. The following sub-sections provide a brief explanation of how the analytical links relate to the SYDPIM objects, as well as to the SYDPIM process logic. Since the overall number of data links proposed for SYDPIM basic mode is large, a summary overview of these links is first presented. The formal specification language is revised and each individual data link is then presented and specified formally. The data links are presented in groups, according their type: data exchange input-input, data exchange output-input, data consistency input-input, data consistency output-input and data consistency output-output. For each type of data link, an overview rationale is presented. While these discussions can make the presentation of the data links longer, the purpose is to provide a basis from which other alternatives can be implemented and extensions can be developed. Where appropriate, particular attention is given to explain why certain alternative routes should *not* be followed. The formal specification of all links can be found in appendix D.2. Finally, the data-structural links are presented. An overview rationale discusses the nature of these links and states some assumptions regarding SYDPIM basic mode. The potential application of these links within the SYDPIM

framework is then discussed. The presentation of the specific data-structural links considered in SYDPIM basic mode is preceded by a rationale discussion, which explains why and how these links can be established – some algorithms are presented. The description of these links ends with a discussion about how other possible links of this type could be useful. A brief summary of all analytical links described in this section 7.4.3 is presented.

In the following section 7.4.4, each of the activities of the SYDPIM process logic is described in detail. These activities make use of the SYDPIM objects and analytical links. The SYDPIM activities are described in a structured manner, making use of formal algorithms where possible, and considering exceptions and flexibility where required. The activities of the planning and monitoring functions of project control are described separately. For the sake of clarity, the sequence chosen to describe the activities is based on the conventional framework for (re)planning a project based on the PERT/CPM model, and only one single iteration is considered. The nine activities of the SYDPIM planning process are described in the sequence proposed. In order to improve the clarity of the overall process, a global picture of the SYDPIM planning process is built-up progressively. Some of the branches within the process logic of SYDPIM planning involve the use of the analytical links. The required operations within these branches are described. The SYDPIM planning process also leads to the concept of an integrated SYDPIM project plan, which merges the information of the two project plans – i.e. the PERT/CPM plan and the SD plan. This issue is clarified, and the remainder of this section describes the SYDPIM monitoring activities. This process is the same as for the planning activities, where a global picture of the monitoring process is progressively built throughout. The section ends with a brief summary of the SYDPIM activities.

The final section 7.5 of this chapter is aimed at providing a summary and an overall discussion about the novelty, impacts and constraints of the SYDPIM Project Management Method, regarding both theory and practice. The SYDPIM methodology approaches a problem never addressed before: embedding the use of a SD project model within the conventional framework of project management, while formally integrating this model with a PERT/CPM model through the implementation of analytical links. The analytical integration of the two models provides a technical novelty. The way in which the SD project model is articulated within the

conventional project management framework, and how this process is re-shaped, provides a “re-engineering” novelty. But what are the practical impacts of these novelties? What are the critical requirements to improve performance of project management in practice, by implementing SYDPIM? What constraints should be considered? Can further theoretical developments help? What are the priority ones? Some of these questions are addressed in the final section of this chapter.

7.3 Overview

The SYDPIM Project Management Method describes how a SD project model is used within the project management control process, integrated with a PERT/CPM model.

The SYDPIM Project Management Method is here formally described as a conceptual *model*, which is composed by the following main elements:

- *process logic* – a logical flow of activities through which project control is implemented;
- *analytical links* – connect the SD model with the PERT/CPM model and are formally defined through equations, formal conditions and algorithms;
- *objects* – entities that represent data repositories and data processing elements, like a metrics database and a simulation model;
- *activities* – specify the major operations within the process logic.

The process logic consists of a formal step-by-step framework, with branches and alternative paths, which account for possible constraints. For example, where the project past behaviour needs to be specified, whether a current PERT/CPM plan is available or not, this will affect the steps to be taken. The process logic works as a formal framework, which interconnects and articulates the other elements of the conceptual model.

The analytical links consist of formal specifications of how the two models (i.e. PERT/CPM and SD model) are integrated. This involves structure and data. Depending on the type of link, equations, formal conditions and algorithms can be used to specify the links. For example, a structural link mapping PERT/CPM tasks to SD-Tasks can be described through a matrix type of data-structure, whereas the

transfer of scheduling data can be specified through attribution type of equations (e.g. $a := c + d$). Checking whether the two models are representing the same project plan can be specified through formal conditions (e.g. $a + d == b + c$). Finally, deriving the work dependencies between two SD-Tasks from the dependencies among various tasks in the PERT/CPM model, can be described through an algorithm. An analytical link can incorporate various of these relationships between the models' elements.

Objects represent what in the real world are data repositories and/or data processing units. For example, the SD model and the PERT/CPM model are the most relevant objects of SYDPIM, which work as both data repositories and as processing units. The project past behaviour, defined as a set of data-patterns over-time, is another important object, which works as a data-repository only.

Activities are major operations which are carried out at specific stages in the process logic. These operations may involved several sub-activities which can be more or less formal, simple or complex. For example, readjusting the PERT/CPM model to incorporate a project plan specified in the SD model is a complex activity. Calibrating the SD model to represent the project plan in the PERT/CPM model is another critical activity. Updating the project past behaviour (an object) from the PERT/CPM model is a formal but simpler activity. Improving the project plan in the SD model can be complex and is an informal activity.

The formal specification of these four elements constitutes the formal specification of the SYDPIM Project Management Method. These specifications are presented in the following sections. A number of terms and expressions will be used with some frequency throughout. Definitions of the main terms and expressions employed are provided in appendix A.

As a formal methodology, SYDPIM proposes a step-by-step and well defined logical process. However, it is important to note that while the ideal of project management is a well defined control process, the real world raises various constraints of social and subjective nature, which must be coped with. Examples are the availability and reliability of data and the regular update of a detailed PERT/CPM project plan. While the SYDPIM Project Management Method pursues the ideal of a well

managed control process, its underlying framework was designed to be flexible and thereby able to cope with this type of constraints.

7.4 The formal model

7.4.1 The process logic

Introduction

Managing a project is about controlling a complex social system towards desired targets. Therefore, the traditional project management framework consists of a control cycle wherein the project status is monitored, compared against the objectives and, when necessary, the work is re-planned. The process logic of the SYDPIM model is an enhancement of this cycle. A SD project model is used within the monitoring and re-planning functions of the project management cycle. The model is used in a formally integrated manner with a PERT/CPM model thereby improving these functions.

The ideal of any control process is automation. This means that the activities of monitoring status and generating corrective actions can be formally specified and thereby implemented through an automated mechanism. In such a scenario, the control process is generally effective and stable (though not necessarily perfect).

Likewise, the ideal of the project management control framework is a formal step-by-step stable process. However, it is well known that the implementation of such a mechanism to regulate a complex social system is very difficult if not impossible. Since a project is a complex social system, in practice various constraints of human nature restrain this desired formality.

The process logic of the SYDPIM model pursues the ideal of a stable control mechanism. As such, it is specified as a formal process as if it could be automated. As any other model of the project management control process, this is unlikely to be achieved. In order to cope with this, the process logic of the SYDPIM model considers explicitly some of the more relevant constraints, and proposes alternative paths as solutions. It is important to stress that this reflects a fundamental principle

of the SYDPIM methodology: the possibility of being implemented at various levels of formality and hence feasible within more or less structured management environments. As it will be seen, the two most important constraints stem from the way in which the PERT/CPM model is used by management and the availability of data.

Overview of general principles and assumptions

As mentioned above the process logic of the SYDPIM framework builds upon the traditional cycle of project control. This cycle consists of two main management functions: planning and monitoring, wherein a PERT/CPM model is used. In planning the model is used to develop a work plan. In monitoring the model is updated with actual results. Based on these results, the model forecasts a new future and current performance is compared against the targets. The work plan eventually is revised in the planning function, and the control cycle repeats. This cycle assumes a period of time for the control process. Depending on the complexity of the project, this may be longer or shorter, constant or varying. A typical control cycle in most projects is the month. This traditional control cycle is described in the figure 7.1 below.

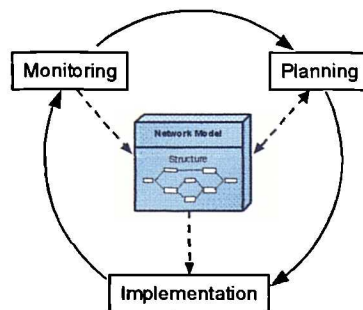


Figure 7.1 – The traditional control cycle of PERT/CPM based project management

An important aspect of this traditional control cycle is the time horizon and the detail of the plan developed in the PERT/CPM model in the planning function. It is common sense that the further the manager looks into the future the less detail should be considered. The reason for this is that the further into the future the higher the uncertainty and hence the more difficult it is to anticipate in great detail. Such an approach could lead to unproductive efforts. How do managers cope with this problem in the traditional framework? There is no standard approach. One

common solution is to develop a work plan within the PERT/CPM model with various levels of detail. The PERT/CPM plan at the most detailed level will have the shortest time-horizon (say one month). The more aggregate PERT/CPM model will cover the whole project life-cycle. In practice, in most cases no more than three levels are considered. Another common solution is to have only one single PERT/CPM plan covering the whole project life-cycle but its tasks consider higher levels of aggregation as one moves into the future. This is an important project management issue considered in SYDPIM.

The critical path analysis provided proves less useful and less accurate as the level of aggregation of the tasks increases. There are two reasons for this: (1) as tasks become more aggregate, their estimated duration and budget encapsulate more implicit assumptions hence unlikely to be accurate, and (2) the principle behind the model is to decompose complexity into simple and easily manageable tasks. Once this is done and the project is decomposed into a logical network with a large number of tasks (hundreds or thousands), the results from the critical path analysis are insightful and could not be derived intuitively. The same would not happen with a simple network with few highly aggregated tasks. The conclusion is that the highest level of aggregation in the PERT/CPM model, covering the full project life-cycle, is only useful to provide “an helicopter” view of the main schedules, project phases, and milestones. “What-if” critical path analysis is of little use at this level. This way, for those tasks where a more detailed plan is available their estimates are generated from below – i.e. the bottom-up approach. Since there will not be a detailed plan available for the later tasks, an overall reliable estimate based on “bottom-up” will not be available from the PERT/CPM model. Typically, the later project phases are scheduled and budgeted to fit the demands of the contract with the Client. A detailed plan will only be developed for these phases later in the project, where the circumstances will differ from the beginning of the project. Interestingly, it is in these later phases that major problems typically emerge. In summary, the PERT/CPM model can only provide insightful visibility into the short-term of the project. Its high-level view of the project only informs management about *desired* rather than *likely* schedules, budgets and milestones.

While enhancing the traditional control cycle, the SYDPIM methodology addresses this issue by considering the possibility of two different SD models being used at

different levels of aggregation. In this scenario, a more detailed model (referred to as “operational model”) is used to analyse the specific short-term issues of the current project phase. The calibration of this model or even its structure does not cover the full project life-cycle. The time-horizon for the simulation of this model is restricted to some number of control cycles, wherein reliable data is available to calibrate the model at that level of detail. Another more aggregated model is used to cover the full project life-cycle from the beginning of the project (referred to as “strategic model”). The strategic model is used to analyse the longer term general issues of the project. Why not to use only one detailed model for the full project life-cycle? Like with the PERT/CPM model, a detailed SD model will most likely require data for calibration which is not available or not enough accurate to produce reliable estimates. Furthermore, management in general prefer to assess longer-term issues under a higher-level clearer perspective.

SYDPIM does not require that both models are used. Only one SD project model is required. In the scenario where the two models are used, “what-if” analysis to assess the impact of decisions or risks is first carried out in the strategic model to inform management about the long-term impacts in the whole project. These are then assessed for shorter-term impacts in the operational model. As in many real life situations, the results may differ, and hence what can be a short-term slippage may prove beneficial at the end of the project. If the two models are used, they should be integrated as formally as possible, in a similar way in which they will be integrated with the PERT/CPM model. This requires that at the end of every control cycle (i.e. after re-planning and prior to implementation) the two models are calibrated to the same scenario, with their input and output being consistent one another.

How does the use of the two models relate to the various possible levels of detail in the PERT/CPM model? Since there is no standard and universal approach to the number of levels used in the PERT/CPM model, SYDPIM considers the three classic levels: operational, tactical and strategic. It is assumed that the relevant usefulness of the critical path analysis of the PERT/CPM can be found at the operational level. At this level, the model can provide reliable forecasts and can be used as the basis for re-planning. A logical network at this level will have 100 tasks or more. At the tactical level, the network will have from 10 to 100 tasks, and at the

strategic level less than 10 – these numbers are indicative only, and do not attempt a formal definition. A SD model generally assumes a level of aggregation much higher than an operational network. The work breakdown is one of the main indices of aggregation in a project model. A SD model typically dis-aggregates a project from one task down to 30 tasks maximum. The SD operational model will therefore lay at the tactical PERT/CPM level, and the SD strategic model will lay at the strategic PERT/CPM level. This scenario for the combined use of the two types of models at different levels of detail is shown in the figure 7.2 below.

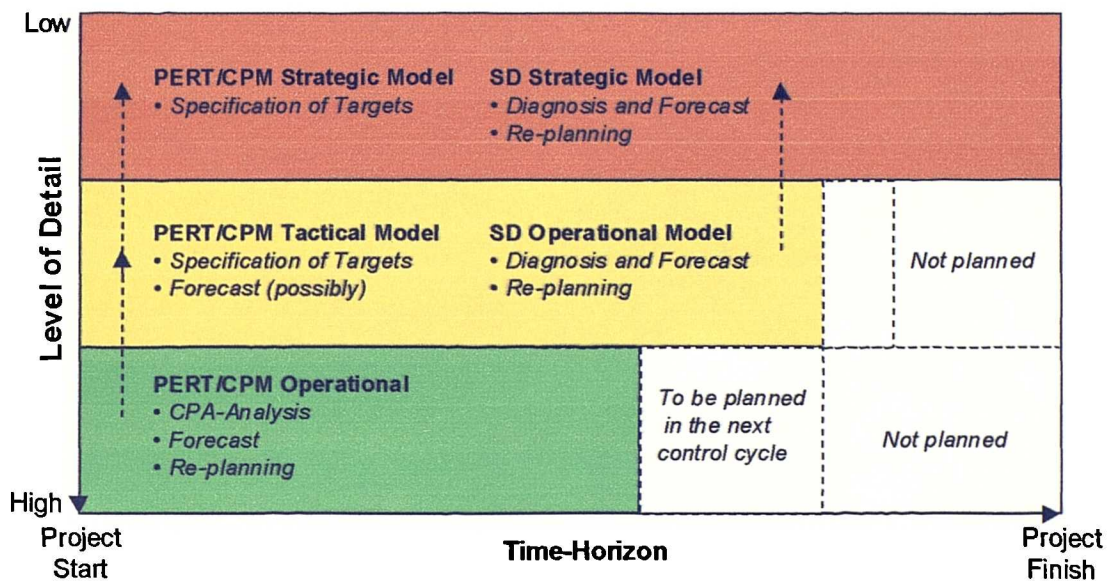


Figure 7.2 – The complete SYDPIM scenario for using the models

In total five models can be considered, which raises the serious issue of the full integration of all models. Regarding the PERT/CPM models, most software tools that implement this type of modelling support the combined use of various plans at different levels of detail into a single model. The same is not true for the SD modelling tools which are not specialised in project management models. Therefore, the integration of the two SD models will have to be ensured by the user. In SYDPIM it is proposed that this is done through analytical links similar to the ones used to formally integrate the SD model with to the PERT/CPM model – these will be described below.

For the sake of simplicity, the formal model of the SYDPIM process logic will be described considering a scenario where only two models are used: the PERT/CPM and the SD model. It will be assumed that the PERT/CPM model will cover the full life-cycle of the project, including its past segment, and encapsulates the three plans (strategic, tactical and operational), with their own time-horizons. In this way, the project past is available at all levels of detail and the full project future is also available but a different levels of detail over-time. Regarding the SD model, it will also be assumed that it covers the full project life-cycle and hence it will be a strategic model. This scenario assumed for the specification of the SYDPIM formal model is shown in figure 7.3 below:

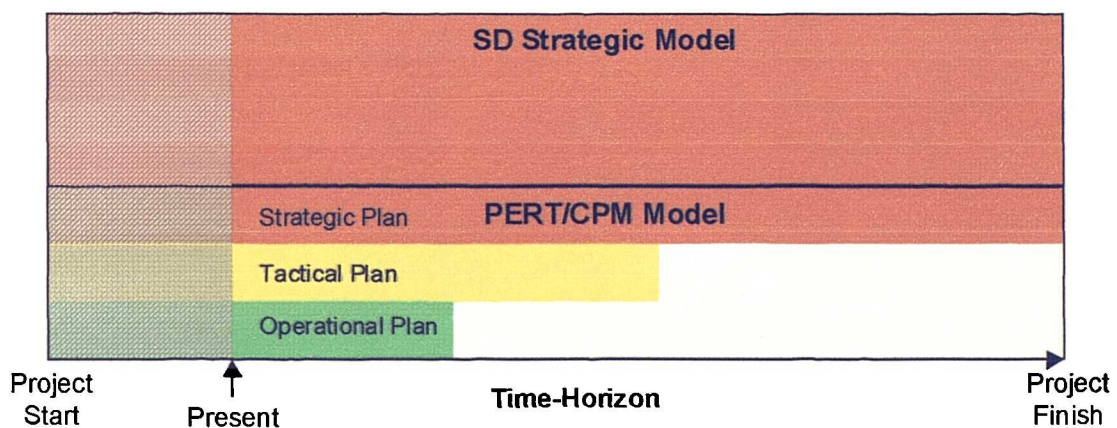


Figure 7.3 – Simplified SYDPIM scenario assumed in this work

Another very important aspect related with the use of the models is the frequency with which the model is updated. Ideally, a PERT/PCM model should be “continuously” updated (i.e. in every control cycle) with actual results before being used for planning purposes. This is because actual results can have a great impact on the project future. Re-planning the project future in the PERT/CPM model disregarding the past results can be misleading: the PERT/CPM analysis is based on the critical path and on the criticality of activities which is also affected by resource availability. Delays in the completion of non-critical tasks and consumption of resources above the planned profile may change the critical path and the criticality of future tasks. In practice however, there are cases where the PERT/CPM model is not updated properly or its update is behind the project timeline. In such cases, the PERT/CPM model becomes a passive mechanism for recording history. The development a new PERT/CPM plan half-way through the project disregarding its past is sometimes also considered, but in general that only

happens when a dramatic major change takes place, as if a new project was being started – otherwise, keeping a meaningful WBS would be difficult. Finally, in some cases a PERT/CPM model may only be used to control only a portion of the project work – i.e. a sub-project.

The assumption in SYDPIM is that the PERT/CPM model is updated with past results whenever it is used for planning and prior to that. This is regardless of whether the update and re-planning takes place in constant or varying periods of time – i.e. whether the period of the control cycle is constant or not. Regarding the SD model this is also always updated with past results before being used for planning purposes. In SD terms, this update consists in calibrating the model to reproduce the project past behaviour. Updating the SD model is even more critical for the accuracy of the forecasts than with the PERT/CPM model. For example, one of the assumptions of a project model is that a certain amount of defects or rework remains undiscovered throughout the project life-cycle. Calibrating the SD model to reproduce the project past up to present is essential to generate the right amount of undiscovered rework in the project in the present moment. Of course, this will have a great impact on the project future. The same effect applies to other intangible model variables, such as “current staff fatigue” – it depends on the past and it will affect the future.

The process logic of the traditional control cycle

As mentioned, the process logic consists of an underlying step-by-step algorithm which articulates the other three elements of the SYDPIM model. This will be described through a simple diagram which interconnects the activities, the objects, and the analytical links. Let us first describe the process logic underlying the traditional control cycle. This is shown in figure 7.4.

The notation used is straightforward and will be explained as required. Activities are represented in green rectangles. They are linked through straight arrows indicating the process flow and are numbered accordingly. The 3D-rectangle in blue represents an object which is a model, in this case the PERT/CPM model. The activities will either read or store information in the objects and this is represented through dashed curve lines linking an activity to an object. Possible branches

throughout the path of activities are represented by a yellow box. Points of exit or points of entry to the process are represented in grey rectangles and are not numbered.

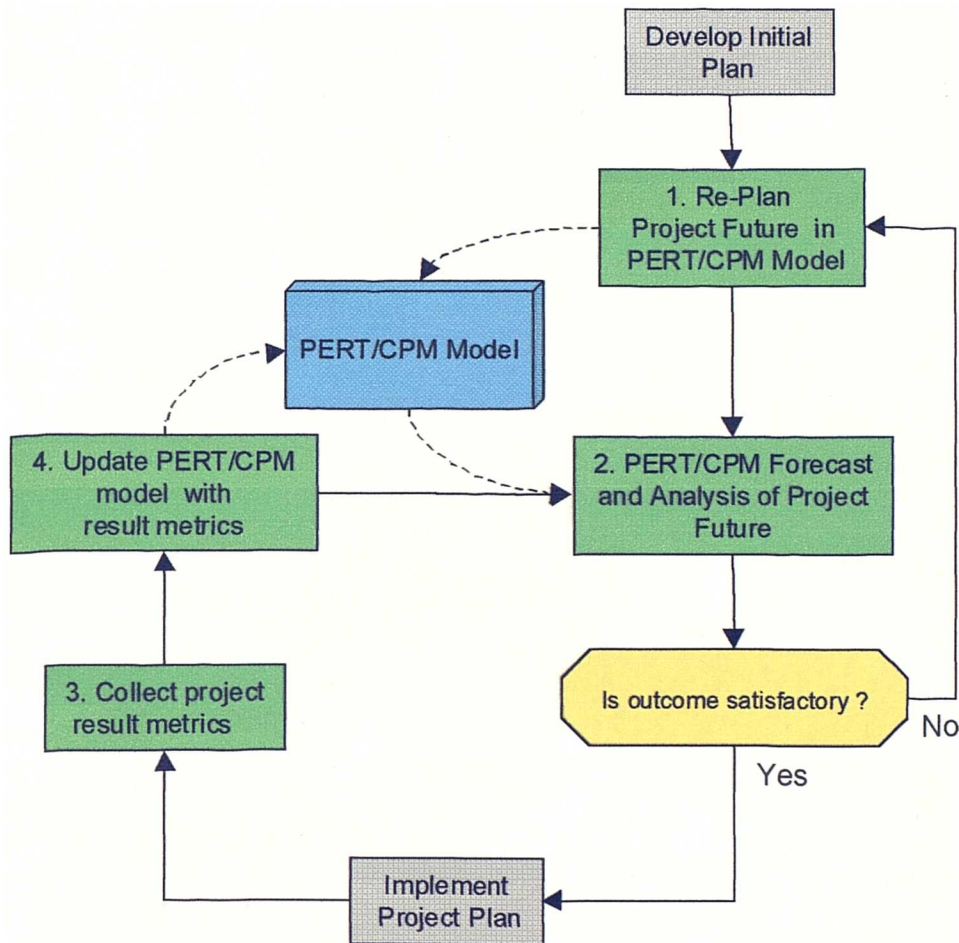


Figure 7.4 – The process logic of the traditional project control cycle

The process logic of the traditional control cycle is fairly simple. In planning, possible alternatives are generated in the PERT/CPM model and their impacts on the project outcome are analysed through critical path analysis. If the outcome is not satisfactory, the plan is revised, otherwise it is implemented. Once the control period elapsed, data about actual results is collected and the past segment of the PERT/CPM plan is updated accordingly. The impacts of these results on the project outcome are analysed, again through a critical path analysis. The cycle is repeats again.

The process logic of the SYDPIM framework

The process logic of the SYDPIM framework is considerably more complex than the traditional control cycle. This is because there are now two models exchanging data and being used for similar purposes. Before describing this process logic in detail, let us have a look at a high level view of the framework.

Basically, the framework consists in adding the use of a SD project model in both planning and monitoring. In planning the SD model provides an *alternative* to re-plan and forecast the future. Therefore, a plan can be developed and tested in both models. The resulting process considers that the user should be able to alternate from one model to the other. In monitoring, the SD model is also updated with results and used for diagnosis of past performance, a function that generally is not performed in the PERT/CPM model.

The high level process logic is described in the figure 7.5. Another object was added to represent the SD model. Where necessary, objects are duplicated using dashed lines. The principle behind the process logic in planning is that once the project work plan is revised or developed in one model, it is immediately tested in the other model. Therefore, a plan is only accepted for implementation if its outcome was analysed in both models, it is acceptable *and* the two models are consistent one another (meaning they are representing the same plan). This required two new activities that allow for one model to incorporate the project plan from the other model: (1) the SD model is calibrated for the PERT/CPM plan, or (2) the PERT/CPM plan is readjusted for the SD plan. These two activities are critical as they rely on the analytical links established between the two models. These links are not represented in this diagram. In monitoring, the SD model is used to diagnose the updated project past. This also required a new activity that makes use of the analytical links: the SD model is calibrated to reproduce the project past behaviour as implied in the PERT/CPM model.

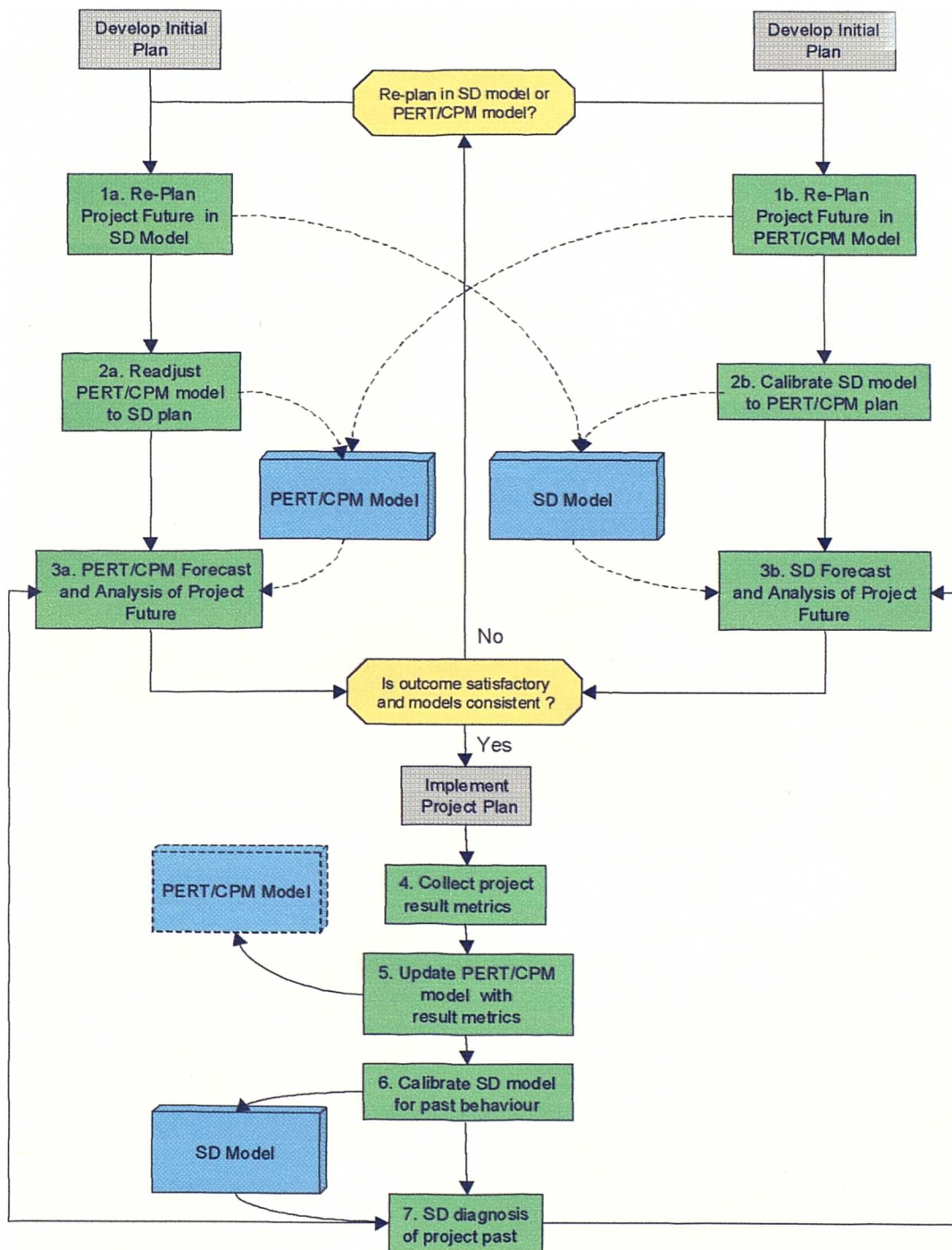


Figure 7.5 – High-level view of the process logic of the SYDPIM project control cycle

Once monitoring is complete, the process re-enters the planning function where the new project outcome is analysed and the plan is eventually revised. As the process logic indicates this should be done in *both* models. Once they provide a new

forecast of the project future (eventually different ones), the plan is then revised using one of the two models and tested on the other. In SYDPIM it is recommended that re-planning is first carried out in the SD model, but the user may decide otherwise. This recommendation stems from the fact that the SD model is richer and likely to provide a more reliable plan – note that only a *steady* plan is accepted (i.e. exhibiting stale targets).

This process logic allows for alternative paths to be followed throughout the activities. The numbering of the activities was adjusted to reflect this. It is also assumed that SYDPIM is implemented from the beginning of the project. Whenever SYDPIM is introduced half-way through the project, it is recommended the following initial steps are implemented:

- (1) SYDPIM planning is implemented based on the initial PERT/CPM plan, as if the project was just starting. This ensures that the SD model will reproduce the successful implementation of the initial PERT/CPM plan;
- (2) SYDPIM monitoring is implemented in the present moment. This ensures that the SD model will reproduce the whole project past up to present. This is fundamental for the validity of both diagnosis and future projections produced by the SD model.

The SYDPIM process can then proceed from activity 7. This diagram does not show explicitly the use of the analytical links and it also omits other important details of the process logic. This is now explained in full detail, separately for the monitoring and planning functions. Let us start with monitoring.

The process logic of SYDPIM monitoring

As mentioned above, the SD model is used in monitoring to diagnose the project past behaviour up to the present moment. This requires that the SD model is calibrated to reproduce this behaviour as reflected in the PERT/CPM model, which was just updated with actual results. This calibration requires the use of the SYDPIM analytical links.

An important issue in SYDPIM monitoring is the implementation of the SYDPIM metrics plan (SYMP), which is proposed in the methodology. The SYMP specifies a set of metrics to be collected in monitoring together with some other metrics to be derived from the SD calibration and diagnosis. There are various benefits in implementing this plan. First, the measurement achieved is *per se* insightful and helps in control. Secondly, the metrics database generated provides important support to the formal implementation of SYDPIM in two ways: (1) it provides a more comprehensive set of behaviour patterns to describe the project past, and (2) it provides metrics to support the calibration of the SD model. Both benefits are likely to improve the validity of the calibration of the SD model and thereby the accuracy of its diagnosis and forecasts. The implementation of this metrics plan is not mandatory but it improves considerably the whole SYDPIM project control process.

Another relevant issue is the availability of an updated PERT/CPM plan. When the project management framework within the organisation is not sufficiently structured, it may happen that when management wants to use the SD model, the PERT/CPM model does not contain an updated project plan. Preferably, this should not happen when SYDPIM is being implemented. The best solution will be to update the PERT/CPM model up to present – this can be effort consuming. Another alternative solution is to implement monitoring without using the PERT/CPM model. As it will be seen, while this is not recommended, it is a possible scenario within the process logic of SYDPIM monitoring.

The detailed process logic is shown in the figure 7.6.

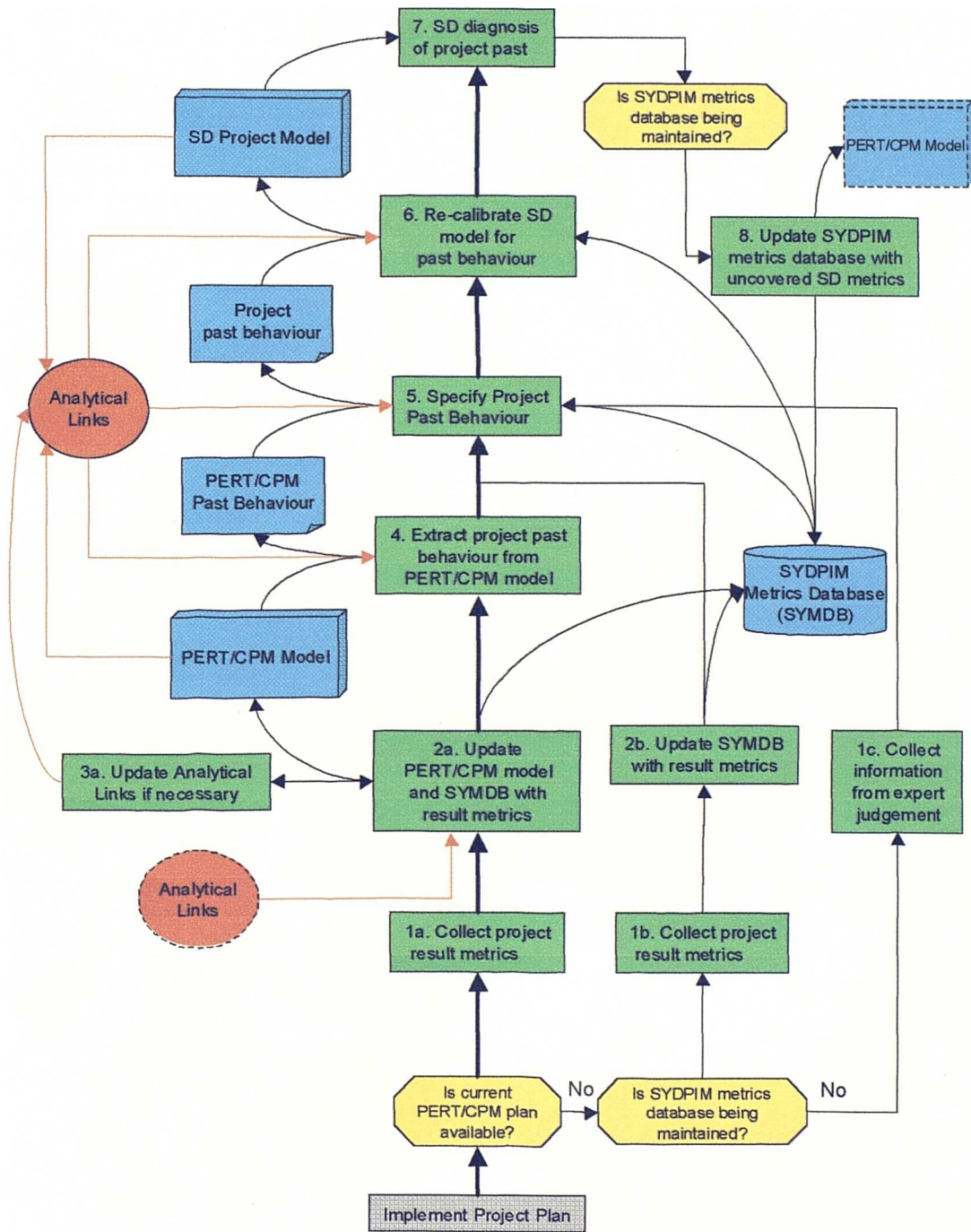


Figure 7.6 – Detailed process logic of the SYDPIM monitoring function

Once the current project plan was implemented for the last control period (e.g. a month), the first step is to collect data and update the PERT/CPM plan, just as in the traditional control framework. First it is considered whether a current PERT/CPM plan is available. Preferably this should be the case. If so, result metrics are collected at the PERT/CPM task level in activity (M1a). In practice, the specific

metrics to be collected depend on how the PERT/CPM model is used by the organisation and on the metrics plan being implemented. As it will be seen SYDPM considers a default set of metrics – this will be discussed in more detail when this activity is specified. The next step is to update the PERT/CPM model with these results as well as the SYDPIM metrics database (SYMDB), in case the SYMP is being implemented. This is done in activity (M2a). Some of the metrics to be stored in the SYMDB can be derived from the PERT/CPM model. Since these are considered at the SD-Task level, the use of the analytical links is required. Updating the PERT/CPM model may also require that the existing analytical links with the SD model are updated. A typical example is when new tasks were added to, or removed from the project in the course of implementation during the last control period. As it will be seen, PERT/CPM tasks are mapped through structural links to the SD-Tasks. Therefore the PERT/CPM tasks eliminated need to be removed from the SD-Tasks to which they were mapped to. Similarly, the new PERT/CPM tasks need to be mapped to a SD-Task. This way, if necessary, the analytical links between the two models are updated in activity (M3a) – the analytical links are represented in the diagram as a circle (in red); the dashed lines indicate that the activity updates the links; the straight lines linking the SD and the PERT/CPM models (objects) with the analytical links indicate that the links depends on the current status of these models; therefore, if one of the models is updated, the analytical links may also need to be updated.

If a current plan is not available in the PERT/CPM model, either the more recent version is updated up to present through activities (M1a) and (M2a), or the PERT/CPM model is not used in monitoring (this time). In this case, if the SYMDB is being implemented, various result metrics are collected in activity (M1b) and the database is updated in activity (M2b). These two activities differ from activities (M1a) and (M2a), because they consider explicitly that the PERT/CPM model is not used. They provide an alternative path within the process logic, to ensure that the data necessary in the following activities is collected – e.g. to later derive the project past behaviour and calibrate the SD model for this behaviour. If the SYMDB is also not being implemented, then the final alternative path is to collect information from expert judgement (management, technical staff, etc.) in activity (M1c).

Assuming that the PERT/CPM model was updated in activity (M2a), the next step is to extract the project past behaviour from the PERT/CPM model in activity (M4). This activity has two main inputs: the PERT/CPM model and the analytical links, and it has one output: the PERT/CPM past behaviour (i.e. the project past behaviour as extracted from the PERT/CPM model). The PERT/CPM model contains the data necessary to specify the behaviour patterns over-time, and the analytical links specify formally how these patterns are to be generated. For example, for each SD-Task, the PERT/CPM tasks mapped to it will provide the necessary data to generate the cost patterns. An important factor that affects the behaviour patterns that can be extracted from the PERT/CPM model, as well as their granularity, is the number of past versions of PERT/CPM plans available within the PERT/CPM model; in particular, whether the initial plan is available – this issue will be discussed later when this activity is described in more detail. The full set of behaviour patterns extracted from the PERT/CPM model are stored in an object called “PERT/CPM past behaviour”. A relevant issue in this SYDPIM activity is whether this object is preserved from one control cycle to the next, or whether it is re-generated whenever this activity is performed. The difference is that in the first case the “PERT/CPM past behaviour” is always available and so this activity consists of updating only the present data-point. In the second case, the patterns are always re-calculated from the beginning of the project. If this process is not automated through a software application, the second scenario can be very time consuming. Both scenarios are acceptable within SYDPIM.

The next step in activity (M5) is to specify the “project past behaviour”. This is based on the “PERT/CPM past behaviour” and on the metrics available in the SYMDB, which, if available, was updated in activities (M2a) or (M2b). The “project past behaviour” is considered as a more comprehensive set of behaviour patterns than those that can be extracted from the PERT/CPM model. An example is the pattern “cumulative number of defects detected”, or the pattern “cumulative effort spent with rework”, which are not tracked in the PERT/CPM model. The three inputs to this activity are the “PERT/CPM past behaviour”, the SYMDB and the analytical links. The output is the object “Project past behaviour”, containing an updated set of behaviour patterns which characterise the project outcome up to present. In principle, if the SYMDB is not being implemented this would be the same as the “PERT/CPM past behaviour”. However, expert judgement and other

information can be used to derive other patterns of project past behaviour. This is considered explicitly in activity (M5), in the case neither the PERT/CPM plan is being implemented nor the current plan in the PERT/CPM model was updated. However, expert judgement can always be used in activity (M5) to derive other patterns perceived relevant.

In the next activity (M6), the SD model is re-calibrated to reproduce the project past behaviour. There are three inputs to this activity: the object "Project past behaviour", which provides the set of behaviour patterns to be reproduced by the SD model; the analytical links, which identify which SD model variables should match which behaviour pattern extracted from the PERT/CPM model – this is specified through output-output data consistency links; finally, the SYMDB provides important process metrics to support the calibration of the model (e.g. observed "defect detection rate"). The process of calibrating the SD model depends on the specific model being used. The full formalisation of this activity would imply automated calibration. Not only this is a "technology" currently not available, as it would also depend on the specific model. Some semi-automated guidelines are provided in the detailed description of this activity.

Once the SD model is reproducing the project past behaviour, it is then used in activity (M7) for diagnosis. This diagnosis should consider three main type of analyses:

- explain *why* the project outcome is as described by the behaviour patterns – typically this consists in tracing back the causes that affect the shape of the behaviour patterns. In a project model, the typical relevant causes are the endogenous effects on productivity, QA activities, defect generation and defect rework activities;
- enhance progress visibility by *uncovering* metrics about the project status – the SD model will contain important variables about the project status which are intangible or which are not measured due to practical constraints. By uncovering these metrics, the visibility of the current project status can be considerably improved. For example, by knowing how many defects still remain undiscovered in the system;
- retrospective what-if analysis and process improvement – testing what-if scenarios in the past allows management to know whether better or worse

results would have been achieved had other decisions been taken. This type of analysis can be the basis to identify specific actions for process improvement.

It is important to note that this type of project diagnosis could not be performed using the PERT/CPM model alone. The only input to this activity is the object “SD project model”, which provides the required “management laboratory”.

In case the SYDPIM metrics database is being implemented, then metrics about the project status which were uncovered in activity (M7) are stored in the database for future use and reference. This is done in the following activity (M8). Some PERT/CPM software tools allow extra data to be stored in the PERT/CPM tasks for management information (through the creation of task fields). In this case, management may also wish to update PERT/CPM tasks with uncovered metrics (since this is not a SYDPIM requirement, these are not used to specify analytical links with the SD model). This possibility is represented through the dashed curve arrow from activity (M8) to the PERT/CPM model.

Overall, the process logic of the SYDPIM monitoring function is based on the principle of updating the PERT/CPM model with actual results, then extract from this model the project past behaviour patterns and calibrate the SD model to reproduce this behaviour, and finally diagnose the project status using the SD model. This is clearly based on “a link” between the PERT/CPM and the SD model: both have to produce the same project behaviour. This also emphasises the added value of the SD model, since this type of diagnosis could not be performed in the PERT/CPM model. This monitoring process is enhanced with the implementation of a metrics plan (SYMP), which provides a more comprehensive description of the project past behaviour and supports the calibration of the SD model. The analytical links are at the core of the whole process, keeping both models consistent one another. These links may need to be updated themselves due to possible structural changes in the PERT/CPM current plan. The SD diagnosis feeds-back uncovered metrics into the SYMDB (and eventually into the PERT/CPM model) providing an enhanced recording of the project history. Overall, the formal process logic of SYDPIM monitoring considers eleven activities, six objects (including the analytical links) and three possible branches.

Given the project past behaviour and its diagnosis, the next step is to analyse the likely outcome of the project. This is done in the SYDPIM planning function described in the following section, where, once more, the two models are used in a formally integrated manner.

The process logic of SYDPIM planning

The use of the SD model in planning is primarily aimed at forecasting the outcome of an existing PERT/CPM plan and improving its performance. This PERT/CPM plan specifies the project targets and how the work is to be accomplished. The SD model simulates the implementation of this plan, producing the likely project outcome. This simulated outcome may or not deviate from the targets. If unsatisfactory, adjustments to the project plan can be tested in the SD model and the new plan can be transferred back to the PERT/CPM model. This framework requires the use of analytical links: first, the initial plan in the PERT/CPM model needs to be represented in the SD model, and then the improved project plan in the SD model needs to be transferred back to the PERT/CPM model.

An important issue in SYDPIM planning is that “testing” a project plan in the SD model is not limited to simulating and forecasting the likely outcome of the project. Overall, there are four types of “testing” analysis that can be carried out in the SD model:

- *forecasting* the likely outcome – this is done by simply simulating the project plan. The SD model will describe the project’s final results and how the outcome unfolds over-time (e.g. how the final schedule slippage evolved over-time);
- *diagnosing* the likely outcome – like in monitoring, the SD model will allow the user to find out *why* the likely outcome is as forecasted. Typically, this is done by identifying what is affecting over-time the key parameters related to the course of the work. This diagnosis is the starting point for devising re-planning actions and improve the plan;
- *uncovering* assumptions – when the SD model is calibrated to reproduce the project future behaviour as portrayed by the PERT/CPM plan, important underlying assumptions regarding the work process have to be made explicit (e.g. productivity, defect generation rate). This analysis provides an assessment of whether the project plan is based on realistic assumptions;

- assessing the plan's *sensitivity to risks* – various scenarios can be tested in the model, representing the occurrence of possible risks (e.g. Client changes). Once the SD model is calibrated to represent these risk conditions, a new forecast of the project outcome is produced. This analysis provides an assessment of the plan's robustness against likely risks;
- testing *re-planning* actions – various re-planning scenarios, which will often include mitigating actions against risks, are represented and tested in the model (e.g. re-scheduling of completion dates). The SD model will forecast the outcome of the project, thereby assessing the effectiveness of these re-planning actions.

The general principle underlying SYDPIM planning is that a project plan is first made available in the PERT/CPM model and is then transferred to the SD model for testing prior to implementation. The SD model works as an experimental laboratory to test the PERT/CPM plan. However, the integrated use of the two models within SYDPIM planning can follow other alternative “paths”. For example, a plan can be first developed in the SD model and only then transferred to the PERT/CPM model, in the form of a logical network. SYDPIM provides a flexible planning process logic where both models can be used to develop a project plan and forecast the project future, and where the analysis of such plan can alternate from one model to the other. Whatever the path followed, there are two principles that must be verified in SYDPIM planning:

- (1) *mutual testing* – once a project plan is developed or readjusted in one model, it must then be tested in the other model. This ensures that no plan follows to implementation without being “approved” by the analysis of the two models. This is important because the two models provide a different perspective about the project outcome;
- (2) *model consistency* – the two models must represent the same project plan before implementation proceeds. In particular, it is not enough to test a PERT/CPM plan in the SD model. *The SD model must be calibrated for the successful implementation of this plan.* Of course, this ensures that the underlying assumptions are uncovered. The two models are consistent when they represent the successful implementation of a same plan.

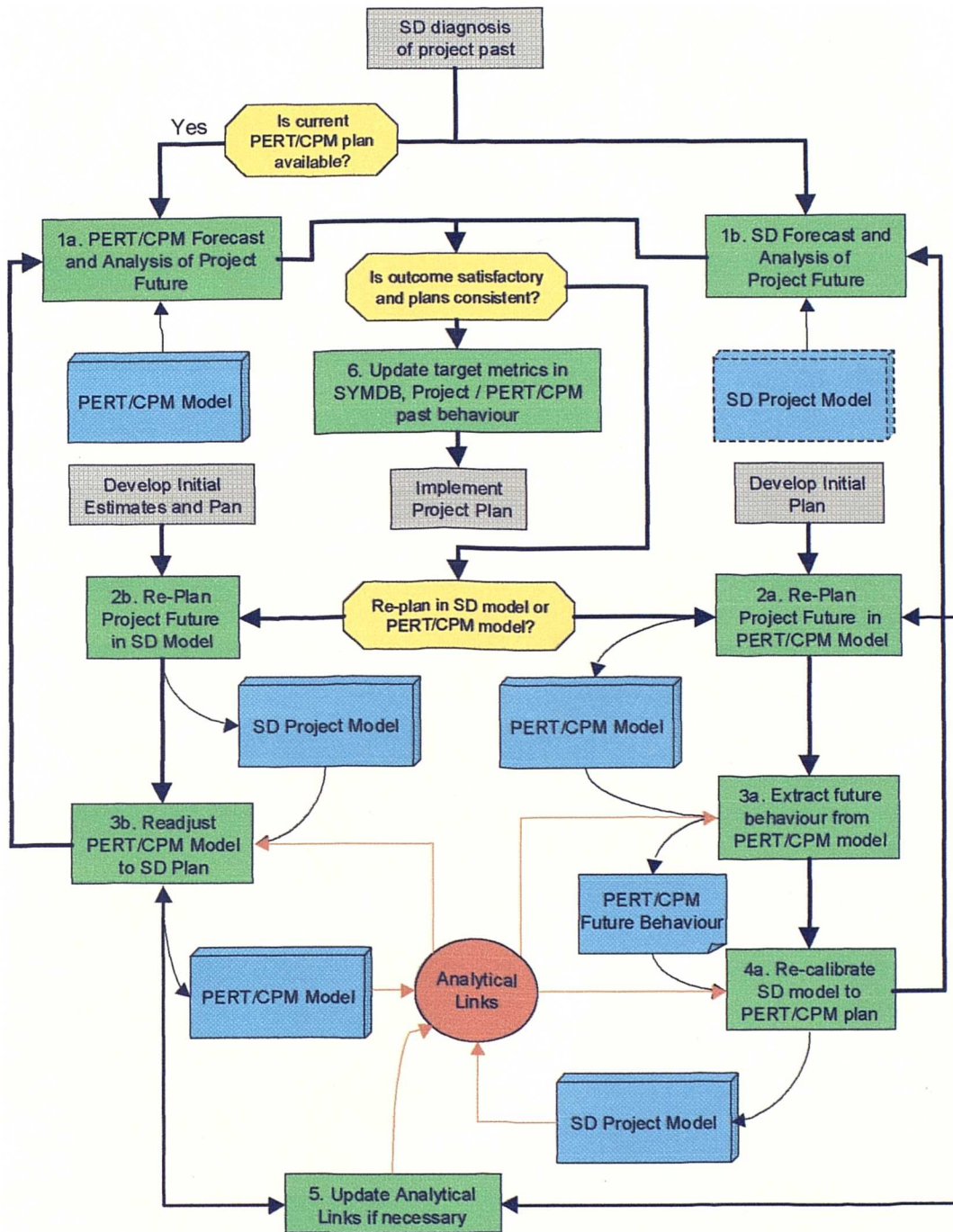


Figure 7.7 – Detailed process logic of the SYDPIM planning function

The detailed process logic of SYDPIM planning is shown in figure 7.7. There are three possible entry points to the (re)planning process (grey boxes): (i) the models have just been updated with the actual results in monitoring and (ii) an initial plan is to be developed in the PERT/CPM model or (iii) in the SD model. If an initial plan is to be developed using the SD model, then the model is first used to develop high-

level estimates for the project (e.g. project overall duration) and based on these estimates a high-level *steady* plan is proposed. The process logic then follows to translate and analyse this plan in the PERT/CPM model. If an initial plan is to be developed using the PERT/CPM model, then possibly a front-end estimating technique or tool was previously used develop the initial high-level estimates upon which the PERT/CPM network is developed (e.g. SLIM, KnowledgePLAN; see chapter 2). The process logic then follows to translate and assess this plan in the SD model.

When the process is entered from monitoring, the SD model was just calibrated with the project past (typically, the implementation period of the last control cycle). In most cases, the PERT/CPM model was also updated with this past (though not necessarily; see previous discussion in SYDPIM monitoring). The first step in (re)planning is to forecast and analyse the project future in both models. This is represented in the “parallel” activities (P1a) and (P1b) (the order is irrelevant). Of course, activity (P1a) is only performed if a current PERT/CPM plan is available updated with the past. It is important to note that the analyses of the project future in these activities should also consider the likely emergence of risks which may interfere with the project outcome. For example, if a Client has just requested scope changes then the models should be adjusted to reflect this.

In most cases, when the planning process is entered first time, the outcome of the project future will be different in the two models. This emphasises that different aspects of the project are considered in each model. When this happens the models are not consistent for the future segment of the project. It may also happen that the project future outcome is not satisfactory. After some re-planning, the models will become consistent and the project future outcome is satisfactory. When this is achieved, the planning process exits and the current project plan proceeds to implementation.

When re-planning is required, there are two possibilities: the plan is either readjusted in the PERT/CPM model or in the SD model. SYDPIM recommends the later option because in principle the SD model will provide a more realistic analysis of the performance of control actions. Re-planning takes place in activities (P2a) or (P2b) respectively – note that these activities should not be implemented in parallel

because two “solutions” would be developed separately, eventually leading to unproductive rework planning efforts. These re-planning activities include the forecast and diagnosis of the project outcome under various planning scenarios, as well as the assessment of risks and their impacts.

If re-planning takes place in the SD model the process within activity (P2b) should finish when a satisfactory and *steady* plan is developed. A satisfactory plan means that the outcome is acceptable. The plan being *steady* means that when the implementation of the plan is simulated in the SD model, the outcome does not deviate from the targets. SYDPIM advocates that only a SD plan with steady behaviour should be considered for implementation. As an output, this activity adjusts the project plan in the SD model, as shown in the diagram by the dashed curve arrow.

The next step is to readjust the PERT/CPM model so that it represents the new project plan developed in the SD model. This takes place in activity (P3b), where the analytical links are required. Readjusting the PERT/CPM model to become consistent with the SD model implies that the input data, the output data and the structure of the models are consistent one another. In most cases, this requires both data and structural readjustments to the current PERT/CPM plan (changing resource allocation to the tasks, start and completion dates, dependencies between tasks, adding or removing tasks, etc.). The ideal of this activity would be an automated algorithm. Given an existing PERT/CPM plan this would be automatically readjusted to become consistent with the new SD plan. There are various critical issues regarding this activity which will be discussed later in more detail. At this stage it is important to note the following points:

- (1) an automated algorithm requires the specification of a particular SD project model (or at least a very specific set of characteristics), and the specification of a particular PERT/CPM modelling approach (e.g. are QA tasks considered explicitly, or is QA effort implicitly incorporated in development tasks);
- (2) this process implies dis-aggregation of SD data and structure into a more detailed level; this is because the PERT/CPM plan assumes a more detailed view of the project work. The main consequence is that there is more than one solution at the PERT/CPM plan level for a given SD-plan – i.e. a “one-to-many”

- type of relationship. Automated decomposition requires assumptions which are unlikely to be correct without the input of human judgement;
- (3) re-planning at the operational level in the PERT/CPM model typically involves many “day-to-day” specific constraints, with which a fully automated algorithm would hardly cope (e.g. compatibility of team members); again, human judgement is an essential ingredient to cope with these issues.

Some guide-lines and strategies to a semi-automated process are proposed in the description of this SYDPIM planning activity. The analytical links are critical to this process: (i) data consistency links ensure that the models are producing the same outcome and are based on consistent assumptions; (ii) data transfer links ensure some level of automation; (iii) by default, structural links are the basis to specify the data-links and ensure structural consistency; they can also be used to implement some level of automation; (iv) data-structural links can be used to support some level of automation. In the diagram, the use of the analytical links is represented through the dashed arrow.

Readjusting the PERT/CPM model not only requires the analytical links but it may also imply changes to these links. For example, if new tasks were added to the PERT/CPM plan in order to represent a scope increase suggested by the SD plan, then this task will have to be mapped to a SD-Task in the SD model. Similarly, if precedence relationships were added or removed in the PERT/CPM plan an update of the analytical links is required. The update of the analytical links takes place in activity (P5), as shown by the dashed arrow.

Once the current PERT/CPM plan was readjusted to represent the SD plan, the next step is to analyse the likely project outcome in the PERT/CPM model, based on the conventional critical-path analysis (CPA). To an extent this could be considered as part of readjusting the PERT/CPM plan in activity (P3b). However, for the sake of clarity and discipline, the process logic cycles back to activity (P1a) where a more comprehensive critical path analysis can be carried out (e.g. *Monte Carlo* simulation analysis; see chapter 2).

At this stage in the process logic, a new plan was developed in the SD model and was then translated to and analysed in the PERT/CPM model. If the project

outcome is satisfactory under the PERT/CPM analysis, then the process may exit and the new plan is implemented for the next control cycle. However, it may happen that a work plan which appeared satisfactory in the SD model may look inappropriate at the PERT/CPM operational level. For example, short floats overall and too many near-critical paths may indicate a risky plan. In this case, re-planning is needed once again. This can be done in the SD model or in the PERT/CPM model.

Let us now consider the path where re-planning takes place in the PERT/CPM model. This takes place in SYDPIM activity (P2a), which can also be the entry-point to the planning process, when a project plan is being developed for the first time. PERT/CPM (re)planning typically consists of scheduling and allocating resources so that acceptable schedules and costs are achieved for the project major milestones – typically, a late project is compressed to the left. The output of this activity is the update of the future segment of the current plan in the PERT/CPM model. As imposed by the SYDPIM principle of “mutual testing”, this plan should now be tested in the SD model. Before that it is necessary to check whether the re-planning actions in the PERT/CPM model require an update of the analytical links. Once more, new tasks and dependencies might have been added or removed from the current PERT/CPM plan. This is done by “calling” activity (P5), as shown in the diagram.

In order to test a PERT/CPM plan, the SD model needs to be calibrated for that plan. This implies reproducing the project future behaviour *implicit* in the plan. This behaviour is “implicit” because in the PERT/CPM approach the concept of “project behaviour” as a set of over-time continuous patterns is not considered. Most PERT/CPM tools will not produce a comprehensive set of patterns. The next step is therefore to extract the project future behaviour portrayed by the PERT/CPM model. This takes place in the following activity (P3a). As shown in the diagram, the project future behaviour is stored in an object called “PERT/CPM future behaviour”, which is the output of this activity. A new set of patterns (and hence a new object) is generated whenever the future segment of the current PERT/CPM plan is changed for re-planning purposes. Like in monitoring, the analytical links are an input to this activity. Again, some patterns require that more than one plan is available in the PERT/CPM model – this will be discussed later in the detailed description of this

activity. It is important to note that the type of project behaviour extracted from the PERT/CPM model will be *steady* – i.e. constant targets over-time. This is due to the nature of a PERT/CPM plan, according to which the project outcome is expected to be as specified in the plan (unlike in SD modelling where the simulated outcome may be different).

Once the PERT/CPM future behaviour is specified, the next step is to calibrate the SD model to reproduce this behaviour. This implies that a *steady* project behaviour will be imposed in the SD model. It is therefore critical that a SD project model is capable of reproducing this type of behaviour. This calibration is done in activity (P4a) which has as main input the PERT/CPM future behaviour to be reproduced. Within a formally integrated process, the ideal of this activity would be an automated calibration of the SD model to the PERT/CPM plan. However, there are again some barriers to this ideal: first, generic automated calibration of SD models is not developed yet (only some optimisation and partial calibration); secondly, if possible to achieve the algorithms would depend in the specific SD project model being used; and thirdly there will most likely be various possible “calibration solutions” and the data available in the PERT/CPM model will not be sufficient to select the “correct” one – e.g. will the schedule be achieved due to high productivity or due to low defect generation. Once more, human judgement proves essential to ensure that a model is representing reality as correctly as possible. However, optimisation techniques and any form of semi-automated calibration can and should support this activity, if available at all. This activity also makes use of the analytical links which, like in monitoring, are used to identify which SD model variables should match each behaviour pattern and provide the criteria to consider when behaviour reproduction is accurate enough (this is specified through output-output data consistency links). The output of this activity is the update of the SD model with a project plan consistent with PERT/CPM current plan.

Based on the principle of “mutual testing” mentioned above, the next step is to test the performance of the new plan in the SD model. The process re-enters activity (P1b), where the project future is forecasted and analysed. However, in this situation the SD model will *always* forecast a successful outcome for the project. The reason is simple: the SD model was calibrated to reproduce the *steady* behaviour portrayed by the PERT/CPM plan. The SD analysis will therefore focus

on uncovering the required assumptions underlying this plan, and which may prove unrealistic. Various risks can also be tested in the SD model to assess the plan's robustness. If the plan proves realistic and robust to risks, then the models are consistent and the outcome is satisfactory. In this case, the process exits and the plan is implemented for the next period of the control cycle. Otherwise, re-planning is required and once again this can be done either in the SD model or in the PERT/CPM model.

According to the SYDPIM planning process logic, re-planning can always take place in any of the two models. Overall the re-planning process may alternate several times between the two models until a satisfactory solution is achieved. However, the recommended path in SYDPIM is that a future plan available in the PERT/CPM model should always be tested and improved within the SD model, and then transferred back to the PERT/CPM model. There are various reasons that support this argument: first the SD model provides a richer analysis and a quicker and more effective way of identifying a reliable plan (note that the SD model is an experimentation tool, whereas the PERT/CPM model is more aimed at specifying the project work in detail); secondly, once developed the SD solution is more likely to be successful at the PERT/CPM level, where various alternatives exist to implement the SD aggregate solution, than the other way around.

Before implementation takes place there is still a final planning activity (P6) which needs to be performed. The purpose is to update target metrics in the SYDPIM objects SYMDB, "PERT/CPM past behaviour" and "Project past behaviour". This update has to do with the fact that the present value of this type of metrics (e.g. scheduled completion date) depends on both actual results and future plan. This present value was first collected, or estimated, during SYDPIM monitoring based on the actual project results and on the "old" plan. Since a new plan was now generated in SYDPIM planning, these values may need to be updated – this issue will be discussed later in more detail.

Overall, the planning process logic considers that once the project past was diagnosed, the impacts on the project future should be immediately assessed in *both* models. If the outcome proves unsatisfactory in any of the models, re-planning should take place in *only one* of the models, and the new plan should then be tested

in the *other* model. This process repeats until a satisfactory planning solution is developed. The planning process may also be entered where an initial plan needs to be developed for the project. Whatever the scenario, SYDPIM recommends that the SD model is used to generate a reliable new plan and that this is then converted to the PERT/CPM model, where the details of the solution are specified. This process of generating a solution in one model and then testing it on the other, is based on the use of the analytical links. Adjusting one model to the other would ideally be an automated operation but various constraints prevent this to be feasible. Nevertheless, there is scope for a semi-automated process based on human expert judgement as an essential input. Whenever the current plan in the PERT/CPM model is readjusted, the analytical links between the two models may also need to be updated. Once the two plans are consistent, representing the same project plan and producing the same project outcome, the plan is implemented for the next period of the control cycle after which the monitoring process is re-entered again. Overall, the process logic of SYDPIM planning considers eleven activities, four objects and three main branches.

As mentioned in the beginning of this section, the formal model of the SYDPIM framework consists of a formal specification of four main elements: the process logic, the objects, the analytical links and the activities. The process logic of the planning and monitoring functions of the SYDPIM project control framework was just described in some detail, as a step-by-step process comprised of a series of activities. Within these activities specific operations take place, exchanging data with the objects and making use of the analytical links. The analytical links depend on the characteristics of the objects “SD model” and “PERT/CPM model”. This way, the objects will be specified first in the next section. Following this, the analytical links and the SYDPIM activities will be specified.

7.4.2 The SYDPIM objects

Introduction

The formal model of the SYDPM framework considers certain entities that store and process information. These entities are here referred to as “objects”. The more important ones were identified in the process logic of the planning and monitoring functions, like the SD model and the PERT/CPM model. As it will be seen, other objects will be required in the detailed specification of the activities.

Some objects will have sub-components which is useful to consider as being objects as well. An object therefore may consist of a hierarchy of sub-objects which can be specified formally.

Objects store information in the form of numerical data. First, the data contents will be proposed for each object. Then the object is formally specified as a data-structure. A formal specification language is required for this. A software specification language is an appropriate candidate.

Being a data repository, it is normal that certain conditions have to be verified so that the information is “valid” in light of its meaning and practical application. For example, the dependencies of a network plan in the PERT/CPM model cannot lead to circular paths. For the object to be well specified, this type of conditions should be made explicit through formal specification. Logical expressions using a mathematical type of notation are the best candidates to implement this specification.

Objects are also useful to answer certain process queries. For example, it may be necessary to know how many predecessors a certain task has, or what are the tasks that currently form the critical path. The ability of an object to answer this type of queries depends on its data contents and how these are structure. In order to answer queries an object needs to have operators associated with it. Operators can be seen as logical processes which have inputs and produce outputs. For example, a given task can be the input and a set containing the predecessors tasks can be

the output. Operators will also need to be formally specified. A formal algorithmic language is an appropriate way to implement such specification.

Overall, the objects require a formal specification language to describe its data structure, conditions of validity and operators. A formal language is required because SYDPIM is a formal methodology. The value of this formality is an unambiguous definition of the SYDPIM process and of its requirements. This language could well be a formal software specification language used for the modelling and design of information systems. However, the potential disadvantage of such level of formality is the barrier to understanding and implementation at the eyes of the non-software expert. Here, it is important to note that SYDPIM is a process framework to be implemented within a social system. It is therefore required that its formality is well balanced with the methodology's ability of being flexible and adaptable to specific situations, persons and environments. For practical purposes of implementation, it is also important that this formal specification is accessible to the non-software specialist manager.

As an attempt to balance these opposing objectives, a simple semi-formal language is proposed. The aim is to provide rigorous and unambiguous definitions which are easy to understand and implement by a management-type of audience. This language is now described.

Formal specification language for objects

Objects have three components: (i) a data-structure, (ii) conditions of data-validity and (iii) operators. The notation used to specify an object's data-structure is based on the following principles:

- a data-structure can be composed of a hierarchy of sub-structures down to elementary data-fields;
- elementary data-fields can be of "primitive" types like a real number, or of structured types like a time-series of numbers.

The conditions of data validity will be specified using common mathematical notation with which most managers will be familiar with. The notation used to specify the operators is also straightforward: it specifies the inputs and the outputs of an

operator. The specific algorithm is not described since this is an implementation issue.

For example, let us consider the following simplified specification of the SYDPIM object "PERT/CPM model":

```

PERT/CPM model =
  Data-structure =
    Project plan =
      Tasks           : {taskk}
      Dependencies    : {(taski, taskj)}
      Resources =
        Personal information : {(resourcek, personal-info)}
        Project information  =
          Allocation         : {(resourcek, profile | R+)}
          Billing rate        : R+
          Task Alloc         : {(resourcek, taski, profile | R+)}
    Operators =
      Depends_on      : taskk x taski → T | F
      ...
    Validity =
      No_circle =
        IF Depends_on(taskk, taski) THEN NOT Depends_on(taski, taskk)
      ...
  
```

Object variables:

task_k = PERT/CPM model.Data-structure.Project_plan.Tasks[k]

Data-types:

```

Personal-info =
  Name       : string
  Phone      : integer
  Address    : string
  
```

```

Task =
  Start date : date
  Finish date: date
  
```

```

profile = {(t, R+)}
  
```

As expected, the object contains three main components: data-structure, operators and data-validity conditions. Let us analyse first the notation used to specify the data-structure.

The data-structure of this object has only one component called Project plan, which will contain the data that describes a network plan. It is composed by two data-fields, "Tasks" and "Dependencies", and by one sub-structure called "Resources". The "Tasks" data-field is specified as a set of tasks. The operator "{ }" is used to define a set and the index "k" is used to define various instances of "task" (the specific number of instances is not relevant). The data-type "task" is specified in more detail below as a sub-structure of data composed by two data-fields: start date and finish date, which type is "date". The "Dependencies" data-field is specified as a set of couples of tasks. The operator "(" is used to define associations, in this case couples. The indices "i" and "j" are used again to identify different instances of "task". The sub-structure "Resources" is specified as composed by a data-field "Personal information" and another sub-structure "Project information". The data-field "Personal information" is specified as a set of couples composed by the specific resource and the data element "personal-info" which is specified below. The sub-structure "Project information" contains three data fields. The "Allocation" data-field associates with each resource an allocation profile or a constant allocation level (defined as a positive real number "R+"). The operator "|" is used to specify a logical "exclusive or". The "profile" data-type is specified below as a time-series of positive real numbers. This time series is defined as a set of couples each formed by a time-point and a positive real number. The data-field "Task Alloc" specifies the allocation of the resources to the tasks. This is specified through a set of triplets formed by a resource, a task and a profile or a constant number.

This simple notation is based on the nesting of data sub-structures, on the use of the operators "{ }" and "(" to specify sets and associations, on the use of logical operators like "|", on well-known primitive data-types like "R+", and finally on the use of indices to specify instances. The generic variables "t", "r", "n" and "y" or "x", which may have indices, will be used to specify a time-point, a resource entity, an integer number, and a real number, respectively. For example, the expression $\{(r_k, \{(t_i, y)\})\}$ specifies a set of resources each with its profile over-time of the data-type "y", which will be used when is not adequate to impose a primitive data-type like "R+".

As it will be seen later, the formal specification of the analytical links and of the SYDPIM activities requires a direct reference to the contents of the objects' data-

structures – i.e. to its variables and components. The proposed specification language allows this to be done using a simple notation. For example:

- PERT/CPM model.Data-structure.Project_plan.Tasks – accesses the tasks in the project plan above;
- PERT/CPM model.Data-structure.Project_plan.Tasks[i] – accesses a specific task “ i “ of the project plan;
- PERT/CPM model.Data-structure.Project_plan.Resources.Project_Information.Task_Alloc[r, i, p[t]] – accesses the specific amount of the resource “r” allocated to the task “i” in the moment of time “t”.

This notation is simple and unambiguous but it tends to lead to long names depending on the level of nesting of the data-structures. In some cases, this may prevent good readability and easy understanding. To overcome this problem, shorter equivalent names will be used where necessary. In these cases, the true long names that specify the object’s contents will also be identified.

The operators are specified as functions with inputs and outputs. The notation used identifies the operator, its inputs and outputs with names that suggest their meaning. In this case, the operator specified is called “Depends_on” and takes two specific tasks as input and produces a logical “True” or “False” as an output. Clearly, this operator checks whether the first task depends, directly or indirectly, on the second input task. The specification of the algorithmic process of the operator is not described – in all cases presented in SYDPIM this algorithm is viable to implement and does not require any research novelty.

The validity condition specified, called “No circle” imposes that there can be no circular path of dependencies in the network plan. So if task A depends on B (directly or indirectly), then B can never depend on A (directly or indirectly). The notation used is based on well-known logical operators and clauses, and on calling the operators. This is straightforward to understand.

In most cases, the operators and validity conditions use variables that refer to the elements within the object’s data-structure. For the sake of simplicity short names are used to identify these elements but the true long names are identified below in “Object variables”.

Both operators and validity conditions can be referenced using the same notation used to access the elements in the object's data structure. For example:

- PERT/CPM model.Operators.Depends_on(task_i, task_j) – calls the operator “Depends on”;
- PERT/CPM model.Validity.No_circle – calls the validity condition “No circle”.

Once the required specification language was described it is now possible to specify the SYDPIM objects. There are three major objects used within the SYDPIM process logic:

- (1) the PERT/CPM model;
- (2) the SD model;
- (3) the SYDPIM metrics database.

There are three additional objects represented explicitly in the process logic:

- (4) the PERT/CPM past behaviour;
- (5) the project past behaviour;
- (6) the PERT/CPM future behaviour.

As it will be seen, objects (4) and (6) are sub-components of the object (1).

The required characteristics of each object depend on the specific circumstances in which SYDPIM is implemented, like data collection and data availability. Since there are many possible scenarios, a *basic mode* of SYDPIM implementation is proposed. This mode is based on various assumptions which will be described in the definition of the objects.

The definition of each object is presented in the following way:

- *overview* – summarises the functionality of the object within the SYDPIM framework and specifies its requirements within the proposed basic mode of SYDPIM;
- *proposed object hierarchy / data contents* – for the objects which are made of other sub-objects, a structured hierarchy of sub-objects is proposed. For the objects which work only as data repositories, the data contents are proposed;

- *formal specification* – both the object hierarchy and the data contents of the object are specified formally.

Each of these objects is now defined separately in the order presented above.

The PERT/CPM model

Overview

The PERT/CPM model will contain one or more project plans. The more important ones are the “initial PERT/CPM plan” and the “current PERT/CPM plan”. The first represents the PERT/CPM network plan that is developed for the project before the implementation starts, and so it can be considered as the project plan at time 0. This plan does not record any work progress. The current plan represents the PERT/CPM network plan that is currently being implemented. Unless it coincides with the initial plan it will record past progress. This way, the current plan is divided into two sub-components: past segment (work accomplished) and future segment (work remaining to be accomplished).

There are other plans that may be stored in the PERT/CPM model and represent past versions of the current plan that were modified as the work was re-planned over-time. Whenever a control cycle begins, the last current plan becomes a past plan. A new current plan is then developed from it within the control cycle: in monitoring actual results are updated and in re-planning the future work may be re-scheduled.

This dynamic view of a PERT/CPM plan evolving over-time is crucial to the concept of linking the PERT/CPM model with the SD model. For example, whenever the PERT/CPM plan is re-planned and its critical path is extended, the variable in the SD model that represents the planned completion date for the project should increase. In an ideal application of the PERT/CPM model, this model would contain a series of historical versions of the PERT/CPM plan from the beginning of the project up to present. From this series of PERT/CPM plans it is possible to extract various dynamic patterns of project behaviour considered in a SD model, in particular management decisions like schedule extensions. In practice, these past

versions are often not available but a current plan and an initial plan are normally available. Most PERT/CPM software tools will allow the control of historical versions of project plans.

As shown in the process logic of SYDPIM monitoring, a set of behaviour patterns describing the project past are extracted from the PERT/CPM model (see activity (M4)). This set of behaviour patterns are represented in an object called "PERT/CPM past behaviour". Likewise, in SYDPIM planning there is an object called "PERT/CPM future behaviour". In SYDPIM these are the two components of another object of the PERT/CPM model, called "PERT/CPM project behaviour", and they represent its past and future segments respectively. In general, the PERT/CPM software tools available do not provide a "behavioural" perspective of the project in the form of patterns over-time. At most, they will provide bar-charts and line-graphs that show few of the relevant patterns. However, because this object is solely based on data extracted from the PERT/CPM model, it is considered a sub-component of the model.

Proposed object hierarchy

The overall hierarchy of objects in the PERT/CPM model is represented in figure 7.8 below:

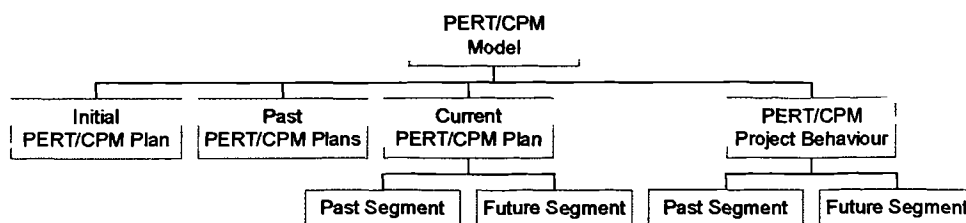


Figure 7.8 – SYDPIM object hierarchy of the PERT/CPM model

The series of past versions of the PERT/PCM plan in the model is considered as one single object. This is because, as it will be seen, there is no need to individualise any of these past versions.

It is important to understand how this object hierarchy based on the PERT/CPM model is used within the SYDPIM framework to provide a dynamic view of the project behaviour, and thereby to establish links with the SD model.

The traditional view of a PERT/CPM plan is one of a network plan being developed in the beginning of the project. This plan is implemented and continuously revised. Typically, it is the last version of the plan that matters because that is the one which specifies what is to be done in the future in order to achieve the current objectives. In a way, past plans were failed attempts. These versions are generally given little use if recorded at all. The SYDPIM view is different in this respect: the PERT/CPM plan is a dynamic entity which evolves over-time. The initial PERT/CPM plan is the first state of that entity. Whenever the plan is revised a new version is released and hence a new state has been generated for this entity. The dynamic evolution of the PERT/CPM plan is the project behaviour as described by the patterns of a SD model. The SD model will not contain the detailed evolution of the PERT/CPM plan over-time but it will produce patterns of behaviour that characterise this evolution. For example, when the SD model produces the pattern “Estimated Completion Date” changing over-time, this implies that the network plan is evolving (i.e. being revised), with the duration of its critical path changing according to this pattern.

Figure 7.9 describes this dynamic view of PERT/CPM planning based on the object hierarchy proposed above for the PERT/CPM model. The initial plan is developed in the beginning of the project and is stored in the correspondent object. In the beginning of the next control cycle, in monitoring this plan is updated and in re-planning it is eventually modified. A second release of the plan then issued and, for the next control cycle it becomes the “current PERT/CPM plan”, also stored in the appropriate object. In the beginning of the next control cycle, this second release becomes a “past PERT/CPM plan” and is stored in the object that will contain all past version of the PERT/CPM plan. After planning, a third release is issued becoming the “current PERT/CPM plan”. This process is repeated over-time throughout the project, with current plans becoming past plans as new releases of the current plan are issued.

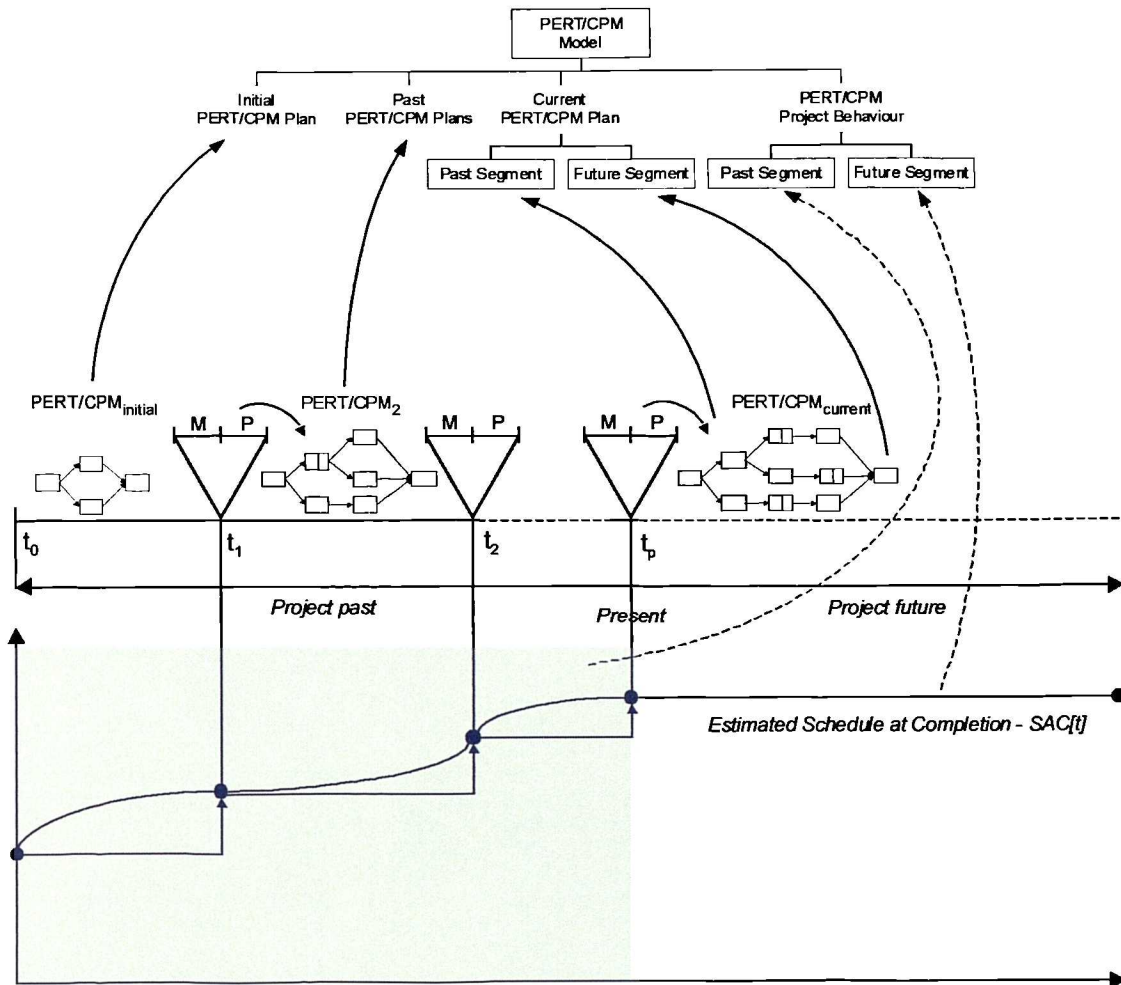


Figure 7.9 – Use of the PERT/CPM model object hierarchy: the evolution of the project plan and the project behaviour are stored in the model's sub-objects

Half-way through the project, at the present moment the current PERT/CPM plan will contain a past segment with tasks referring to work already accomplished, and a future segment with tasks that refer to work remaining to be accomplished. These two segments are stored in the correspondent sub-objects. As new versions of the PERT/CPM plan are released, new data-points for the patterns of the “past segment” of the “PERT/CPM project behaviour” can be generated, as shown in figure 7.9. The data-points for the patterns of behaviour of the “future segment” are derived from the “current PERT/CPM plan”, and will show a *steady* behaviour. These behaviour patterns are stored in the correspondent sub-objects of the “PERT/CPM Project behaviour”.

It is important to note that the patterns of behaviour generated within the PERT/CPM model will consist of discrete data-points. In figure 7.9, the pattern “Estimated

Schedule at Completion – SAC[t]” is shown as an example (other patterns will be generated). Patterns like this one will describe aspects of the project control decision-making process and will not evolve continuously from one data-point to the other. Instead, these patterns will evolve as steps. This is because the PERT/CPM control framework assumes that no re-planning takes place between two consecutive control points, while the current plan is being implemented (at least in modelling terms). On the other hand, the patterns that will be generated by the SD model will be continuous (as suggested by the dashed curve). This is because in a SD project model it is generally assumed that most, if not all, of the control re-planning actions are generated endogenously within the model in a continuous fashion. This highlights the contrast between the PERT/CPM discrete perspective versus the SD continuous view of the project.

Formal specification

As this framework suggests, the PERT/CPM model will contain sub-objects like “work plans” and “segments of project behaviour”. It is necessary to define what is exactly contained within each of these objects. The contents of the work plan will depend on the specific PERT/CPM tool used, on data availability, and on how the PERT/CPM model is used. Regarding the project behaviour, the specific patterns to be considered will also depend on the data-contents of the work plans and their of availability. While the contents of these objects are relative to tools and procedures, it is important to establish a set of assumptions. This is because, as it will be seen, the patterns of behaviour to be derived from the PERT/CPM model will be the basis to establish data-links with the SD model.

The reality of PERT/CPM based project control is that there is no universal tool and process commonly applied to implement this model. Therefore, in SYDPIM a standard mode of implementation is considered. This is based on some proposed assumptions which will be considered throughout the description of the methodology. These are as follows:

- (1) *components of a work plan* – a PERT/CPM work plan will comprise the following components:
- (i) elementary tasks (work packages);
 - (ii) dependencies between every two tasks of type “finish-to-start” – in most PERT/CPM tools currently available, other types of dependencies can be established, like start-to-start or even start-to-finish. The type of dependencies implemented in the PERT/CPM plan will affect the algorithms of the analytical links to be established with the SD model. For the sake of simplicity, in the SYDPIM basic mode only finish-to-start dependencies will be considered, which are the more important ones anyway. Where appropriate, other possibilities will be discussed;
 - (iii) resources – it is considered that resources are defined either as individuals or as types of resources (e.g. designers);
 - (iv) profiles of resource availability (per resource group) – most PERT/CPM tools also support the definition of profiles of resource availability to the project. These profiles specify the pool of resources available that can be allocated to the tasks. They are often used as input to optimisation algorithms. Generally, they do not apply to individual resources (e.g. amount of “Johns” available) but to some established grouping of these resources. Resource availability is *not* the same as resource *utilisation*, which is the overall allocation of resources across all tasks. Resource utilisation cannot exceed resource availability but it can be less. Resource availability in the project is also sometimes considered as the resource allocation to the project. It will be assumed that the PERT/CPM model will have two profiles of resource availability per established group: the planned and the actual profile. The grouping will correspond to a certain SD-Resource;
 - (v) allocation of resources to tasks – within the project, resources are allocated to each individual task. This allocation can be according to profiles that vary over time or a constant value;
 - (vi) a project initial date – typically, there is an initial project start date considered in the model from where all the elementary tasks are scheduled, possibly using relative dates (e.g. task x will start in month *n*);

- (2) *types of tasks* – SYDPIM considers that there are two main types of work carried out in the project: engineering work directly related with product development, and management work related with project control (e.g. monitoring progress, managing sub-contractors). In the SYDPIM basic mode each task is either an engineering task or a management task. SYDPIM also considers the possibility of engineering tasks being split into sub-categories, like “rework”, “QA”, “development”, “running tests”. While this is not mandatory in SYDPIM basic mode a discussion of this possibility will be provided where appropriate;
- (3) *task fields* – PERT/CPM tools vary in terms of the task fields allowed in the model. Some tools even provide blank fields to be defined and applied as decided by the user. Most of these fields are only informative and/or aimed at communication with other tools. In SYDPIM basic mode it will be considered that the following task fields will be available and used:
- (i) *schedule fields:*
 - (i.1) planned start date
 - (i.2) actual start date
 - (i.3) planned completion date
 - (i.4) actual completion date
 - (i.5) planned duration
 - (i.6) actual duration
 - (ii) *effort / cost fields*
 - (ii.1) planned initial budget
 - (ii.2) actual effort spent to date
 - (iii) *scope fields (*)*
 - (iii.1) planned initial scope
 - (iii.2) actual scope accomplished to date
 - (iv) *resources fields*
 - (iv.1) planned level / profile by resource
 - (iv.2) actual level / profile of resources employed to date

It is considered that the fields with planned decisions (e.g. planned start date) refer to the moment of that version of the PERT/CPM plan – this is not necessarily the same as the planned decision at the beginning of the project. The actual values will only be updated once and when appropriate. For example, the actual start date when the task starts and the actual completion

date when the task finishes. Once the actual values are updated the correspondent planned values are never changed again;

- (4) *availability of initial plan* – SYDPIM advocates that the initial plan should always be available. This initial plan provides the two main targets of cost and schedule established in the beginning of the project, and also the planned profile of resources;
- (5) *availability and recording of past plans* – SYDPIM also advocates that whenever a new version of the current plan is released, the previous version should be stored as a past plan and will be available throughout the project. In SYDPIM basic mode it will be considered that this is the preferred scenario but it is not mandatory. The availability of past version will affect the way in which the behaviour patterns of the PERT/CPM past behaviour are defined over-time, and the granularity of these patterns. Where necessary this issue will be clarified;
- (6) *types of plans and level of detail* – as mentioned previously, within the PERT/CPM model three integrated plans are maintained at different levels of detail (operational, tactical, and strategic), and each with different time-horizons. In SYDPIM basic mode an equivalent single PERT/CPM plan will be considered, covering the full project life-cycle and with a varying level of detail over-time (less detail into the far-future).

As it will be seen, these assumptions address issues which affect the specification of the analytical links between the PERT/CPM and the SD project model. They also affect the detailed processes within each of the activities of the SYDPIM process logic. The assumptions above are in accordance with the current practice of PERT/CPM project control within project organisations.

The task fields that refer to scope, marked with (*) above, raise a potential problem which often in PERT/CPM practice is not addressed properly: what is the definition of scope and how is it measured? Perhaps in most practical applications scope is not measure at all in PERT/CPM models. The reason why this measurement is included in the assumptions above is that scope is an essential input to a SD project model, upon which many other critical metrics are measured (e.g. productivity, defect generation index). There are two main reasons why scope is difficult to measure: it depends on the type of work being performed (and hence on the type of industry) and possible measurement units are often difficult to implement. While it is

not the purpose of SYDPIM to propose a final solution for this problem, it is important to clarify what will be the underlying assumptions for SYDPIM basic mode. To the author's experience there are two different perspectives that can be taken to define scope: (i) it is considered as work to be accomplished or (ii) it is considered as functionality to be delivered. The amount of work is highly related to the functionality delivered but it is not the same. For example, the technology employed to develop a certain functionality affects the amount of work required. A good example from the software industry is the "lines of code" as work and "function points" as functionality. According to the conventional project management body of knowledge, the WBS is used to specify the project scope. The WBS breaks down the project into elementary work tasks which altogether form the project scope. Therefore, the definition assumed is that scope is "work to be accomplished". This will be also assumed for SYDPIM. The second problem has to do with measurement. Again, to the author's experience there are two possible approaches: (a) an absolute measure of scope related to the type of work, or (b) a relative measure of scope based on the initial budget. Examples of the first approach are "no. of pages" to measure the scope of a design task, and "lines of code" to measure the scope of a coding task. The second approaches assumes that the scope for a task is measured in the scale 0% – 100%, possibly based on the initially estimated budget for the task measured as effort (person-month). The first approach is more tangible but makes it difficult to add scope across various tasks (e.g. adding "no. of pages" to "lines of code"). The assumption in SYDPIM is that it is the user's responsibility to measure scope as perceived easier and more appropriate for the specific project and industry. Since in terms of definition it is assumed that scope is "work", the only SYDPIM requirement for the measurement adopted is that, somehow, *scope can be added across various PERT/CPM tasks*. In particular, the analytical links will be based on this assumption and hence they will add scope across tasks as if it was measured in the same way in all tasks. If the assumption regarding the definition had been that scope was "functionality", the same analytical links could be established but their mathematics would have to be readjusted accordingly.

Given the object-hierarchy proposed for the PERT/CPM model and given these assumptions, it is now possible to provide a formal specification for the SYDPIM object "PERT/CPM model".

Object: PERT/CPM model

PERT/CPM model =

Initial date : date
 Current PERT/CPM plan =
 Past segment : plan-segment
 Future segment : plan-segment
 Initial PERT/CPM plan : plan-segment
 Past PERT/CPM plans : {plan-segment_k}
 PERT/CPM project behaviour =
 Past segment : {behaviour-pattern_k}
 Future segment : {behaviour-pattern_k}

Data-types:

plan-segment =

Tasks : {task_k}
 Dependencies : {dependency_k}
 Resources : {resource_k}
 Plan_Resource-availability : {(SD-Resource_k, {(t_i, y)})}
 Actual_Resource-availability : {(SD-Resource_k, {(t_i, y)})}

dependency = (task_i, task_j)

behaviour-pattern = {(t_i, y)}

Task =

planned start date : date
 actual start date : date
 planned completion date : date
 actual completion date : date
 planned duration : R+
 actual duration : R+
 planned budget : R+
 actual effort spent to date : R+
 planned scope : R+
 actual scope to date : R+
 planned resource profile : {(r_k, n)} or {(r_k, {(t_i, y)})}
 actual resource profile : {(r_k, n)} or {(r_k, {(t_i, y)})}
 task type : ENG | MAN

Formal specification of the SYDPIM object "PERT/CPM model"

The next SYDPIM object to be described is the SD model.

The SD model

Overview

Ideally, the SD project model to be used within the SYDPIM framework will be developed using the SYDPIM model development and validation method. Such a model will comply with all the requirements imposed by SYDPIM.

However, the organisation interested in applying this methodology may have developed its own model in another way, or may already have an existing model available. Whatever the situation, various types of SD project models can be developed, as shown in the review of chapter 2. SYDPIM is a flexible methodology and so it tries to accommodate the use of different types of models. It is therefore necessary to specify explicitly what are the characteristics required from the SD project model considered in SYDPIM basic mode.

Within the SYDPIM framework the SD model is used for two main generic purposes: (i) diagnosis of past performance and (ii) simulating the implementation of a PERT/CPM plan into the future, within various scenarios. In both cases the SD model will have to reproduce certain patterns of project behaviour which will be extracted from the PERT/CPM model. In order to accomplish this the model will have to satisfy the following requirements:

- (1) it must input or represent a PERT/CPM project plan – to do this the SD model must consider explicitly the following *inputs*:
 - (1.1) a project schedule with start and completion dates. This schedule may or not have an explicit break-down within the model;
 - (1.2) a project budget which may have or not a break-down within the model;
 - (1.3) a project scope which may have or not a break-down within the model;
 - (1.4) profiles of resource allocation and availability to the project, which may have or not break-down per resource category within the model;

- (2) it must input or represent some conditions within which the project plan is to be implemented – these conditions may be internal or external to the project and may be under more or less management influence. A typical example are risk factors, like product complexity (internal) and Client changes (external). There is no specific set of conditions that the model will have to consider;
- (3) it must produce output that allows the user to understand why the project is behaving in a certain way – this is practically required to any proper System Dynamics model. The outcome must result primarily from the internal feedback structure and so the feedback effects can be traced within the model. A typical example are the factors affecting productivity and defect generation. There is no specific formal condition to define this requirement, and so this will be considered as an assumption;
- (4) it must produce the patterns of project behaviour that will be extracted from the PERT/CPM model – as it will be seen, this is essential to establish some of the analytical links with the PERT/CPM model. There is a specific set of patterns that the SD model must reproduce in basic SYDPIM, which will be specified in the definition of the analytical links. This set of patterns must include the evolution over-time of the project objectives of cost and schedule, and it must also produce a pattern that measures the quality objective (e.g. defects detected);
- (5) it must allow the user to make changes to the project plan – i.e. re-planning decisions. Because the model will work as a laboratory to test alternative re-planning scenarios, it would be of no use without this requirement. The specific set of re-planning decisions that the model will support must cover all the re-planning decisions that can be implemented in the PERT/CPM model (though at a different level of aggregation). These are as follows:
 - (5.1) scope adjustments – increase, decrease and changes;
 - (5.2) schedule adjustments – extension and compression;
 - (5.3) budget adjustments – increase and decrease;
 - (5.4) resources adjustments – changes to the allocation profile;

These re-planning decisions will be imposed exogenously in the model throughout the project, as the model is used in every control cycle. The model must be able to record the occurrence of these exogenous decision over-time.

- (6) it must allow the user to change the conditions specified in (2). Again, this is essential for the model to perform a “what-if” analysis role, in particular for the diagnosis of past outcome and risk analysis. The specific set of changes allowed by the model will depend in the nature of the project conditions considered in (2). Again, these changes are also exogenous events that may occur over-time. Like in the previous assumption, the model will have to be able to record these.

A SD project model developed using the SYDPIM method will satisfy many other requirements which will improve the formal integration with the PERT/CPM model and thereby the performance of SYDPIM. An example is the breakdown of the project work into sub-tasks and the implementation of dynamic work dependencies within and among these tasks. These dynamic dependencies will allow a much better representation of the operational characteristics of a PERT/CPM plan. Another example is the breakdown of the effort expenditure among various types of work accomplishment activities. The set of requirements specified above provides the minimum required for any other SD project model to be used in SYDPIM.

Given these requirements, a structure can now be proposed for the object “SD model”. The SD model contains an internal architecture which specifies the breakdown of the project work into sub-tasks (if any), the dynamic dependencies between these tasks, and the resource (types or categories). A project work plan is specified upon this architecture. In SYDPIM basic mode this plan will consist of decisions regarding schedule, scope, budget and resource allocation, as mentioned in the assumptions above. Many other relevant decisions can be considered depending on the specific SD model being used, like, for example, work concurrency within and among tasks. The project work plan will contain a past and a future segment. The past segment of the work plan in the SD model differs considerably from the PERT/CPM model. This is because it must contain management re-planning decisions over-time and not just the initial decisions. For example, if the schedule was extended progressively, then the SD model must record the evolution of this target. In the PERT/CPM model various past versions of the PERT/CPM plan are recorded for this purpose. With the SD model, the single model will have to reproduce how the decision-making process of project control evolved. The critical issue is that while the SD model generates control decisions

endogenously within the model, these decisions might not be what management are willing to implement. The reason for this is that the decision-roles within the model do not capture all aspects that affect management decision-making in the real world. As shown in the monitoring process logic, SYDPIM requires that the SD model is recalibrated to reproduce the past, including management decisions. Since changing the endogenous decision-roles in the model might not be sufficient, the SD model will have to allow the user to impose exogenous management decisions over-time and record these, as stated in assumption (5) above. It is the combination of exogenous and endogenous control decisions that will ensure that the model will reproduce the patterns of past behaviour of the decision-making process. Therefore, the past segment of the SD plan will contain three elements: (i) the initial targets, (ii) the endogenous decision-roles, and (iii) the exogenous decisions. Likewise, the future segment of the SD plan will also contain these three elements. In the case of a *steady* plan, exogenous decisions are unlikely though possible – for example, to include mitigating actions that counter the likely occurrence of a risk. The decisions regarding the initial targets of the future segment are the project targets at the present moment. These targets must also result from the last decisions taken in the past segment. For example, if the current decision is to complete the project in day 100, then the last decisions in the past segment of the plan must have adjusted the schedule target accordingly.

The project conditions in which the project is implemented are also a relevant component of the SD model for SYDPIM purposes. Therefore, it needs to be made explicit in the abstract object structure here proposed. As mentioned before, the project conditions can be internal or external to the project. Like the project plan, they will have a past segment (the conditions that occurred) and a future segment (the conditions that are likely to occur). In the SD model, project conditions can be modelled through exogenous variables (constants or functions) or through more complex structures. Both are generally related with the moment in time when they occur (which can be an instant or a period of time). At the abstract level at which the SD model object is being defined, these conditions will be considered as input-parameters to the model.

Finally, the numerical results from the simulation are also a relevant sub-component of the SD model. These results can be considered as behaviour patterns, since

practically all the variables within the model are calculated for every time step. Some of these patterns will correspond to the patterns extracted from the PERT/CPM model and will be used to establish analytical links between the models. The set of patterns produced form the project behaviour, which will also have a past and a future segment.

Proposed object hierarchy

The object hierarchy for the SD model is shown in the figure 7.10 below:

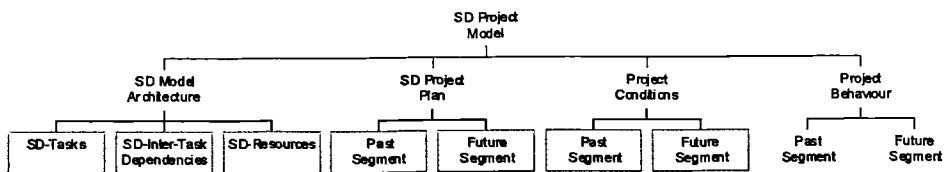


Figure 7.10 – SYDPIM object hierarchy of SD project model

It is important to note that this particular representation of the SD model is abstract and tailored to the formal specification of SYDPIM. In any case, it is possible to establish a correspondence to the structure of the SD model within the tool in which the model is developed.

Formal specification

The formal specification of the SD project model is shown in appendix C.2.

This specification is according to the object structure proposed for the SD model. It is considered that a SD plan consists of schedule, budget, scope and resources decisions. These decisions are split by SD-Task, reflecting the possibility of a project breakdown within the SD model. For example, each task will have their own schedules and schedule adjustments policies. It is also considered that the SD plan will include aggregate decisions at the project level, like the overall project completion date. In principle, these decisions should be consistent with the decisions at the SD-Task level. It could be assumed that the aggregate decisions are implicit in the SD-Task decisions. However, not only this might not be the case (e.g. an overall project contingency float may be considered), as it is important to

make the overall project decision explicit for integration purposes. As it will be seen in the definition of the analytical links, it is also considered that the resources decisions at the project level correspond to the profile of overall project resources availability in the PERT/CPM model.

The sub-object "Project behaviour" will contain all the behaviour patterns produced by the SD model. In principle, this can include all the variables in the SD model, which can be plotted against time. Therefore, it will depend on the specific variables available in the SD model. In the assumptions above, a subset of these variables was considered explicitly. These refer to the behaviour patterns which will be extracted from the PERT/CPM model and which any SD model considered will have to reproduce (see assumption (4) above).

Some of the data-types specified are defined as SD sub-structures, like the complex decision roles of staff adjustment. It is also considered that decision roles can incorporate various sub-decision-roles. The project conditions are specified as input parameters, which can be a constant, a function of time, an endogenous function of other model variables, or an entire SD sub-structure. An example of an endogenous function is the impact of work progress on productivity, which consists of a "learning curve" within the project – the shape and magnitude of this learning curve can vary according to the type of product, development process and organisation and hence it will be an internal project condition.

It is considered that the inter-task dependencies (SD-Dependency) will contain the identification of the predecessor and successor tasks and also a called "progress curve", which specifies the work progress that can be accomplished in the successor task given the progress in the predecessor task. The same type of curve is also considered explicitly as part of the SD-Task. This curve specifies the amount of work that can be started given the work progress already accomplished within the task. This issue regarding work dependencies will be clarified later.

Like the object hierarchy, this formal definition is abstract providing a particular but convenient view of the SD model. It considers explicitly the elements which are relevant for the formal specification of SYDPIM. These elements will be further used in the specification of the analytical links and detailed activities of the SYDPIM

process logic. Other possible characteristics of the model which are not relevant for this purpose remain implicit in this specification. For example, the exact decision role within the model that adjusts the project completion date can be more or less complex, but is not relevant for the implementation of SYDPIM. Therefore, it is not specified explicitly in the definition above.

The next object to be formally specified is the SYDPIM metrics database.

The SYDPIM metrics database (SYMDB)

Overview

One of the proposed components of the SYDPIM methodology is a specialised metrics plan (SYMP). This plan specifies a number of management metrics to be collected in every control cycle, throughout the project life-cycle. These management metrics have two major roles within SYDPIM:

- (1) they will support the implementation of the SYDPIM process framework:
 - providing data to derive the behaviour patterns of the project past, which the SD model will have to reproduce in monitoring. This is particularly useful if past versions of the PERT/CPM plan are not being recorded frequently in the PERT/CPM model;
 - providing data that will help calibrating and validating the SD model for both past and future segments of the project. This will improve the likely accuracy of the model forecasts;
- (2) they will provide extra information that will improve management decision-making:
 - measuring and tracking product and process quality. The quality metrics to be collected will allow the monitoring and control of quality as a project objective. This is not provided by the standard use of the PERT/CPM model alone;
 - providing unmeasured or intangible information about work progress and about the project status. This refers to metrics derived from the project diagnosis using the SD model. Typical examples are “undetected defects” and “level of staff fatigue”. These metrics improve management visibility of

progress, which is a key factor to improve the overall decision-making process of project control.

The implementation of the SYDPIM metrics plan is not mandatory but it will improve considerably the whole SYDPIM control process.

The output of implementing the SYMP is a metrics database which will record all the metrics collected over-time throughout the project life-cycle. As described in the previous sections, this database is an object within the SYDPIM process logic.

The SYMP groups the metrics to be collected in a set of generic categories. Within each category a sub-set of specific metrics can be considered. In practice, the specific metrics to be used depend on four main factors:

- (1) the project management information system (PMIS) used by the organisation must be capable of providing these metrics – i.e. their collection is feasible;
- (2) the effort and time required to collect these metrics must be acceptable and balanced by the perceived benefits;
- (3) the metrics should be tailored to the specific type of industry, product development process, and organisational processes. The metrics must be meaningful within the context of the specific project ;
- (4) the metrics must also be tailored to the specific SD project model in use. This is because some of the metrics are aimed at supporting the calibration of the model, while other metrics are derived from the model's results.

Factor (2) addresses a critical issue in the success of any metrics plan. While this is a subject of some controversy, it is often perceived by organisations that metrics collection programs often result in considerable investments of effort while the benefits are not perceived or are minor. One of the key factors for the success of a metrics program is to ensure that the contents of the metrics plan is driven by the project needs. The SYMP is tailored to support the implementation of the SYDPIM project control framework. Project control is a primary need of any project. Other factors, like the simplicity of the metrics (e.g. definition, measurement), are also considered as critical. As it will be seen, the metrics proposed in the SYMP are simple and meaningful for project management purposes.

Metrics plans are used, more or less commonly, in various industries. They are commonly used in the hardware industry, and over the last decade they have also been given particular attention in the software industry. A formal metrics plan can be specified in different ways and at different levels of detail. In general, a good metrics plan should specify a list of metrics to be collected and used in the project, and for each metrics it should specify, at least, the following elements:

- (1) *definition* – meaning and mathematical formulation;
- (2) *collection mechanism* – used to derive the numerical value the metric;
- (3) *interpretation* – how the value of the metric should be interpreted for a certain purpose;
- (4) *recommended actions* – given the interpretation of the current value of the metric, what control actions may be recommended.

The SYDPIM metrics plan (SYMP) will not be here specified at this level of detail. The metrics to be proposed are already used in various industries. Some are part of the conventional project management approach. Therefore, the four elements above are already available in the literature or from current practices. It is also important to note that these elements may also depend on the specific industry. A list of the metrics to be collected in the SYMP is here presented. It will be assumed that the four elements above will be known or available to the user of SYDPIM.

Different types of metrics can be used for different purposes. There are various criteria to classify metrics. For example, they are sometimes classified as “direct” or “indirect” depending on whether the result from the direct measurement of tangible characteristics of the product (e.g. size of software measured in “lines of code”). A classification scheme is very dependent on the specific industry. A possible generic scheme to classify metrics is to classify them depending on their focus. Three levels of increasing aggregation within a project are here proposed: (i) product, (e.g. size) (ii) development process (e.g. productivity), and (iii) project (e.g. schedule variance).

The metrics proposed in the SYMP focus essentially at the project and at the process level. The SYMP considers the following generic categories of metrics:

- (1) *schedule* – planned and actual start and completion dates
- (2) *effort* – planned budgets and actual effort expenditures
- (3) *resources* – planned and actual profiles

- (4) *scope* – planned, changes and actual scope
- (5) *quality* – based on defect counting
- (6) *project performance indices* – cost and schedule assessment
- (7) *process metrics* – parameters of the development process
- (8) *uncovered metrics* – derived from diagnosis in the SD model
 - (8.1) *quality* – process and product, based on defect counting
 - (8.2) *staff related* – based staff conditions modelled;
 - (8.3) *effects on process parameters* – based on feedback effects modelled.

Categories (1) through (4) refer to metrics which are already collected and used in the conventional use of the PERT/CPM model. Categories (5) and (7) refer to metrics which are not commonly used in the conventional PERT/CPM approach. The lack of quality data in a PERT/CPM plan prevents this model to analyse the trade-offs against the quality project objective. On the other hand, SD project models tend to consider quality measurement in some way. A project model developed using the SYDPIM method will measure this project objective, based on defect counting. Similarly, process metrics like “productivity” or “defect generation rate” are not considered in the conventional use of a PERT/CPM model, but are generally considered in the SD model since this is required to represent the work accomplishment process within the model. These metrics should be directly related with input parameters of the SD model and hence are very important for calibration purposes. Category (6) includes a conventional set of indices used to assess the project cost and schedule performance, like the *earned value* and the schedule performance index (SPI). These indices are generated from the effort metrics in category (2) (Nicholas 1990). Some PERT/CPM tools generate these metrics automatically from a project plan. A SD model can easily produce the calculation of these indices. Finally, category (8) refers to metrics which are generally not collected from the real world, but which are relevant to assess the project status and progress. They are not collected because of lack of procedures, or because they are intangible. These metrics are produced by the SD model and therefore the specific metrics to be considered within this category depends on the characteristics of the SD project model used. Three sub-categories are proposed in basic SYDPIM, based on project aspects that a good SD project model should address and quantify: undetected quality issues (e.g. undetected defects), staff issues of

subjective nature (e.g. staff fatigue) and feedback effects on process parameters (e.g. effect of defect density on cost to detect).

This set of metrics categories should be considered as a basis from which any organisation may develop their own metrics plan, tailored to their own projects. In the following section, a specific set of metrics is proposed for each category. This constitutes the SYMP proposed for SYDPIM basic mode. The author considers that the critical metrics are identified in this basic plan for SYDPIM purposes. However, the user of SYDPIM may wish to adjust this plan, for a better fit to the specific project and SD model in use. More metrics can be considered in each category, or other categories of metrics can be introduced in the plan.

Proposed data contents

In SYDPIM basic mode, within each metrics category a specific set of metrics is proposed. This basic metrics plan is consistent with all the assumptions described so far regarding the requirements of both PERT/CPM model and SD model in the SYDPIM basic mode. These metrics are to be collected periodically in every control cycle and are stored in the SYDPIM metrics database. These are presented below. The comments after each metrics are aimed at illustrating the collection (or calculation) procedure.

Metrics for SD-Task: *whole project*

- (1) schedule – can be derived from PERT/CPM model
 - (1.1) Start date – from current PERT/CPM plan
 - (1.2) Finish date (SAC) – from current PERT/CPM plan
- (2) effort – can be derived from PERT/CPM model
 - (2.1) ACWP – effort spent in tasks of current PERT/CPM plan
 - (2.1.1) engineering effort
 - (2.1.1.1) spent with defect detection – from QA tasks or estimated
 - (2.1.1.2) spent with defect rework – from rework tasks or estimated
 - (2.1.1.3) spent in development – from development tasks or estimated
 - (2.1.2) management effort
 - (2.1.2.1) human resource management – from HRM tasks or estimated
 - (2.1.2.2) project control – from management tasks or estimated
 - (2.2) BCWP – from initial and current PERT/CPM plan
 - (2.3) BCWS – from initial PERT/CPM plan
 - (2.4) CTC – from current PERT/CPM plan or calculated $CTC=CAC-ACWP$
 - (2.5) CAC – from current PERT/CPM plan or calculated $CAC=CTC+ACWP$

- (3) resources – can be derived from PERT/CPM model
 - (3.x) resource category x
 - (3.x.1) ASP – from current PERT/CPM plan
 - (3.x.2) PSP – from initial PERT/CPM plan
 - (3.x.3) CASP – from current PERT/CPM plan
 - (3.x.4) CPSP – from initial PERT/CPM plan
 - (3.x.5) CSPAC – from current PERT/CPM plan
- (4) scope – can be derived from PERT/CPM model
 - (4.1) SCAC – from current PERT/CPM model
 - (4.2) CSCC – from initial and current PERT/CPM plans
 - (4.3) ASCWP – from current PERT/CPM plan or calculated $ASCWP = SCAC - SCTC$
 - (4.4) SCTC – from current PERT/CPM plan or calculated $SCTC = SCAC - ASCWP$
- (5) quality – collected from project
 - (5.1) defects detected – collected
 - (5.2) defects reworked – collected
 - (5.3) cumulative defects detected – calculated
 - (5.4) cumulative defects reworked – calculated
 - (5.5) defects awaiting rework – calculated
- (6) project performance indices – calculated (conventional project management)
 - (6.1) EV (earned value) – initial budget – CTC
 - (6.2) CPI (cost performance index) – $BCWP/ACWP$
 - (6.2) SPI (schedule performance index) – $BCWP/BCWS$
 - (6.3) CV (cost variance) – calculated – $ACWP - BCWP$
 - (6.4) SV (schedule variance) – $BCWS - BCWP$
 - (6.5) AV (accounting variance) – $ACWP - BCWS$
 - (6.6) TV (time variance) – present – $t : BCWS[t] = BCWP[present]$
- (7) process metrics
 - (7.1) productivity – calculated
 - (7.1.1) Gross productivity
 - (7.1.2) Net productivity
 - (7.2) defect metrics – calculated
 - (7.2.1) defect detection index
 - (7.2.2) cost to detect
 - (7.2.3) cost to rework
- (8) uncovered metrics – generated by the SD model
 - (8.1) defects related
 - (8.1.1) undetected defects
 - (8.1.2) cumulative defects generated
 - (8.1.3) defect generation rate
 - (8.1.4) current density of undetected defects
 - (8.1.5) actual cost to detect next defect
 - (8.5) staff related
 - (8.5.1) staff fatigue
 - (8.5.2) staff experience
 - (8.6) effects on process parameters
 - (8.6.1) on productivity
 - (8.6.2) on defect generation
 - (8.6.3) on defect detection
 - (8.6.4) on defect rework

Abbreviations:

SAC – schedule at completion
ACWP – actual cost of work performed
BCWP – budgeted cost of work performed
BCWS – budgeted cost of work scheduled

CTC – cost to complete
CAC – cost at completion
ASP – actual staff profile
PSP – planned staff profile
CASP – cumulative actual staff profile
CPSP – cumulative planned staff profile
CSPAC – cumulative staff profile at completion
SCAC – scope at completion
CSCC – cumulative scope changes
ASCWP – actual scope of work performed
SCTC – scope to complete

Metrics collected in SYDPIM basic mode

The SYDPIM metrics database will store patterns of these metrics over-time, from the beginning of the project up to the present moment. The metrics are collected at the SD-Task level, according to the project work breakdown assumed in the SD model. Depending on the specific type of tasks considered, some of the metrics may not be applicable. For example, if the SD model was developed using the SYDPIM method, the metric “defects detected” is not applicable to a management type of task. Global metrics for the whole project are also generated from the metrics at the SD-Task level. As shown, resource metrics are further split according to the resource types considered in the SD model.

Each metric is collected (or calculated) at a specific stage in the SYDPIM process logic, according to a specific procedure which will be presented. Some of the metrics will be derived from the PERT/CPM model, other metrics will have to be collected from the project, others can be derived analytically from previous ones, and others will be generated by the SD model – see comments. A more detailed description of the collection (or calculation) procedure for each metric will be presented when the activities of the SYDPIM process logic are described in detail.

It is important to note that, in an ideal scenario, many of these metrics will be also stored within the SD model, in particular metrics in category (7). So why also store them in the metrics database? There are at least three good reasons for this: first for historical reasons; secondly, the SD model may not retain this data all the time – a specific simulation-run must be recorded, and the purpose of the model is not to work as a database; and finally, the some metrics considered in the SYMP may not reproduced by the SD model. A database is safer, more appropriate for analysis

purposes. It will also retain historical metrics in an organised manner, over the course of various projects – this is useful to apply SYDPIM in future projects.

Formal specification

The formal definition of the SYDPIM object “SYMDB” is presented in appendix C.3.

All metrics stored in the database are decomposed by SD-Task. This is important because in this way the patterns of the project past behaviour will be available within each task. The database also stores the aggregate behaviour patterns for the project. Like with the definition of a SD-plan in the SD model object, it could be argued that these aggregate project patterns could be derived from the patterns at the task level. However, this might not be the case for all patterns at the project-level and it is important to have the project patterns explicitly defined. Of course, a particular implementation of the database could impose consistency and thereby derive some of the patterns at the project level from the patterns at the task level.

As already mentioned, some of the metrics may only be appropriate for some types of tasks within the SD model. Again, this is an implementation issue which will depend on the characteristics of the SD model being used. For example, in some models defect generation can be considered within a management task, in other models this might not be the case. The metrics about “effects on process” will show the impact over-time of certain factors on process parameters. For example, the effect of error density (factor) on defect generation (process parameter). In the definition above, an effect is considered a SD model variable which in most cases will be endogenous.

Other types of metrics or categories of metrics can be considered in the SYMDB, to support the implementation of the SYDPIM framework. This should depend primarily on the project issues and management needs, and will be restricted by the structure of the SD model and its variables. This is particularly true for quality and process metrics. For example, Burdick et al (1998) propose the quality metrics “no. of reviews” and “time between reviews”. This is primarily aimed at supporting the calibration of “rework-cycle” type of models, where these metrics are required. On the other hand, these models do not consider explicitly the entity “defects” (thus not

counting them). From experience with working with this and other types of models, it is the author's opinion that the metrics proposed above should always be collected regardless of the type of SD model used. Other additional metrics can be collected if necessary and *useful*.

Other objects

The three main objects used to describe the SYDPIM process logic have just been specified formally: the PERT/CPM model, the SD model, and the SYMDB. There are three other objects which were used explicitly in the description of the SYDPIM process logic: "PERT/CPM past behaviour", "Project past behaviour" and "PERT/CPM future behaviour". These objects have to do with the explicit representation of the project behaviour as a set of patterns over-time.

These objects are now specified by order of appearance in the process logic. The object "PERT/CPM past behaviour" is used in monitoring as an output from activity (M4) and is an input to activity (M5). The object "Project past behaviour" is further used in monitoring as an output of activity (M5) and is an input to activity (M6). Finally, the object "PERT/CPM future behaviour" is used in planning as an output of activity (P3a) and is an input to activity (P4a).

The objects "PERT/CPM past behaviour" and "PERT/CPM future behaviour" are sub-components of the object "PERT/CPM model".

The PERT/CPM past behaviour

Overview

This object is generated and/or updated in activity (M4). It is the sub-component "PERT/CPM project behaviour.Past Segment" of the PERT/CPM model. It stores the behaviour patterns the project past which are extracted *directly* from the PERT/CPM model, just after this model has been updated with results. The main aim of this object is to provide the basis for a set of patterns which will have to be reproduced by the SD model.

The specific patterns to be extracted from the PERT/CPM model depend on three main factors:

- (1) data availability from the PERT/CPM model;
- (2) past versions of PERT/CPM plans available;
- (3) analytical links with the SD model.

Since the patterns extracted are to be reproduced by the SD model, the analytical links will establish the level of work and resource breakdown considered. For example, if the SD model considers only two SD-Tasks, then the project effort expenditure will be split into those two patterns. A similar situation applies to resource categories. The analytical links will specify which specific patterns have to be reproduced by the SD model and the formal level of "goodness of fit" required.

Of course, for each pattern to be extracted, the required data must be available in the PERT/CPM model. In this respect, the basic mode of SYDPIM assumes that a number of task fields will be available and are updated in the PERT/CPM plan. These fields were identified in the specification of the object "PERT/CPM model" as follows: schedule, effort, scope and fields.

The availability of past versions of PERT/CPM plans restrains both the possibility of some patterns to be extracted and their granularity. The unavailability of the PERT/CPM plan developed at the beginning of the project (here referred to as "initial plan") prevents a number of patterns that depend on initial targets to be extracted. Again, in the specification of the PERT/CPM project model it was assumed that this initial plan would preferably be available and if not that the initial targets of cost and schedule would be available where required.

The availability of other past versions of the PERT/CPM plan within the model restrains the level of granularity for those behaviour patterns that make use of these versions. It may also affect the way in which the patterns are extracted. A typical case, are the patterns related with control decision-making, like the project targets. For example, the evolution of the scheduled completion date over-time is extracted from the series of critical-paths of the past versions of the PERT/CPM plan. For this type of patterns, each past version of the PERT/CPM is a data-point. Thus, the higher the number of past versions available the higher the granularity of the pattern

extracted from the PERT/CPM model. In the specification of the object "PERT/CPM model", it was assumed that past versions would preferably be available but this is not mandatory to implement SYDPIM in basic mode.

There are two more factors that may affect the patterns that can be extracted from the PERT/CPM model: (1) explicit consideration of different types of tasks in a PERT/CPM plan, and (2) level of detail of the plans within the PERT/CPM model. If different types of tasks are considered, then a split of certain behaviour patterns is possible. For example, if all QA work and defect rework are allocated only to activities which are exclusively dedicated to that type of work, then the effort expenditure pattern can be split into a QA and a rework pattern. In SYDPIM basic mode it was assumed that only two types of tasks are mandatory: engineering and management. Therefore, some behaviour patterns will be split among these two types of work. In the specification of the SYMDB the engineering work was further split into three sub-types: development, QA and rework. If specialised tasks are considered in the PERT/CPM plan then the update of the SYMDB can be made from the PERT/CPM model – this is not mandatory. The level of detail will affect the granularity of the patterns. If certain past phases of the project were planned and updated only at the tactical level, fewer data points can be extracted from the PERT/CPM model for each pattern.

An implementation issue previously raised is whether in activity (M4) the patterns within this object are simply updated every control cycle or whether they are generated from the beginning of the project. This will be discussed in more detail in the specification of activity (M4) and is not relevant for the specification of the object contents.

Proposed data contents

Based on all previously stated the assumptions about the PERT/CPM model, the following patterns of behaviour will be extracted directly from the model and will be stored in the object "PERT/CPM past behaviour":

Patterns of SD-Task: whole project

- (1) schedule
 - (1.3) Start date
 - (1.4) Finish date (SAC)
- (2) effort
 - (2.1) ACWP – actual cost of work performed
 - (2.1.1) engineering effort
 - (2.1.1.1) spent with defect detection – from QA tasks or estimated*
 - (2.1.1.2) spent with defect rework – from rework tasks or estimated*
 - (2.1.1.3) spent in development – from development tasks or estimated*
 - (2.1.2) management effort
 - (2.1.2.1) human resource management – from HRM tasks or estimated*
 - (2.1.2.2) project control – from management tasks or estimated*
 - (2.2) BCWP – budgeted cost of work performed
 - (2.3) BCWS – budgeted cost of work scheduled
 - (2.4) CTC – cost to complete
 - (2.5) CAC – cost at completion
- (3) resources
 - (3.x) resource category x
 - (3.x.1) ASP – actual staff profile
 - (3.x.2) PSP – planned staff profile
 - (3.x.3) CASP – cumulative actual staff profile
 - (3.x.4) CPSP – cumulative planned staff profile
 - (3.x.5) CSPAC – cumulative staff profile at completion
- (4) scope
 - (4.1) SCAC – scope at completion
 - (4.2) CSCC – cumulative scope changes
 - (4.3) ASCWP – actual scope of work performed
 - (4.4) SCTC – scope to complete

Patterns stored in object “PERT/CPM past behaviour”

These patterns are extracted from the PERT/CPM model for the whole project and for each SD-Task. The breakdown per SD-Task is based on the analytical links established between the SD and the PERT/CPM model. A split per resource category is also considered. The breakdown of the engineering and management effort is in italic to stress that this is optional, depending on the types of tasks considered explicitly in the PERT/CPM model.

The patterns above are subset of the metrics considered in the SYMDB. So, why repeat this? There are various reasons. The SYDPIM process logic of monitoring considers that neither the PERT/CPM plan is always updated nor the SYMDB is

always in use. The possible scenarios are as follows: (1) if the PERT/CPM model has not been updated in this control cycle, then the object "PERT/CPM past behaviour" will not be updated as well. In this case the SYMDB will be used as the source of data to derive the project past behaviour to which the SD model will be calibrated; (2) if the PERT/CPM model has not been updated and the SYMDB is not in use, the object "PERT/CPM past behaviour" will not be used and expert judgement will be used to derive the project past behaviour; (3) if PERT/CPM model was updated and the SYMDB is not in use, then the "PERT/CPM past behaviour" object will be updated and will be used as the basis to calibrate the SD model; finally, (4) if the PERT/CPM model was updated but the SYMDB is in use then the "PERT/CPM past behaviour" will not be generated because its behaviour patterns are already stored in SYMDB. In summary, the object "PERT/CPM past behaviour" will only be used if the PERT/CPM model was updated and the SYMDB is not in use.

Formal specification

The formal specification of the object "PERT/CPM past behaviour" is presented in appendix C.4.

The patterns stored in this object are used to produce the final set of patterns to be stored in the object "project past behaviour".

The project past behaviour

Overview

This object is generated and/or updated in monitoring activity (M5). It is an independent object and so it is not a sub-component of any other object. It stores the behaviour patterns used in SYDPIM to describe the *full* project past behaviour. In principle, this is a more comprehensive set of patterns than the "PERT/CPM past behaviour". It will contain not only those patterns extracted from the PERT/CPM model but also other patterns derived from the metrics database (SYMDB) and even from expert judgement. This final set of patterns will have to be reproduced by the SD model, according to the analytical links established.

The aim of this object is to provide a most comprehensive and accurate description of the project past behaviour. This is because the SD model will be calibrated to reproduce this behaviour and hence the accuracy of subsequent SD analysis will depend on this description.

The specific set of patterns to be included in this object depend on the following factors:

- (1) the set of pattern stored in the object "PERT/CPM past behaviour";
- (2) the set of metrics recorded in the SYDPIM database;
- (3) the patterns derived from expert judgement;
- (4) the analytical links with the SD model.

As discussed previously, the analytical links will establish which patterns are to be reproduced by the SD model and the level of "goodness of fit". As for the other factors, the ideal is that those patterns that refer to quantitative aspects are taken from the "PERT/CPM past behaviour" or derived from the SYDPIM metrics database. The patterns to be derived from expert judgement should refer to non-quantitative issues.

In the basic mode of SYDPIM it will be assumed that this object will store the patterns specified in the objects "PERT/CPM past behaviour" and SYMDB, plus some expert judgement patterns.

The patterns derived from expert judgement are important and will play a distinctive role. They may be considered in three situations:

- to address important temporary issues that occurred in the project – this may happen only occasionally. A typical case are the risks. For example, the risk "Client changes": if the Client asked for scope changes during a certain period of time, then it is important that the SD model will reproduce accurately the shape of these requests and of some related impacts over-time. While the data from the PERT/CPM model and the metrics in the SYMDB may help to derive these patterns, in most occasions expert judgement will be necessary;

- to address intangible issues which management are not able to measure on a regular basis (or even not willing to measure at all), like “staff motivation”. For these patterns and ordinal scale 0-1 or a qualitative scale may be preferred;
- finally, if the SYMDB is not being maintained, and/or the PERT/CPM model is being not updated as assumed in basic SYDPIM, then expert judgement will have to be used to derive the required patterns.

So, the set of patterns derived from expert judgement can be dynamic and change over the course of the project (i.e. some patterns are included and other are excluded). The key requirement to consider this type of patterns is that the SD model will have to reproduce them.

Like with the object “PERT/CPM past behaviour”, the patterns stored in this object may be either updated in every control cycle or generated from the beginning of the project. Once this object contain patterns derived from expert judgement, the first option is recommended. Again, this is an implementation issue to be considered in activity (M5) and is not relevant for the specification of the object contents.

Proposed data contents

The patterns proposed in basic SYDPIM for this object are as follows:

Patterns of SD-Task: *whole project*

IF SYMDB is being maintained THEN

- (1) all patterns in SYMDB except category (8)
- (2) expert judgement patterns
 - (2.1) risk-related patterns
 - (2.2) intangible-issues patterns

ELSE

IF PERT/CPM model is updated THEN

- (1) all patterns in “PERT/CPM past behaviour”
- (2) expert judgement patterns
 - (2.1) SYMDB patterns
 - (2.1.1) quality patterns (category (5) in SYMDB)
 - (2.1.2) performance patterns (category (6) in SYMDB)
 - (2.1.3) process metrics (category (7) in SYMDB)
 - (2.2) other patterns
 - (2.2.1) risk-related patterns
 - (2.2.2) intangible-issues patterns

ELSE

- (1) expert judgement patterns
 - (1.1) SYMDB patterns except category (8)
 - (1.2) other patterns
 - (1.2.1) risk-related patterns
 - (1.2.2) intangible-issues patterns

Patterns stored in object "Project past behaviour"

The IF-THEN-ELSE clauses address the possible scenarios regarding the implementation of the SYMDB or the update of the PERT/CPM model. Partial updated of the PERT/CPM model are not considered explicitly but it is implicit that expert judgement would have to cover the lack of data. The "PERT/CPM past behaviour" will contain the patterns of data categories (1) to (4) of SYMDB. The category (8) of SYMDB refers to uncovered metrics using the SD model after calibration for past behaviour. These patterns are not included in the object "Project past behaviour". Two main categories of expert judgement patterns are proposed: risk related and intangible issues. The specific patterns are to be defined by the user and must be reproduced by the SD model.

Formal specification

The formal specification of the "Project past behaviour" is presented in appendix C.5.

The data-type generic-pattern was introduced to allow the expert judgement patterns to be specified by the user to be project wide, task specific, resource specific, or task and resource specific. The other patterns derived from the metrics database and from the PERT/CPM model were specified previously to consider these possibilities.

The PERT/CPM future behaviour

Overview

This object is generated and/or updated in planning activity (P3a). It is the sub-component "PERT/CPM project behaviour. Future segment" of the PERT/CPM model. It stores the patterns of the project future behaviour which are extracted

directly from the PERT/CPM model, just after a new plan has been developed or the current plan has been readjusted for the future. This set of patterns describes the project future behaviour implicitly forecasted by this plan.

The main aim of this object is to help re-calibrating the SD model for the PERT/CPM plan. This is done mainly by making the SD model reproduce the project future behaviour implicit in the PERT/CPM plan.

Whenever the project is re-planned in the PERT/CPM model and thus the future segment of the PERT/CPM plan is changed, a new project future behaviour is being anticipated. Therefore, unlike with the object “PERT/CPM past behaviour”, the patterns in this object will need to be re-generated all the time.

The specific patterns to be considered in this object depend on the following factors:

- (1) data availability from the PERT/CPM model;
- (2) past versions of PERT/CPM plans available;
- (3) analytical links with the SD model.

The situation is similar to the object “PERT/CPM past behaviour”. The analytical links establish which patterns must be reproduced by the SD model and their breakdown across tasks and resources. In basic SYDPIM it is assumed that the task-fields specified in the description of the object “PERT/CPM model (i.e. schedule, effort, scope and resources), will be used and updated, in this case only referring to “planned” type of fields.

The availability of past versions of PERT/CPM plans is only relevant for the initial PERT/CPM plan, which is required to produce some of the patterns. Again, it is assumed that if this plan is not available the initial targets of cost and schedule will be made available. No other past version of the PERT/CPM plan is required and their absence has no impact on the granularity of the patterns to be extracted.

The explicit consideration of different types of tasks in the PERT/CPM plan (e.g. rework and QA tasks) will affect the possible breakdown of some patterns of future behaviour, just as discussed previously with the patterns of the “PERT/CPM past

behaviour". The level of planning detail of the future segment of the current PERT/CPM plan will affect the granularity of the patterns produced. Often, future phases of a project are planned only at the tactical or even strategic level. In this case, fewer data points can be extracted from the PERT/CPM model for each pattern.

Proposed data contents

Based on this, the following patterns of future behaviour will be extracted directly from the PERT/CPM model and will be stored in the object "PERT/CPM future behaviour":

Patterns of SD-Task: *whole project*

- (1) schedule – *steady*
 - (1.2) Start date
 - (1.3) Finish date (SAC)
- (2) effort
 - (2.1) ACWP – actual cost of work performed
 - (2.1.1) engineering effort
 - (2.1.1.1) *spent with defect detection – from QA tasks or estimated*
 - (2.1.1.2) *spent with defect rework – from rework tasks or estimated*
 - (2.1.1.3) *spent in development – from development tasks or estimated*
 - (2.1.2) management effort
 - (2.1.2.1) *human resource management – from HRM tasks or estimated*
 - (2.1.2.2) *project control – from management tasks or estimated*
 - (2.2) BCWP – budgeted cost of work performed
 - (2.3) BCWS – budgeted cost of work scheduled
 - (2.4) CTC – cost to complete
 - (2.5) CAC – cost at completion – *steady*
- (3) resources
 - (3.x) resource category x
 - (3.x.1) ASP – actual staff profile
 - (3.x.2) PSP – planned staff profile
 - (3.x.3) CASP – cumulative actual staff profile
 - (3.x.4) CPSP – cumulative planned staff profile
 - (3.x.5) CSPAC – cumulative staff profile at completion – *steady*
- (4) scope
 - (4.1) SCAC – scope at completion – *steady*
 - (4.2) CSCC – cumulative scope changes
 - (4.3) ASCWP – actual scope of work performed
 - (4.4) SCTC – scope to complete

Patterns stored in object "PERT/CPM future behaviour"

The patterns are the same as for the PERT/CPM past behaviour. The planned fields refer to the initial plan. For example, the pattern BCWS refers exclusively to what is specified in the initial plan. As already mentioned, some patterns will require the availability of the initial plan. Some patterns *must be steady*, unlike with the past behaviour. These are identified above and refer to planning targets: cost, schedule, scope and resources. This is because a PERT/CPM plan for the future assumes that the desired targets will be achieved, and the outcome is expected as planned. Steady patterns consist of targets being constant over-time.

Formal specification

The formal specification of the object “PERT/CPM future behaviour” is presented in appendix C.6.

Again, a breakdown per SD-Task is considered and will depend on the analytical links with the SD model. Likewise, a split per resource category is also considered. These patterns are to be reproduced by the SD model when calibrated for the PERT/CPM future plan.

Summary

The six main objects used in the SYDPIM process logic were specified. Objects are considered as individual abstract entities which store and process information within this process. The three more important objects are the PERT/CPM model, the SD model, which are integrated, and the SYMDB which supports this integration. The models work primarily as data processors and have an object hierarchy. The SYMDB works as the main data repository in SYDPIM. Two sub-objects of the PERT/CPM model are also used explicitly in the process logic. They store the data that describes the project past and future behaviours, as implicitly portrayed by the PERT/CPM model. These sub-objects are the “PERT/CPM past behaviour” and the “PERT/CPM future behaviour”, which work as data repositories. They can be generated only temporarily while they are needed in specific steps within the process logic. Finally, the object “Project past behaviour” is an independent object which stores the actual behaviour exhibited by the project up to present. It works as

a data repository which preferably exists throughout the whole project life-cycle and is continuously updated.

The definition of these objects included the formal specification of their object hierarchy and data contents. This specification will be used in the definition of the analytical links and of the activities of the process logic.

7.4.3 Analytical links

Basic principles of integration

Before defining the analytical links proposed in SYDPIM to formally integrate the two models, it is essential to understand the basic principles behind this concept.

When using SYDPIM, the project manager will quickly be confronted with the following questions: (1) how is data exchanged between the two models, and (2) what should be the relationship between the structure of both models? The scheduled completion date of design has just been changed in the SD model and the project performance looks fine. Of which tasks in the PERT/CPM plan should the schedules be changed to get the same plan? Which are the "design" tasks in the PERT/CPM? Which tasks need to be removed? Do dependencies need to be changed? How? Answering these questions is critical to the formal and quantitative integration of the two models. The answers lay in the *analytical links* that need to be established between the two models. But what are these analytical links?

Analytical links are the formal bridge through which two models can be integrated. Both PERT/CPM and System Dynamics models consist of two main elements: structure and data. The analytical links to be established between the two models must address these elements and thereby establish structural and data relationships. Furthermore, throughout their application, the structure and the data of the models generally change over-time. Therefore, these relationships may need to be dynamic and thus readjusted whenever required. How should these dynamic analytical links be established and maintained?

Each model consists of a particular abstract way of representing a project, according to human judgement. Therefore, analytical links establish relationships between the two different forms of representing a common reality. It would then appear logical that the establishment of analytical links also requires human judgement: what in this model is element “x” corresponds to element “y” in the other model – how to decide? If the two types of models benefit from formal representational rules then it could be expected that analytical links could be formally deduced and hence no human judgement would be required. However, in this case both models represent a complex social system. In many cases their level of formality does not allow for such logical automation. As it will be seen, human judgement is required to establish and maintain the analytical links

The specific set of analytical links that should be implemented in any particular implementation of SYDPIM will depend on various factors: data availability in the PERT/CPM model, way in which the PERT/CPM model is used, structure and variables of the SD model and other issues related with the two particular models used. In order to overcome this problem of “relativity” to the models, a basic mode will be assumed for SYDPIM. In this mode a well defined set of links can be proposed, within which a core sub-set of links *must* be established so that formal integration is achieved.

The scenario of SYDPIM basic mode here described should be used as a reference to implement links in any other real world scenario that the project manager may have to face.

As mentioned above, there are two major aspects of model integration in SYDPIM: (1) structure, and (2) data. This leads to three types of formal links that can be established:

- (i) data-links,
- (ii) structural- links,
- (iii) data-structural links.

Let us start with the data issues.

Principles of data links

Both models consider *input* and *output* data. Input data is generally *entered* by the user whereas output data is *produced* by the model. Each of these two sets of data can be divided into two main sub-sets: data about the project *past* and data about the project *future*. In turn, each of these sub-sets can be further divided into other two sub-sets: planned *targets*, and actual *results*. As a result, the data used in any of the models can be classified into eight possible categories, as shown in the table 7.1 below.

	Past	Future
Input	(1) – planned (e.g. scheduled completion of Design) (2) – actual (e.g. actual completion of Design)	(3) – planned (e.g. scheduled completion of Coding) (4) – actual (*) (e.g. actual completion of Coding)
Output	(5) – planned (e.g. project staff profile) (6) – actual (e.g. actual project staff profile)	(7) – planned (e.g. planned project cumulative cost) (8) – actual (**) (e.g. forecasted project cumulative cost)

Table 7.1 – The eight types of data handled by a project model

Category (4) is marked with (*). This is because, in normal conditions, actual results are not *entered* in the model unless when used to support certain types of “what-if” analysis. In a PERT/CPM model a project plan ready to be implemented will not incorporate this type of data. Instead, the expectations about the future outcome are specified in category (3). In a SD project model this type of data is not very common as well and tends to be used only to reflect external influences in the project, like risks, which are not part of the project plan (e.g. Client changes).

Category (8) is also marked with (**). This is because unless a model carries out simulation, then the actual project outcome in the future is not produced as an output. Again, in a basic PERT/CPM model the output produced by the model in the future is part of the planned expectations (i.e. it is a target). More advanced network models (e.g. Golenko-Ginzburg 1988, Elmaghraby 1997) carry out simulation, in which case this type of data may be used. In a SD model, this type of data is used because the model simulates the project being implemented and an outcome being unfolded over-time.

In each of these eight categories, various types of data can be used. For the PERT/CPM model, various tools consider a wide range of possible data. It is assumed that in SYDPIM basic mode four types of data are used:

- (1) *schedules* – measured in units of time (e.g. day, month);
- (2) *effort (budget/cost)* – measured in person-time (e.g. person-hour). Sometimes also translated into units of currency (e.g. \$);
- (3) *resources* – measured in units of resources, typically “person” for human resources;
- (4) *scope* – measured in units of work (e.g. tasks, lines of code).

A SD project model should consider at least these four types of data, but most likely will consider many other types of data (e.g. measure of staff experience level, measures of productivity, etc.)

There are two types of *data-links* that can be established in SYDPIM between the two models:

- (1) *data exchange* – this implies that some data can be exchanged between the two types of models. This can be considered as the “most formal” type of link. These links can be established between the following data categories:
 - (i) *input-input* – the input of one model can be used as an input to the other model. This may involve inter-relationships among data categories (1) to (4) of both models;
 - (ii) *output-input* – the output of one model can be used as an input to the other model. This may involve inter-relationships between the data categories (5) to (8) and data categories (1) to (4) of both models;
- (2) *data consistency* – this implies that some data is not exchanged but must be consistent with other data in the other model. These links can be established between the data categories:
 - (i) *input-input* – the input of one model must be consistent with the input of the other model. Again, data categories (1) to (4);
 - (ii) *output-input* – the output of one model must be consistent with the input of the other model (and vice-versa). Again, data categories (5) to (8) and (1) to (4);

- (iii) *output-output* – the output of one model must be consistent with the input of the other model. This may involve inter-relationships between data categories (5) to (8) of both models;

Overall, five types of data-links can be considered. In order to be established, most data links require the definition of structural relationships between the two models. This is, they are built upon structural links.

Principles of structural links

The other aspect of model integration refers to the structure of both models. The two models are aimed at providing a logical representation of the same project, and so their structure must be related. SYDPIM considers two types of structural links: (i) structural correspondence and (ii) structural consistency. Structural correspondence links establish which elements of one model correspond to the elements of the other model. Two elements should correspond one another when they represent the same reality. Structural consistency links establish conditions that must be respect so that structural correspondence links are valid. These two types of links have sub-types as follows:

(1) *structural correspondence*:

- (i) *work breakdown* – any work package in one model must be mapped to another work-package in the other model. This suggests that the work break down in the two models must be made according to a common WBS (Rodrigues and Williams 1997);
- (ii) *work dependencies* – any work dependency between two work-packages must be mapped into some form of work dependency in the other model;
- (iii) *organisation breakdown* – any resource, or resource type, must be mapped to another resource (or resource type) in the other model. This suggests that the organisation break down in the two models must be made according to a common OBS (Rodrigues and Williams 1997);

(3) *structural consistency*:

- (i) *resource-task allocation* – the correspondence established between tasks and resources in both models must keep the resource

allocation consistent. If a resource is allocated to a task in one model, then the correspondent resource in the other model must also be allocated to the corresponding work task;

- (ii) *work dependencies* – like with the resources, the correspondence established between the work dependencies in both models must keep the work precedence consistent.

Overall, structural links require a direct and unambiguous mapping between (i) work packages, (ii) work dependencies, and (iii) resources between the two models. As already mentioned, *structural links are also the basis for the implementation of data links*. For example, if a design phase in the SD model aggregates various design tasks in the PERT/CPM model, then this mapping will be the basis to check the consistency of the profiles of designers considered in both models.

In some situations it is also possible to establish a relationship between the structure of one model with the data in the other model. This leads to the establishment of data-structural links.

Principles of data-structural links

There is a third type of links considered in SYDPIM: data-structural links. These links establish that the input or output data from one model must be consistent with and/or determine the structure of the other model. This is the more difficult link to be established between the two models. In great part, it results from the fact that the two types of models assume different levels of aggregation and/or breakdown of certain aspects of the project. For example, the PERT/CPM model is typically more detailed in the breakdown of the project work. Consider that the SD project model splits the whole effort spent in a work package among various continuous activities over-time: development, review, rework. Consider also that the PERT/CPM model contains work packages (or tasks) that correspond exclusively to these types of activities – i.e. development tasks, reviewing tasks, rework tasks. Then the profiles of “effort expenditure” over-time of these three activities in the SD model will have to be consistent with the number of these types of tasks, their schedules and dependencies in the PERT/CPM model.

Unlike the other links, data-structural links are useful but not mandatory in SYDPIM. They clearly move the integration of the two models into a more detailed and formal level, close to automation of model readjustment. Its practical implementation depends on data availability, formality of the traditional project control system in use, and on the SD project model used. The establishment of data and structural links is more important but, as mentioned before, they are also restricted by these factors. It is the user's responsibility to determine which type of link can be established and whether the benefit/cost is worth. The benefits stem from a more rigorous integration of the two models and hence a more thorough control of the project. The costs relate to data collection and implementation of the data/structural consistency and data exchange operations.

The analytical links are used within the SYDPIM process framework as described in the process logic (see figures 7.6 and 7.7). Some of the activities will use these links to perform their task. For example, activity (P4a) in planning uses data-consistency links in order to ensure that the SD model will reproduce accurately the PERT/CPM future behaviour. In doing so, the SYDPIM activities of the process logic will perform a set of more elementary activities of model integration, which we may call operations, like "data exchange".

SYDPIM elementary operations of model integration

The use of data and structural links are the basis of four distinctive SYDPIM operations used within the activities of the SYDPIM process logic. These are as follows:

- (A1) *data exchange* – numerical data is directly transferred from one model to the other. This can be formally specified by defining, analytically or through an algorithm, the value of the variables in the receiving model as a function of the variables in the source model. For example, the planned schedule of a SD-Task equals the critical path of the PERT/CPM sub-network mapped to that SD-Task. This activity requires the structural links as an input and uses the data exchange links as the "channels" to transfer the data.
- (A2) *data consistency checking* – the quantitative data-input or data-output of one model must respect certain conditions that depend on the input/output data of the other model. This can be formally specified and verified by defining,

analytically or through an algorithm, the relationships that must exist between the variables of the two models – i.e. a formal condition. For example, the staff profile produced by the SD model for a certain SD-Task must be the same as the staff profile produced by the PERT/CPM model for the corresponding sub-network mapped to that task through a structural link. This activity requires the structural links as an input and uses the data consistency links as the “windows” to check the data.

- (A3) *structural consistency checking* – the structural characteristics of one model must respect certain conditions that depend on the structural characteristics of the other model. This can be formally specified and verified by defining, analytically or through an algorithm, the conditions that must be respected. For example, “any task in the PERT/CPM model must be mapped to *one and only one* SD-Task in the SD model”; or “the work dependencies between two SD-Tasks in the SD model are derived from the task dependencies between the corresponding sub-networks in the PERT/CPM plan according to the following relationship...”. This activity uses the structural links as the “channels” to check the structure.
- (A4) *structural-data consistency checking* – the structure of one model must respect a certain condition depending on the input and/or output data in the other model. Depending on the specific data-structural link, this may be more or less difficult to specify and verify formally. It also depends on the particular variables used in one model which values represent structural characteristics or have structural implications on the other model. This activity uses the data-structural links as the “window” to check the structure of one model based on the data of the other model.
- (A5) *structural readjustment* – the structure of one model and/or the structural links between the two models are readjusted so that both structural and data consistency is achieved. This activity assumes that prior to being implemented the two models are not consistent in terms of structure and/or data. It is difficult to be specified formally, except for simple situations. This is because there can be many ways in which one model can be readjusted to satisfy structural and data consistency with the other. Of course, the “valid” solution will also have to be consistent with other aspects of the real world, which are not captured in any of the models. But even considering these restrictions, there can be several possible solutions. The previous activities (A2), (A3) and

(A4) check *what* may be inconsistent. This activity will have to specify *how to* change the structure of one model to solve the problem. For example, if the SD model was re-calibrated to reproduce an alternative plan where the schedule of a SD-Task was extended, then the PERT/CPM needs to be readjusted so that the critical path of the corresponding sub-network is also extended by the same amount – some algorithms that support some level of automation of this task will be discussed later. Structural readjustment of one model will often require the updating of structural links, which in turn may also imply updating the definition of the data links between the two models. These updates will take place within the activities of the process logic.

The dynamic nature of the links

As considered in the activities (M3a) and (P5) of the SYDPIM process logic, the analytical links may need to be updated as the structure of one model is changed in the course of the project. This will happen mainly when the structure of one model is adjusted either to become consistent with other, as just described in elementary activity (A5) above, or to reflect actual results. For example, re-planning the project future in the PERT/CPM model will most likely lead to adding or removing tasks from the plan and changing dependencies.

Various control actions throughout the project life-cycle will lead to continuous changes to the project plan and thereby to the structure and data of the two models. Therefore, the implementation of SYDPIM requires that the analytical links established between are *dynamic*. This has to main implications:

- (1) the user will have to identify when an update of the analytical links is required and perform this operation;
- (2) this creates the concept of “past versions” of the analytical links. Whenever a the analytical links are updated, is it necessary to keep a record of the previous version? If so, for what purpose?

Regarding (1), the structural links will require changes whenever the structure of one of the models is changed. A structural change in one of the models may require an update of the mapping of tasks, resources and dependencies between the two models. For example, if new tasks are added to the PERT/CPM plan, then these

need to be mapped into a specific SD-Task in the SD model. The new dependencies created in the PERT/CPM plan may also require an update of the dynamic dependencies in the SD model. Likewise, if a new SD-Task is added to the architecture of the SD model then some PERT/CPM tasks may need to be re-mapped into this SD-Task. Changes in the structural links may also result from modelling options: for example, the project manager may want to re-map some PERT/CPM tasks currently mapped into the design SD-Task into the coding SD-Task.

Once the structural links are changed, the data links may also need to be updated. For example, if a new SD-Task was added then it is necessary to specify the data-links to be used with the PERT/CPM model for the activities of (A1) data exchange and (A2) data-consistency checking. Changes in the structural links may also require changes in the data-structural links. As it will be shown later, the more common structural changes are the re-planning adjustments in the PERT/CPM model: as the plan is updated with actual results and changed for the future, new tasks are added or removed, and precedence relationships are changed.

Regarding (2), as it will be seen the recording of past versions of the analytical links is desirable. In SYDPIM basic mode it will be assumed that this is implemented. Past versions of the analytical links are required so that the relationships between the two models are interpreted correctly in the past segment of the project. If in a certain moment in the past, the structure of the two models was different than it is now then mapping of tasks and resources was probably also different. When behaviour patterns are extracted to characterise the project past behaviour based on the past versions of the models, then the analytical links used must be the ones which were linking those past versions of the models. For example, the planned duration of the SD-Task "Design" will correspond to the critical path of the PERT/CPM sub-network mapped to it. If the project is now in month 4 and the specific PERT/CPM tasks mapped to "Design" has changed from month 1, then in order to derive the planned duration in month 1, the past version of the analytical links as specified in that month must be used. Otherwise, the wrong duration could be derived. As it will be seen, in SYDPIM basic mode the main requirement is the recording of past versions of the structural links in order to derive output-output data-consistency links. Nevertheless, as a general principle SYDPIM advocates

that all links of past versions are recorded (i.e. structural, data, and data-structural). As a consequence of this, the SYDPIM formal object “Analytical Links” will have to store various versions of the links over-time.

Overview of SYDPIM basic principles

An overview of the basic principles of model integration as discussed in this section is presented in figure 7.11 below.

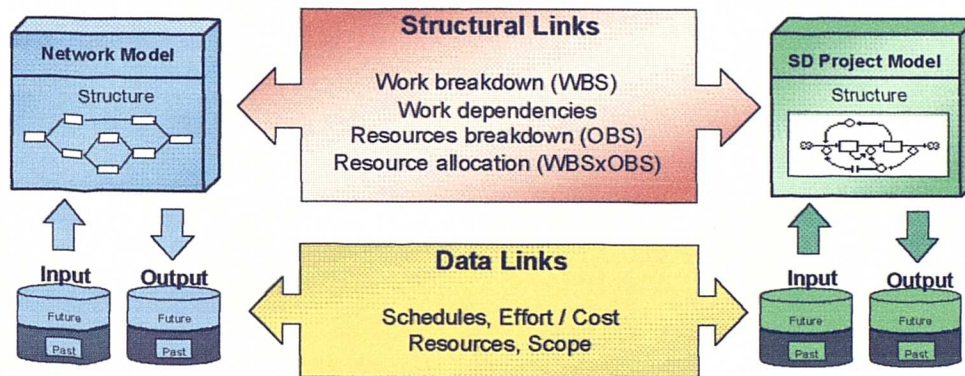


Figure 7.11 – Overview of SYDPIM principles of model integration

In SYDPIM basics mode, a specific set of analytical links is proposed. Before that, let us have a look at some illustrative examples of how these links look like and are used.

Illustrative examples of SYDPIM links

Although the concepts discussed above are extremely important as the basic principles of SYDPIM, they are also abstract and not easy to understand at a first reading. In practice, however, data and structural links tend to become meaningful and more clear. For the sake of clarity a brief illustrative example is here presented as shown in figure 7.12.

As an example of a structural link, the detailed tasks in a PERT/CPM plan (in some projects there can be thousands!) are grouped and mapped into higher-level SD-Tasks (in practice, generally no more than 10). So any detailed task in the network is captured in only one SD-Task in the SD model. This mapping originates

“clusters” of tasks in the PERT/CPM plan, each corresponding to a single SD-Task (as shown the figure 7.12). This is an example of a “work breakdown structural link”.

Another example of a structural links are the dynamic dependencies between the SD-Tasks in the SD model (e.g. see Ford and Sterman 1998 for a discussion). These can be derived from the elementary precedence relationships in the PERT/CPM model, or at least must be consistent with these. For example, in figure 7.12, the dynamic relationship between the SD-Tasks C and D should be derived from (or be consistent with) the two precedence relationships that cross the two correspondent “clusters” of tasks in the PERT/CPM model. This is an example of a “work dependencies structural link”.

Examples of a data links are: (1) the profile of resources allocated to a sub-network (or cluster) of tasks in the PERT/CPM plan, which is an output of the model, must match the profile specified for the SD-Task in the SD model, which is an input to this model. This could be an “output-input data exchange link”; (2) the cumulative effort spend by the resources in the group of tasks in the PERT/CPM plan, which is an output of the model, must match the pattern produced by the correspondent SD-Task in the SD model, which is also an output of the model. This could be an “output-output data consistency link”.

In summary, four types of links were identified:

- *work breakdown structural link* – detailed PERT/CPM tasks are grouped and mapped into the higher level SD-Tasks;
- *work dependencies structural link* – dynamic SD-Task dependencies are derived from PERT/CPM precedence relationships between the corresponding sub-networks;
- *data exchange link* – the resultant resource profile of a group of tasks in the PERT/CPM plan is an input to the planned resource profile in the SD model;
- *data consistency link* – when calibrated for a same project plan, the behaviour produced by both models must be the same.

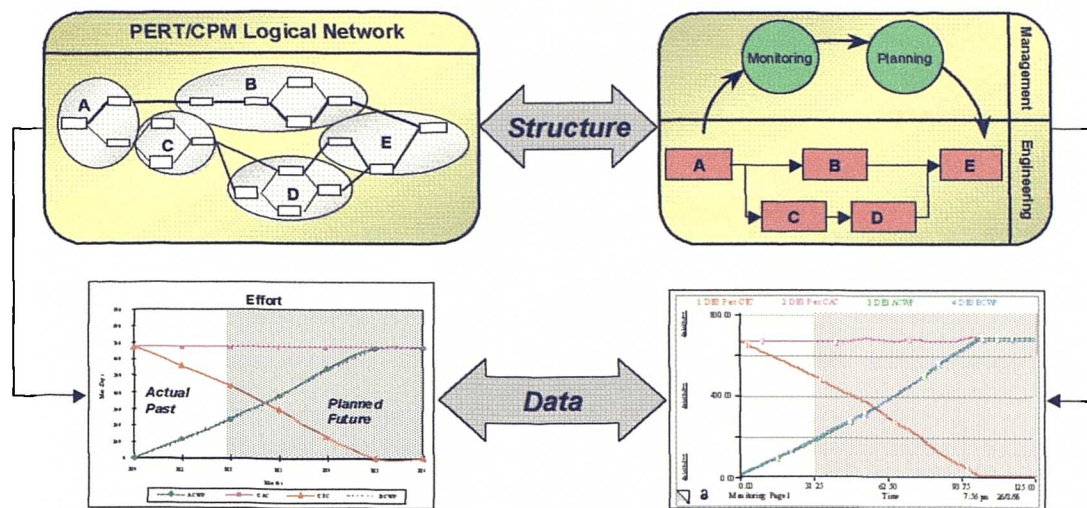


Figure 7.12 – An illustrative example of SYDPIM model integration

The analytical links of SYDPIM basic mode

The analytical links proposed in the SYDPIM basic mode are based on a set of assumptions regarding the PERT/CPM model and the SD model. Most of these assumptions were already identified when the models were described as objects. Some further assumptions will be required for the SD model, which are revised in this section.

The specification of the analytical links requires a formal specification language, which is also proposed in this section. This is an extension of the notation used for the specification of the objects. As with the objects, this language must be both formal and simple enough to be easily understandable by a management audience.

Assumptions for the PERT/CPM model

The following assumptions were proposed when this model was described as an object:

- (1) *components of a work plan* – a PERT/CPM work plan will comprise the following six components: tasks, finish-to-start dependencies, resources, profiles of resources availability, and allocation of resources to tasks;

- (2) *types of tasks* – there are two main types of tasks considered: engineering and management. A further breakdown of engineering tasks into sub-types may be considered in SYDPIM basic mode for illustrative purposes;
- (3) *task fields* – the following fields will be available in each task of a PERT/CPM plan: schedule dates (planned/actual, start/finish), duration (planned/actual), effort/cost (planned/actual), scope (planned/actual), resources (planned/actual);
- (4) *availability of initial plan* – the initial plan will always be available in the PERT/CPM model.
- (5) *availability of past plans* – it will be considered that past versions of the PERT/CPM plan will be available in the PERT/CPM model. Nevertheless, where necessary possible ways of handling the lack of these plans will be proposed;
- (6) *types of plans and level of detail* – a single plan covering the full project life-cycle will be considered. This plan will aggregate the three planning levels (operational, tactical, strategic) and may thereby have a varying level of detail over-time.

The analytical links proposed in this section are based on these assumptions.

Assumptions for the SD model

The following assumptions were proposed when this model was described as an object:

- (1) it is possible to represent a project plan in the SD model as an input, which will include the following elements: schedule, budget, scope, and resource allocation and availability profiles. Work and resource break-down may be considered but this is not mandatory;
- (2) it is possible to represent a set of project conditions, like risks, in the SD model as an input. No specific set of project conditions is required;
- (3) when simulated the model will produce some form of diagnosis information;
- (4) the model will be able to produce the project behaviour patterns to be extracted from the PERT/CPM model (as specified in the data contents of the objects “PERT/CPM past behaviour” and “PERT/CPM future behaviour”);

- (5) the model will allow the user to make changes to the project plan mentioned in (1);
- (6) the model will allow the user to make changes to the project conditions mentioned in (2).

Apart from these assumptions, there are two characteristics of the SD model that have a considerable impact on the number of analytical links that can be established with the PERT/CPM model: (i) the possible work and resource break-down within the model, and (ii) the internal work concurrency considered within SD-Tasks. Regarding these two issues, in order to include some important types of analytical links in the SYDPIM basic mode which otherwise could not be exemplified, further assumptions will be considered.

Internal work concurrency has to do with internal work progress within a task restricting the amount of work that can be started within the same task (i.e. an intra-Task work-progress dependency; see Chapter 7). If this phenomenon is explicitly considered in the SD model, then important structural dependency links can be established with the PERT/CPM model.

The work break-down also leads to the possibility of inter-task work concurrency being considered (see chapter 7). Work concurrency among tasks determines how much work can be started in the successor task depending on the work progress in the predecessor task. Again, if this type of concurrency is considered explicitly in the SD model, important structural dependency links can be established with the PERT/CPM model.

Work and resources break-down also lead to other critical structural links which have to do with the mapping of work packages and resources from one model to the other.

When work break-down is considered within the SD model, there is another factor that affects the analytical links: the type of work being accomplished within each sub-task. This has to do with the mapping of tasks between the models. Two types of tasks were already assumed for the PERT/CPM model: management and engineering. The SYDPIM model development method (see chapter 7) proposes

three types of tasks for a SD project model: engineering, management, and human resource management.

Considering these issues, the following assumptions will be further considered:

- (7) the SD model will consider both work and a resources breakdown. The specific number of SD-Tasks and type of resources is not relevant;
- (8) the SD model will consider explicitly inter-task work-progress dependencies;
- (9) the SD model will consider explicitly intra-task work-progress dependencies;
- (10) the sub-tasks in the SD model will be classified into three possible types: engineering, management, and human resource management.

The analytical links proposed are based on these assumptions about the two types of models.

Formal specification language for analytical links

What is an analytical links and hence what needs to be contained within its specification? An analytical link is basically a form of relationship between elements of the two models. The specification of such relationship will make reference to the relevant elements of the models and will determine how these are related.

The two models have been specified as objects and a specific notation has already been proposed to make reference to an object's components or elements. The specification of how these elements are related will consist of formal conditions that must be verified. These conditions will be specified using common mathematical notation and well-known data structures to which most managers are familiar, with like a matrix.

Another important feature of the analytical links in SYDPIM is their ability to answer specific questions. For example, what are the PERT/CPM tasks mapped to a certain SD-Task? The answer to this type of question can only be achieved through a process which inspects the conditions and data-structures contained within the link. Like with the objects these processes are formally specified as operators.

The third important feature of an analytical link is the “validity” of its conditions. For example, in the analytical link that maps PERT/CPM tasks to SD-Tasks, a certain PERT/CPM task cannot be mapped to more than one SD-Task. Like with the objects, the validity of an analytical link is specified through a set of “validity conditions”.

In order to illustrate these concepts, let us consider the example of a structural link that maps the PERT/CPM tasks to the SD-Tasks. This could be specified as follows:

```

Structural WBS_Link(PERT/CPM Model, SD Model) =
  Relationships =
    WBS-Map      : {(SD-Taski, {taskk})}
  Operators =
    Is_mapped    : taskk x SD-Taski → T | F
    Mapped_to_SD: SD-Taski → {taskk}
  Validity =
    No_multiple_map =
      IF Is_mapped(taskk, SD-Taski) THEN
        whatever j NOT Is_mapped(taskk, SD-Taskj)
    
```

Objects variables:

task_k = PERT/CPM model.Current PERT/CPM plan.<x>.Tasksk[k]
 <x> = Past segment or Future segment

SD-Task_k = SD Model.SD model architecture.SD Tasks[k]

This analytical link has two input parameters which are the two models being linked. Like any other analytical link, this structural link has three components: relationships, operators and validity conditions. Only one relationship is considered and this is specified as a set of couples formed by a SD-Task and a set of PERT/CPM tasks. This is clearly a convenient way to specify that to each SD-Tasks one or more PERT/CPM tasks are mapped to it. This relationship can easily be represented in a matrix. The notation used is the same as the one proposed for the objects.

Two operators are considered. The first checks whether a specific PERT/CPM is mapped to a specified SD-Task and returns a “True” or “False”. The other operator

returns the set of PERT/CPM tasks mapped to a specific SD-Task. The notation used to specify the operators is the same as in the objects and is straightforward: it specifies the inputs and the outputs and what the operator does is described by its name – when this name is not clear enough an informal description will be added below. The specific algorithm that implements an operator is not presented in SYDPIM because that is a particular implementation issue – of course, all operators presented are feasible to implement.

Finally, this analytical link considers one condition of validity called “No_multiple_map”, which imposes that one PERT/CPM task cannot be mapped to more than one SD-Task. The notation used is based on well-known logical if-clauses and on calling the operators – the same as in the specification of the objects.

As mentioned, analytical links make reference to variables in the models and hence to elements of the correspondent objects. This reference is made by using short names (to improve readability) and their true long name is identified below in “Objects variables”.

In order to reference the contents of the analytical links, a similar notation proposed for the objects can be used. For example:

- Structural WBS Link(PERT/CPM model, SD model).Relationships.WBS-Map – accesses the data-structure that maps the PERT/CPM task to the SD tasks;
- Structural WBS Link(PERT/CPM model, SD model).Operators.Is_mapped(task_k, SD-Task_i) – calls the operator “Is_mapped” to check whether task_k is mapped to SD-Task_i.

Like with the objects, this notation is simple and unambiguous but it tends to lead to long names. In order to improve readability, shorter equivalent names will be used where necessary. These will be referenced to the true long names that specify the contents of the analytical links.

Once the assumptions about the two types of models have been revised and a formal specification language has been proposed, it is now possible to specify the SYDPIM links. As already mentioned, there are three types of links that can be

established: structural, data and data-structural links. Since the last two are based on the structural links, these will be described first.

Structural links

Overview

Structural links establish relationships between the structures of both models. The structure of each model consists of four main elements: (i) tasks, (ii) dependencies, (iii) resources and (iv) resource allocation to tasks. However, there are important differences between these elements of the two models. The following assumptions will be considered:

- (1) *tasks* – in most cases, the tasks in the PERT/CPM model are defined at a more detailed level than the tasks in the SD model (referred to as SD-Tasks). For this reason, it will often happen that a sub-network of tasks in the PERT/CPM model will correspond to a single SD-Task. While the reverse situation is also possible, this would imply that the PERT/CPM plan is so aggregated that its simplistic representation of a work task would be of little use. If this happens, however, structural links can still be implemented in the reverse way (and in fact they will be simpler). Comments will be made where appropriate;
- (2) *resources* – a similar scenario applies. It will be assumed that sets of elementary resources in the PERT/CPM model will correspond to resource categories in the SD model (referred to as SD-Resources);
- (3) *dependencies* – based on the assumptions that the SD-Tasks are more aggregated, the dependencies between SD-Tasks will also be more aggregated than the elementary finish-to-start dependencies in the PERT/CPM model. Again this will be an assumption. There are also intra-task dependencies considered explicitly within SD-Tasks but not in PERT/CPM elementary tasks. Since a SD-Task corresponds to a PERT/CPM sub-network, its intra-task work dependency will be more aggregated and will correspond to the elementary dependencies within that sub-network;
- (4) *resource allocation* – resource allocation to a task can be considered in two ways: as a simple full-time allocation or as a profile over-time. Some PERT/CPM software tools allow for individual resources to be allocated to elementary tasks as a profile. However, many tools do not consider this type of

allocation nor this is a dominant PERT/CPM practice. For the sake of simplicity, a full-time allocation will be considered. Where appropriate, comments will be made to explain how a profile could be considered instead. Regarding the SD model, given the higher level of aggregation, a profile over-time will be considered. SD project models that do not support this type of allocation are likely to be over-simplistic for SYDPIM purposes.

Given these assumptions, it is now possible to specify the structural analytical links of SYDPIM basic mode. Overall there are five types of structural links:

- (1) correspondence of work breakdown
- (2) correspondence of organisation breakdown
- (3) correspondence of work dependencies
- (4) consistency of resource allocation
- (5) consistency of work dependencies.

These are now described separately.

Work breakdown links (structural correspondence)

Overview

In both types of models, one of the key elements that represents the project system is the project work structure. This is particularly relevant for the PERT/CPM model which is primarily based on the formal specification of the project work break-down structure (WBS). The WBS is the decomposition of the *whole* project work into successive levels of detail, down to elementary tasks. Ideally, these elementary tasks feed directly the PERT/CPM logical network.

A SD project model also considers the project work explicitly. The dynamics of a project model typically consider a life-cycle of phases through which various work units (i.e. elementary work tasks) flow towards completion. However, a SD model gives less attention to the details of the work decomposition. It decomposes the project into major work tasks (SD-Tasks), but these are defined at a much more aggregate level than the PERT/CPM tasks.

In SYDPIM, the structural links of work breakdown between the two models consist of a formal mapping of the PERT/CPM tasks into the SD-Tasks. This can be done using a simple matrix data-structure (Rodrigues and Williams 1997; Rodrigues 1997).

Establishing the links: basic principles

In practice, it is management responsibility to establish this mapping. This way, management will have to decide to which particular SD-Task should each elementary task in a PERT/CPM plan be mapped to. Since SD-Tasks are more aggregated, this relationship is of type “one-to-many” (i.e. one SD-Task to many PERT/CPM tasks). How to decide about this mapping? In concept, a *PERT/CPM task should be mapped to a SD-Task wherein its work contents, or scope, will be accomplished*. For example, if the SD model contains a SD-Task called “System Design” representing the accomplishment of the whole design work in the project, then the PERT/CPM task called “Interface Design” should be mapped to this SD-Task.

Although this concept is simple, there is a number of issues that need to be addressed carefully:

- can this mapping be totally arbitrary, depending solely on management views? Or will it be constrained? And by what factors?
- can this mapping be “partial”? This is, can a PERT/CPM task be mapped to more than one SD-Task?

Regarding the first question, in general the mapping can be arbitrary as far as the consistency of the scope contents is respected. For example, the user may want to consider that a certain design task in the PERT/CPM model is mapped to a SD-Task called “Coding”. This is because the user knows that this specific design work will be accomplished in the coding phase of the project. So this mapping is allowed as far as, in the SD model, the design work of the PERT/CPM task is considered to be accomplished within that SD-Task. The second question raises the need for the definition of some basic rules and conditions that must be respected. These are now discussed.

Establishing the links: rules and conditions

Scope consistency is the key principle of structural links. Since in both models the work tasks are classified according to their type of scope contents, it is possible to define some general rules that will help to prevent undesired mappings. The table 7.2 below shows the proposed rules.

		SD Model		
		Engineering	Management	HRM
PERT/CPM Model	Management	Possibly	Yes	Yes
	Engineering	Yes	No	No

Table 7.2 – Mapping rules for work breakdown structural links

Engineering tasks in the PERT/CPM model represent direct product development work and hence they can only be mapped to engineering tasks in the SD model. By definition, the work accomplished in these tasks does not represent any managerial work of project control. On the other hand, management tasks in the PERT/CPM model represent this type of work and so can be mapped to “SD Management” or HRM SD-Tasks. The mapping of PERT/CPM management tasks to engineering type of SD-Tasks is also considered. This is because some times low-level management tasks in the PERT/CPM plan may have more to do with internal control of certain product development work (e.g. team management). Most likely, there will be no SD management task considered at that level of detail. Therefore, the work of such PERT/CPM task is better represented as direct product development work and thereby in a SD Engineering task.

Partial mapping is also a critical issue. It is easy to anticipate that the implementation of analytical links will be simpler if partial mapping is not allowed. Partial mapping would appear appropriate when the work contents of a PERT/CPM task are actually accomplished in more than one SD-Task of the SD model. However, because the level of aggregation of a SD-Task is higher, in principle, this is unlikely to occur. But if it does occur, because the PERT/CPM task is a small elementary work package, the impact of ignoring partial mapping would probably be small. On the other hand, if partial mapping is to be considered, then management will have to decide how the contents of the PERT/CPM task will be distributed to the various SD-Tasks in terms of scope, effort, resources and schedule. While

percentages could be a simple solution to the first two elements, distributing the schedule and the resources would not be so simple and would probably require a more complex type of input from the user. Most likely, the mathematics of the analytical data-links would also become considerably more complex. So if partial mapping is difficult to implement, can it be safely “ignored”? The need for partial mapping is more likely when the work breakdown in the two models follows a different criteria. There are various possible criteria for work decomposition: product functionality, organisational structure, organisational geography, structure of the development process, among others. As an example, if the work decomposition in the PERT/CPM model is “product oriented” and in the SD model it is “process oriented”, then the “product tasks” in the PERT/CPM model will contain a work scope which may be spread throughout the various “process SD-Tasks”. On the other hand, if the work breakdown in the two models is based on a common WBS, then most likely partial mapping will not be required. In this case, the mapping process can even be done automatically: a SD-Task will correspond to a high-level non-terminal task in the WBS and all the PERT/CPM terminal tasks that derive from that node will be mapped to it.

For these reasons, partial mapping is not considered in SYDPIM basic mode. It is also recommended that the work breakdown in the two models follows the same criteria, preferably based on a common WBS.

Other obvious rules are that :

- (1) every PERT/CPM task must be mapped to a certain SD-Task;
- (2) at least, one PERT/CPM task must be mapped to a SD-Task.

These two rules impose that no work task should remain unmapped to the other model. If a certain amount of work is represented in one model, then this work exists in the project and therefore should also be represented in the other model. This is, the “work domain” of the two models is the same.

In summary, the rules and conditions to establish work breakdown links of structural correspondence are as follows:

- (a) a PERT/CPM task should be mapped to a SD-Task if the accomplishment of the work contents of the first is simulated in the second;

- (b) no PERT/CPM task or SD-Task remains unmapped;
- (c) PERT/CPM engineering tasks cannot be mapped to management or HRM SD-Tasks;
- (d) PERT/CPM management tasks can be mapped to engineering SD-Tasks when they stand at a low management level not represented in SD management tasks;
- (e) No partial mapping is allowed: a PERT/CPM task cannot be mapped to more than one SD-Task.

Implementing the links

As suggested, the work breakdown links of structural correspondence can now be implemented in a matrix. The generic structure of this matrix is shown in table 7.3. The PERT/CPM tasks are divided in two main categories: management and engineering. The SD-Tasks are divided in three categories: engineering, management and human resource management. Some specific tasks and mappings are considered as example:

- (1) both engineering and management PERT/CPM tasks are mapped to the SD-Task called "Requirements Specification". This is because these SD-Tasks will represent the whole project work of specifying the system requirements, including low-level management work, like the development of plan for this requirements work and the periodic review of progress (i.e. tasks "Develop Reqs Plan" and "Review Reqs Status");
- (2) the PERT/CPM tasks of "whitebox" and "blackbox" testing of the system interface are mapped to the SD-Task "System Testing";
- (3) the PERT/CPM task which represents the work of developing test plans is mapped to the SD management task "Testing Control", which probably represents in the SD model the management work of controlling testing activities;
- (4) all other examples are straightforward. As suggested in the proposed mapping rules, PERT/CPM engineering tasks cannot be mapped to the SD management tasks.

A matrix like this can be implemented manually or using a computer software tool. For example, some PERT/CPM tools allow the user to specify extra information

fields in each PERT/CPM task. This type of fields can be used to specify the SD task to which each PERT/CPM task is mapped to. Other intermediate tools linked with the PERT/CPM tool, like a spreadsheet or a database application, can be used to implement this matrix. As a process methodology, SYDPIM does not depend on the particular implementation platform.

Once this matrix is specified in the beginning of the project, it may need to be updated in each control cycle. This update is considered explicitly in activities (M3a) and (P5) of the SYDPIM process logic (see figures 7.6 and 7.7). For example, if re-planning is taking in the PERT/CPM model and new tasks are mapped to the PERT/CPM plan, then before the SD model is calibrated to test this new plan, these new PERT/CPM task need to be mapped to a SD-Task. As the matrix is updated, its past version is stored within the object "Analytical links", which will store the "dynamic" history of the links.

Formal specification

The formal specification of the structural links of work correspondence (SC-WBS) is shown in the following box (also in appendix D.1).

The mapping presented in table 7.3 is specified through a set of couples formed by a SD-Task and a set of PERT/CPM tasks. This specification is based on the assumption stated above that a SD-Task will aggregate one or more PERT/CPM task, while one PERT/CPM task can only be mapped to one SD-Task. As also discussed previously, while the reverse is conceptually possible, it is not very likely to be of practical use; adjusting the specification of the links to such scenario would bring unnecessary complexity. For example, the conditions that would prevent partial mapping would be:

- for each SD-Task to which more than one PERT/CPM task is mapped, then each of these PERT/CPM tasks cannot be mapped to another SD-Task;
- for each PERT/CPM task mapped to more than one SD-Task, then each of these SD-Task cannot have any other PERT/CPM task mapped to it.

SC-WBS: Structural correspondence of work breakdown

SC-WBS(PERT/CPM Plan, SD Model) =

Relationships =

WBS-Map : {{SD-Task_i, {task_k,}}}

Operators =

Is_mapped : task_k x SD-Task_i → T | F

Mapped_to_SD : SD-Task_i → {task_k}

Is PERT_ENG : task_k → T | F

Is PERT_MAN : task_k → T | F

Is_SD_ENG : SD-Task_k → T | F

Is_SD_MAN : SD-Task_k → T | F

Is_SD_HRM : SD-Task_k → T | F

Validity =

No_partial_map =

IF Is_mapped(task_k, SD-Task_i) THEN

WHATEVER SD-Task_j NOT Is_mapped(task_k, SD-Task_j)

SD_mapping =

FOR EACH SD-Task_k THERE IS AT LEAST ONE task_i

SO THAT Is_mapped(SD-Task_k, task_i)

PERT/CPM_mapping =

FOR EACH task_k THERE IS AT LEAST ONE SD-Task_i

SO THAT Is_mapped(SD-Task_i, task_k)

Task_types:

IF Is_mapped(task_k, SD-Task_i) THEN

IF Is PERT_ENG(task_k) THEN Is_SD_ENG(SD-Task_i)

Objects variables:

task_k = PERT/CPM Plan.<x>.Tasksk[k]

<x> = Past segment or Future segment

SD-Task_k = SD Model.SD model architecture.SD Tasks[k]

It would also be a more generic scenario, which would have further impacts in the specification of other SYDPIM links. Since the benefits are not worth the extra complexity, this will not be considered in SYDPIM basic mode.

The specification presented above allows for an easy identification of the PERT/CPM tasks mapped to a specific SD-Task_k, as well as to each of these individual task_i, as follows:

- SC-WBS(PERT/CPM Plan, SD Model).Relationships.WBS-Map[k] – accesses the set of PERT/CPM tasks mapped to SD-Task_k
- SC-WBS(PERT/CPM Plan, SD Model).Relationships.WBS-Map[k][i] – accesses the PERT/CPM task_i mapped to the SD-Task_k

The first reference is the output of the operator specified as “Mapped_to_SD”. Other operators are also specified above and their meaning is straightforward. The four validity conditions specified make use of these operators. These conditions are no more than the formal translation of the conditions of structural correspondence already discussed: no task can remain unmapped, there is no partial mapping, and PERT/CPM engineering tasks can only be mapped to SD engineering type of tasks.

Organisation breakdown links (structural correspondence)

Overview

Organisation breakdown links are similar to the work breakdown links. Instead of mapping the project work represented in both models, they map the project resources.

Project resources are an important element in both types of models. In the PERT/CPM model, project resources can represent human or material resources and they can also represent a single resource or a group of resources. While material resources can be critical in certain types of projects, human resources are typically the more important ones in design and development projects, for planning purposes. Most PERT/CPM software tools encourage the user to specify human resources as specific individuals. Typically, this helps integration with other Enterprise Resource Planning (ERP) applications in use.

On the other hand, in a SD model the project resources are not considered as specific discrete individuals or machines. The continuous and aggregate nature of the SD modelling paradigm encourages the specification of human resources grouped into categories. For example, “designers”, “coders” or “testers”. These groups of resources are defined in SYDPIM as SD-Resources. It is not mandatory that these refer to human resources only, but in most real life situations that will be the case.

Establishing the links: basic principles

Like with the work breakdown links, the mapping of resources between the two models will generate “one-to-many” type of relationships (i.e. one SD-Resource category to many PERT/CPM individual resources). Again, these relationships can be implemented in a matrix.

How is this mapping established? Are there any rules that can be proposed to prevent invalid mapping? Once more, this mapping is of management responsibility. In principle, there is no major restriction to prevent the mapping a certain PERT/CPM resource to a SD-Resource category. The grouping of PERT/CPM individuals into SD-Resources should be consistent with what these resources are in reality. Generally, SD-Resources represent groups of staff that will accomplish a certain type of work. The individual PERT/CPM resources mapped to a certain SD-Resource should accomplish the same type of work. For example, it will make sense that a specific designer in the PERT/CPM model is mapped to the SD-Resource category “Designers” and not to the SD-Resource “Testers”. This general principle also implies some consistency regarding the allocation of the resource to the work tasks: a PERT/CPM resource should be mapped to a SD-Resource category which is allocated to the same work in the SD model. However, this type of consistency is formally addressed by structural consistency links.

Establishing the links: rules and conditions

Like with the work breakdown links, if the resources were to be classified in both models into certain types, like “Managers” and “Designers”, it would make sense to propose certain mapping rules like “PERT/CPM Designers cannot be mapped to SD Testers”. However, unlike with the work tasks, SYDPIM does not impose any particular classification of resources in any of the two models. Therefore no particular set of rules can be proposed based on resource classification. SYDPIM does not impose any resource classification because this has not proven useful in practice. Unlike work tasks, in reality, a same resource can perform many types of activities and hence a strict classification is more likely to raise difficulties than to help in the integration process.

There are only two basic rules that must be respected:

- (1) every PERT/CPM resource must be mapped to a certain SD-Task;
- (2) at least one PERT/CPM resource must be mapped to a SD-Resource.

No resource can remain unmapped to other model. This implies that the “resources domain” of the two models is the same.

Partial mapping could again be considered – i.e. a certain PERT/CPM resource being mapped to more than one resource category. For example, “John Someone” may work as a designer early in the project and later as a tester. So it needs to be mapped to both SD-Resource types “Designers” and “Testers” at different periods of time. It could even happen that such PERT/CPM resource would have to be mapped to these two SD-Resources in a same period of time. This type of partial mapping would require extra information regarding the periods of time and the percentage of mapping. Clearly, it brings more complexity to the specification of structural consistency links and of data-links. Alternatively, the need for partial resource mapping can be overcome in two ways: (i) by considering two or more PERT/CPM resources to represent a single resource in the real world (e.g. “John Someone Designer” and “John Someone Tester”), or (ii) by readjusting resource allocation in the SD model (e.g. some Designers working in the Testing SD-Task later in the project). These solutions are simpler to implement and reasonable to be assumed. Therefore, for the sake of simplicity of the structural and data links, partial resource mapping will not be considered in SYDPIM basic mode. This type of mapping can however be considered and implemented in a more complex and detailed implementation of SYDPIM. The SYDPIM basic mode is here proposed as a starting point.

In summary, the rules and conditions to establish organisation breakdown links of structural correspondence are as follows:

- (a) a PERT/CPM resource should be mapped to a SD-Resource if the work to which the first is allocated in the PERT/CPM model is the same as the work the SD-Resource will perform in the SD model;
- (b) no PERT/CPM resource or SD-Resource remains unmapped;

- (c) no partial mapping is allowed: a PERT/CPM resource cannot be mapped to more than one SD-Resource.

Implementing the links

A simple matrix like the one presented below can be used to implement the organisation structural links of SYDPIM basic mode. Because partial mapping is not considered, each PERT/CPM resource can only be mapped to one SD-Resource. It is important to note that, in the end, this mapping only classifies PERT/CPM resources into the SD-Resource categories. There is no explicit implication of *quantities* allocated to the project and to the work tasks. For example, this matrix is not intended to specify how many “Designers” there are in the project and their allocation profile over-time. As a structural link, this matrix simply establish the conceptual correspondence between the project resources specified in both models. For example, “John” who is scheduled to work in a number of tasks in the PERT/CPM plan, is somewhere in the SD model accumulated in a stock of the life-cycle of the SD-Resource “Designers”. The examples below refer to human resources, but the same principles apply to material resources in case these are considered.

PERT/CPM Resources	SD-Resource Categories		
	Designers	Coders	Testers
John	✓	–	–
Carl	✓	–	–
Peter	–	✓	–
Champions (group)	–	–	✓
...

Matrix Specifying Organisation Breakdown Structural Correspondence Links

Again, this matrix can be implemented manually or using a PERT/CPM tool or any other appropriate application as a software platform.

Being a structural link, this matrix will need to be updated as the project plan is updated and resources are added, removed or re-scheduled in the project. This takes place in activities (M3a) and (P5) of the SYDPIM process logic (see figures 7.6 and 7.7). As this matrix is updated, its past version is stored within the object “Analytical links”, which will store the “dynamic” history of the links.

Formal specification

The formal specification of the structural links of organisational correspondence (SC-OBS) is presented in appendix D.1.

Like with the SC-WBS link, the relationship specified considers that one or more PERT/CPM resource is mapped to each SD-Resource. Two operators are specified, one that checks the mapping and the other produces the set of PERT/CPM resources mapped to a specific SD-Resource. The validity conditions impose that partial mapping is not allowed and that no resource remains unmapped.

Work dependency links (structural correspondence)

Overview

These links are also of structural correspondence. They specify how the elementary dependencies in the PERT/CPM plan relate to the work dependencies within the SD model. The following assumptions were proposed for the SYDPIM basic mode:

- (i) only sequential "finish-to-start" dependencies are used in the PERT/CPM model;
- (ii) some level of work breakdown is considered in the SD model and inter-task work dependencies are considered between SD-Tasks. These dependencies refer only to work progress and can be dynamic (see chapter 7);
- (iii) within each SD-Task, intra-task work dependencies are also considered. These dependencies also refer to work progress and can be dynamic (see chapter 7).

These assumptions restrict the two models in the following way:

- (1) in the PERT/CPM model:
 - (1.1) no lagged/overlapped dependencies are allowed in the PERT/CPM plan;
 - (1.2) no other type of elementary dependencies are allowed in the PERT/CPM plan (e.g. start-to-start);

(2) in the SD model

- (2.1) no other type of inter-task or intra-task dependencies that may be considered in the SD model (e.g. rework discovery or QA effects; see chapter 7) will have a correspondence in the PERT/CPM model. This is, the effects they capture are not represented explicitly in the PERT/CPM model.

Work dependencies links can be considered in a more complex way to include these possibilities. However, such links will be more complex and can be derived from the simpler links proposed in SYDPIM basic mode.

Establishing the links: basic principles

There are two aspects of relating PERT/CPM dependencies to SD dependencies: (i) which dependencies in one model are represented in what dependencies of the other model and (ii) how should a dependency of one model be defined quantitatively given its correspondence to the dependencies of the other model. The first issue refers to structural correspondence while the second refers to data-links. It was assumed that the SD dependencies are more aggregated, thereby corresponding to one or more PERT/CPM dependencies – again, a “one-to-many” type of relationship. Therefore, issue (i) takes the form of “which PERT/CPM dependencies are mapped to each SD dependency”, and issues (ii) takes the form of “how can be a dynamic SD dependency be derived from (or restricted by) the PERT/CPM task mapped into it”. The work dependency links of structural correspondence address only the first issue.

The specification of these links is once more a mapping process. So, how to decide which PERT/CPM dependencies should be mapped to each SD dependency? While this is again a decision of management responsibility, it is highly constrained by the work breakdown links of structural correspondence defined above. Therefore the work breakdown links must be specified before the work dependency links.

Establishing the links: rules and conditions

From the structural links, for each SD dependency it is possible to derive automatically a set of PERT/CPM dependencies which can *potentially* be mapped to it. From this set, management may wish to consider all the PERT/CPM dependencies or exclude some from the mapping. The following mapping rules are used to derive this set:

- (1) for every SD-Task in the SD model, the set of PERT/CPM dependencies that can be *potentially* mapped to its intra-task work dependency are those for which their successor and predecessor PERT/CPM tasks are mapped to that SD-Task, according the work breakdown structural correspondence links;
- (2) for every SD inter-task dependency in the SD model, the set of PERT/CPM dependencies that can be *potentially* mapped to it are those for which, according to the work breakdown structural correspondence links, their successor PERT/CPM task is mapped to the SD-Task successor of the SD dependency and their predecessor PERT/CPM task is mapped to the SD-Task predecessor of the SD dependency.

These rules can be specified formally in many ways. For example, a formal software specification language can be used to specify these sets as a data-structure, or to specify a function/procedure that produces these sets, based on the formal specification of the models also as data-structures. The following algorithms provide a semi-formal specification of the use of these rules to implement an automated mapping, assuming that management will consider all the potential dependencies:

Algorithm SC-WD-1: {Map PERT/CPM dependencies to SD intra-task dependencies}

```
FOR each <SD-Task in the SD model> DO
  FOR each <PERT/CPM dependency> DO
    IF <predecessor PERT/CPM task is mapped to SD-Task> AND
      <successor PERT/CPM task is mapped to SD-Task> THEN
      Map <PERT/CPM dependency> to <SD intra-task dependency>
      of <SD-Task>
    ENDIF
  ENDFOR
ENDFOR
```

Algorithm SC-WD-2: {Map PERT/CPM dependencies to SD inter-task dependencies}

```
FOR each <SD inter-task dependency in the SD model> DO
  FOR each <PERT/CPM dependency> DO
    IF <predecessor PERT/CPM task is mapped to predecessor SD-Task of
      SD inter-task dependency> AND <successor PERT/CPM task is
      mapped to successor SD-Task of SD inter-task dependency> THEN
      Map <PERT/CPM dependency> to <SD Inter-task dependency>
    ENDIF
  ENDFOR
ENDFOR
```

As expected these algorithms make use of the structural correspondence links of work breakdown, which are implicit in the conditions of the type "... PERT/CPM task is mapped to SD-Task...". They can be used to develop software routines or functions that produce the work dependency links as the output.

Once the work dependency links of structural correspondence can be derived automatically from the work breakdown links, it remains for management to decide whether some of the potential PERT/CPM dependencies should be excluded from this mapping. Once excluded, the PERT/CPM dependency cannot be re-mapped to another SD dependency. The mapping is not arbitrary: it must respect the two rules proposed above. Also, the mapping cannot be partial (i.e. a PERT/CPM dependency being mapped to more than one SD dependency), because according to these rules one PERT/CPM dependency can only belong to one specific SD dependency. This is true as far as the work breakdown links do not consider partial mapping. If this was considered, partial mapping of work dependencies would be possible and would be a complex issue to handle for now. So management only have the choice remove a PERT/CPM dependency from the mapping. But in what circumstances would this be appropriate? The presence of a PERT/CPM dependency in the work dependency links affects the quantitative definition of the SD dynamic dependencies. In principle, the more the PERT/CPM dependencies mapped to a SD dependency, the less the amount of work that can be done in parallel, thereby restricting more the potential work rate in the SD model. So, if a PERT/CPM dependency is removed it means that in the SD model the level of parallelism considered will be higher than in the PERT/CPM model. In principle, this should not happen because the two models would not be representing the same reality and hence would not be consistent. In practice however, there may be occasions where certain "finish-to-start" dependencies are not relevant and not being considered as restrictive. For example, some of these dependencies may be

used in the PERT/CPM plan just to represent ideal sequences of work accomplishment but without being necessarily implemented that way in the real world.

In summary, the rules and conditions to establish work breakdown links of structural correspondence are as follows:

- (a) a PERT/CPM task dependency can be potentially mapped to a SD intra-task dependency of a SD-Task if both predecessor and successor tasks of the first are mapped to that SD-Task. If mapped, the restriction to work accomplishment represented in the specific PERT/CPM dependency must be captured in the specific dynamic SD intra-task dependency;
- (b) a PERT/CPM task dependency can be potentially mapped to a SD inter-task dependency if the predecessor and successor tasks of both dependencies are mapped one another respectively. If mapped, the restriction to work accomplishment represented in the specific PERT/CPM dependency must be captured in the specific dynamic SD inter-task dependency;
- (c) no partial mapping is allowed: a PERT/CPM dependency cannot be mapped to more than one SD dependency.

Implementing the links

The links themselves can be represented and stored in a matrix. Separate matrices are proposed for SD intra-task and SD inter-task dependencies. This is because in the first case only one SD-Task is involved whereas in the second it is necessary to identify two SD-Tasks. These two matrixes are exemplified.

SD Intra-Task Dependency SD-Task	PERT/CPM Dependencies	
	Predecessor Task	Successor Task
Requirements Specification	Requirements Clarification	Formal Reqs Stage 1
	Formal Reqs Stage 1	Formal Reqs Stage 2
	Formal Reqs Stage 2	Final Reqs Specification
System Design	High level design	Detail Design Comp A
	High level design	Detail Design Comp B
	High level design	Detail Design Comp C
	Detail Design Comp A	Detail Design Comp B
	Detail Design Comp A	Review Detail Design A
	Detail Design Comp B	Review Detail Design B
	Detail Design Comp C	Review Detail Design C
	Review Detail Design A	Integrated Design
	Review Detail Design B	Integrated Design
Review Detail Design C	Integrated Design	
...

Matrix specifying work dependency links involving SD intra-task dependencies

According to this example, the intra-task dependency of the SD-Task “Requirements Specification” captures all the “finish-to-start” dependencies within the PERT/CPM sub-network which is composed by a simple sequence of four tasks. In the structural links of work breakdown, these four PERT/CPM tasks of this sub-network must be mapped to this SD-Task. Likewise, the intra-task dependency of the SD-Task “System Design” captures the ten PERT/CPM dependencies of the sub-network mapped to this SD-Task. In every case, both successor and predecessor tasks of the PERT/CPM dependency must be mapped to the SD-Task.

SD Inter-task Dependency		PERT/CPM Dependency	
Predecessor Task	Successor Task	Predecessor Task	Successor Task
Reqs Specification	System Design	Formal Reqs Stg 2	High level design
		Final Reqs Spec.	Detailed Design A
		Final Reqs Spec.	Detailed Design B
		Final Reqs Spec.	Detailed Design C
System Design	Coding Interface	Review Detail Des A	Coding HCI
		Review Detail Des B	Coding HCI
		Integrated Design	Unit test HCI
...

Matrix specifying work dependency links involving SD inter-task dependencies

The SD inter-task dependency from “Requirements Specification” to the “System Design” SD-Tasks captures the “finish-to-start” dependencies which in the PERT/CPM plan “cross” the two sub-networks mapped to each of these SD-Tasks. For example, the PERT/CPM task “Formal Reqs Stg2” belongs to the PERT/CPM

sub-network mapped to the SD-Task “Reqs Specification” and the PERT/CPM task “High level design” belongs to the sub-network mapped to the SD-Task “System Design”. Therefore, the “finish-to-start” dependency between these two PERT/CPM tasks crosses the two sub-networks and so it should be mapped to the SD inter-task dependency between the two SD-Tasks.

Since the work dependency links are specified upon the work breakdown links of structural correspondence, whenever these are changed the work dependency links will also need to be updated. This will happen whenever the PERT/CPM model is updated with past results and when the future is re-planned – activities (M3a) and (P5) of SYDPIM process logic.

Formal specification

The formal specification of the structural correspondence links of work dependency (SC-WD) is presented in appendix D.1.

Two relationships are considered: the mapping of PERT/CPM dependencies to SD intra-task dependencies, and the mapping of PERT/CPM dependencies to inter-task dependencies. The first mapping consists of relating PERT/CPM dependencies to SD-Tasks, since each SD-Task contains its own intra-task dynamic dependency within its structure. The operators specified check the mapping and produce the set of PERT/CPM dependencies mapped to a certain SD-Task or SD-Dependency. The validity conditions are a formal translation of the rules and conditions previously identified: no partial mapping is allowed and the successor and predecessor tasks in the two models must be mapped in the SC-WBS link in a consistent manner.

Consistency links

The three types of links proposed above are links of structural *correspondence* because their aim is to establish which elements of one model represent the same reality as certain elements of the other model. This correspondence is based on the mapping of: work tasks, organisational units (resources) and work dependencies. For each of these elements one type of link is considered. Some rules and conditions were proposed to ensure that the links are “valid”, meaning that they

establish a correspondence between elements in the two models that actually represent the same reality. The formal integrated use of the two models is based on the exercise of readjusting both models and the links between them so that consistency is achieved. Throughout the SYDPIM process logic there will be situations where the two models will not be consistent and where the links will not be valid. For example, when a new plan for the future is developed in the PERT/CPM model, the SD model will need to be re-calibrated and in some occasions the structural links may also need to be updated. This raises the need for two things: (i) to identify whether and which links need to be updated and (ii) to identify what needs to be changed in one model so that it becomes consistent with the other. In order to support this, SYDPIM considers that structural consistency links are established between the two models. The role of these links is not to establish correspondence between the two models but to check whether: (a) the models are consistently representing the same reality according to the correspondence links currently established, and (b) the structural correspondence links are consistent one another. In the later case it is important to note that while the rules and conditions proposed for the structural correspondence links ensure their individual “validity”, they do not ensure consistency among them (resource allocation is a typical case). So, if this checking fails, then either the models are not representing the same reality and so one model needs to be readjusted, or there is some inconsistency between the mapping of the structural correspondence links.

There are two types of structural consistency links proposed in SYDPIM basic mode: the ones that check resource allocation and the ones that check the work dependencies. These are now described below.

Resource allocation links (structural consistency)

These links check whether the structural correspondence links of work breakdown and the links of organisational breakdown are consistent one another regarding resource allocation. For the links to be consistent, the following condition must be verified for all resources in the PERT/CPM model:

If a PERT/CPM resource r is allocated to a certain PERT/CPM task t in the PERT/CPM model, then if: (i) according to the work breakdown structural correspondence links t is mapped to the SD-Task T and (ii) according to the organisation breakdown structural links r is mapped to the SD-Resource R , then within the SD model R must be allocated to T .

In practice this means that if “John” is allocated to work in the task “Design HCI” in the PERT/CPM plan and this task is mapped to the SD-Task “System Design” and “John” is also mapped to the SD-Resource “Designers”, then in the SD model the resource “Designers” must be allocated to the SD-Task “System Design”.

So if “Carl” is allocated to “Design HCI”, but “Carl” is mapped to the SD-Resource “Testers”, which is not allocated to the SD-Task “System Design”, then the mapping of the work breakdown between the two models is not consistent with the mapping of the organisation breakdown regarding resource allocation. This consistency link identifies which resource is causing the problem: in this case it is “Carl”. The four possible solutions to overcome this inconsistency are:

- (i) re-map “Carl” to “Designers”, thereby adjusting the organisation breakdown link;
- (ii) re-map a PERT/CPM task where Carl is working to “System Design”, thereby adjusting the work breakdown link;
- (iii) remove “Carl” from working in the task “Design HCI”, thereby adjusting the PERT/CPM model;
- (iv) allocate the resource “Testers” to “System Design”, thereby adjusting the SD model.

These potential solutions show that a structural inconsistency can have two distinctive causes:

- (a) the structural correspondence links are not consistent one another and hence need to be adjusted (solutions (i) and (ii));
- (b) the models are inconsistent, not representing the same reality (i.e. project plan), and so need to be readjusted (solutions (iii) and (iv)).

The formal definition of a consistency link can be used to develop an algorithm of a formal procedure that checks structural consistency. The following algorithm

describes a procedure that checks whether there is consistency between the work breakdown and the organisation breakdown structural correspondence links in terms of resource allocation:

Algorithm SCN-RA-1: {consistency-check work and organisation breakdown links for resource allocation}

```
FOR each <resource r in PERT/CPM model> DO
  FOR each <task t to which r is allocated in PERT/CPM model> DO
    IF <t is mapped to SD-Task T in SD-Model> AND
      <r is mapped to SD-Resource R in SD model> THEN
      IF NOT <R allocated to T in SD model> THEN
        <inconsistency: r, t, R, T>
        <Implement one possible solution:
          (i) re-map r to a SD-Resource allocated to T
          (ii) re-map a task t' where r is allocated, to T
          (iii) remove r from t
          (iv) allocate R to T >
        ENDIF
      ENDIF
    ENDFOR
  ENDFOR
```

Whenever an inconsistency occurs, the specific tasks and resources of both PERT/CPM and SD model are identified and a possible solution is proposed. This algorithm can be used to support and/or implement the SYDIM elementary activities (A4) and (A5) previously described (see section "SYDPIM operations: the elementary activities of model integration"). They are also ideal to implement a software routine to support a semi-automated process.

Formal specification

The formal specification of the structural consistency links of resource allocation (SC-RA) is presented in appendix D.1.

This type of structural link (i.e. consistency) is only composed by relationships that impose certain conditions. Although there is no data structure specified within the link, two operators are specified which check whether a certain resource is allocated to a certain task. These operators access the data structures of the two input parameters to the link: the PERT/CPM plan and the SD model.

Work dependency links (structural consistency)

These links check whether the structural correspondence links of work breakdown and the links of work dependency are consistent one another. The consistency required between the two types of links is implicit in the rules already proposed to specify the work dependency links. The following conditions must be verified for all dependencies in the PERT/CPM model:

-
- (1) If, according to the work dependency links of structural correspondence, the dependency d in the PERT/CPM model is mapped to the SD intra-task dependency D of SD-Task T in the SD model, and tp and ts are the predecessor and successor tasks of d respectively, then both tp and ts must be mapped to T according to the work breakdown structural correspondence links;
 - (2) If, according to the work dependency links of structural correspondence, the dependency d in the PERT/CPM model is mapped to the SD inter-task dependency D in the SD model, and tp and ts are the predecessor and successor tasks of d respectively, and Tp and Ts are the predecessor and successor SD-Tasks of D respectively, then according to the work breakdown structural links of correspondence tp must be mapped to Tp and ts must be mapped to Ts .
-

The first condition refers to the mapping of PERT/CPM dependencies to SD intra-tasks dependencies. The second condition refers to SD inter-task dependencies. As mentioned, these conditions are implicit in the rules proposed to specify the work dependency links. This way, when specific links are specified for the first time, the mapping established will ensure consistency with the work breakdown links. However, as the project plans in both models are changed, inconsistencies may emerge. These consistency links support the process of identifying them and finding a solution.

If an inconsistency of type (1) above occurs, there are various solutions that can be followed depending on why the consistency failed. The corrective actions may imply:

- (i) re-mapping the tasks tp and/or ts to the SD-Task T , thereby adjusting the work breakdown links;

- (ii) re-mapping the dependency d to another SD intra- or inter-task dependency D' , as appropriate, thereby adjusting the work dependency links.

If an inconsistency of type (2) occurs, there are also various solutions that can be followed. Again, the corrective actions may imply:

- (i) re-mapping the task tp to the SD-Task Tp and/or the task ts to the SD-Task Ts , thereby adjusting the work breakdown links;
- (ii) re-mapping the dependency d to another SD intra- or inter-task dependency D' as appropriate, thereby adjusting the work dependency links.

So, if the two types of links are not consistent, one of the links needs to be adjusted to the other. The user may also adopt other solutions like changing both links, removing PERT/CPM dependencies from the mapping, or even change/eliminate the dependencies in the models. Whatever the choice, the two conditions above must be verified.

Again, formal algorithms for the consistency-checking procedure can be proposed:

Algorithm SCN-WD-1: { consistency-check intra-task work dependency links with work breakdown links }

```
FOR each <intra-task dependency  $D$  of SD-Task  $T$  in SD model> DO
  FOR each <dependency  $d$  in PERT/CPM model mapped to  $D$ > DO
    IF NOT (<predecessor( $d$ ) mapped to  $T$ > AND
      <successor( $d$ ) mapped to  $T$ > ) THEN
      <inconsistency:  $d, D$ >
      <Implement one possible solution:
        (i) re-map predecessor( $d$ ) and/or successor( $d$ ) to  $T$ 
        (ii) re-map  $d$  to another SD dependency  $D'$ 
      >
    ENDIF
  ENDFOR
ENDFOR
```

Algorithm SCN-WD-2: {consistency-check inter-task work dependency links with work breakdown links }

```
FOR each <inter-task dependency D in SD model> DO
  FOR each <dependency d in PERT/CPM model mapped to D> DO
    IF NOT (<predecessor(d) mapped to predecessor(D)> AND
      <successor(d) mapped to successor(D)> ) THEN
      <inconsistency: d, D>
      <Implement one possible solution:
        (i) re-map predecessor(d) to predecessor(D)
            and/or successor(d) to successor(D)
        (ii) re-map d to another SD dependency D'
      </Implement one possible solution:
    ENDIF
  ENDFOR
ENDFOR
```

In both cases, if an inconsistency is detected the dependencies in both models are identified and the more direct and simpler solutions are proposed. These algorithms can be used to support and/or implement the SYDIM elementary activities (A4) and (A5) previously described (see section “SYDPIM operations: the elementary activities of model integration”). They are also ideal to implement automated or semi-automated procedures based on a software application.

Formal specification

The formal specification of the structural consistency links of work dependencies (SC-WD) is presented in appendix D.1. The relationships of this link impose the two conditions (1) and (2) defined above.

Summary

The structural links of the SYDPIM basic mode have been proposed. There are three links of structural correspondence which map the following elements respectively: work tasks, resources and work dependencies. There are two links which check the consistency of this structural correspondence: consistency between work and organisation breakdown links in terms of resource allocation, and consistency between work breakdown and work dependency links. These structural links are the basis to formally integrate the two models. They ensure that the two modes are representing the same reality in terms of the structure of the project work and of the project resources. The other aspect of model integration has to do with the *amounts* or *quantities* associated with each of these three entities. In order to

ensure that the two models represent the same reality in this dimension, data-links are required. These are now proposed for the SYDPIM basic mode.

Data links

Overview

If a certain scheduled completion date was input to the SD model, as part of a project plan, how is this date transferred to the PERT/CPM model? What variables need to be adjusted and with what values? If certain resources were re-allocated to certain tasks in the PERT/CPM model, as part of an alternative plan, how should this be reflected in or transferred to the SD model? What variables need to be adjusted and with what values? In general, how to be sure that the two models are calibrated with the right data so that they represent the same project reality (status and plan)? The SYDPIM data links address these fundamental questions of quantitative integration.

Data links establish relationships between the data entered in and produced by the two models. As mentioned previously, overall eight categories of data can be considered in each model based the three following criteria:

- (i) input / output,
- (ii) past / future
- (iii) planned / actual.

In each category, both models will contain several variables that store specific data. How do all these variables inter-relate? Several combinations between different types of variables can be considered. Are all of these relationships valid? What is their meaning and what kind of relationship will each combination generate? Establishing these data relationships between the two models is a complex issue. In order to cope with this problem, a classification framework for the data-links is proposed.

There are various factors affecting the specification of data links. In general, the data links that can be established will depend on the specific characteristics of the two models. Furthermore, the establishment of data links is primarily based on the

structural links. Another critical factor is also that many variables in the two models stand at different levels of aggregation. The consequence is that exchanging data or checking consistency may imply dis-aggregation. Can this be done? In what conditions and how?

Before proposing the specific links of SYDPIM basic model, the classification of data links and the dis-aggregation of data is discussed in more detail. The data-links are then related with the SYDPIM models' objects and their usefulness is discussed in the context of the SYDPIM process logic. An overview of the data-links proposed in SYDPIM basic mode is presented. A formal specification scheme is proposed and the data links are then specified.

Classification of data-links

The data handled by the models was classified into eight possible categories. Assuming that this data can be inter-related across all of these eight categories, 64 potential categories of relationships can be generated. Depending on the specific SYDPIM scenario, all or only some of these categories will be "populated" with specific data links. As it will be seen, in SYDPIM basic mode only a few of these categories will include links. However, it should be noted that in another more sophisticated SYDPIM scenario, further categories could be considered.

For each category there can be various types of relationships. In some cases data can be *exchanged* and in other cases data will just have to be *consistent*. This leads to two main types of data relationships identified previously: (1) data exchange and (2) data consistency. Data exchange cannot occur between output type of data.

Conceptually, 64 categories of data consistency links can be considered and 48 categories of data exchange links can also be considered. A total of 112 categories of data links! a generic classification scheme can help understanding, identifying and validating the possible data-links. Table 7.4 provides such a framework.

PERT/ CPM Model	SD Model							
	Past				Future			
	Planned		Actual		Planned		Actual	
	Input	Output	Input	Output	Input	Output	Input	Output
Past Planned								
Input	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI
Output	DEOI, DCOI	DCO	DEOI, DCOI	DCO	DEOI, DCOI	DCO	DEOI, DCOI	DCO
Actual								
Input	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI
Output	DEOI, DCOI	DCO	DEOI, DCOI	DCO	DEOI, DCOI	DCO	DEOI, DCOI	DCO
Future Planned								
Input	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI
Output	DEOI, DCOI	DCO	DEOI, DCOI	DCO	DEOI, DCOI	DCO	DEOI, DCOI	DCO
Actual								
Input	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI	DEI, DCI	DEOI, DCOI
Output	DEOI, DCOI	DCO	DEOI, DCOI	DCO	DEOI, DCOI	DCO	DEOI, DCOI	DCO

Table 7.4 – SYDPIM classification framework for data links

Each cell in the matrix above identifies one of the 64 generic categories of data relationships. Within each of these cells, the possible data-links are identified as follows:

- data exchange links (DE):
 - DEI – data exchange input-input link;
 - DEOI – data exchange output-input link;
- data consistency links (DC):
 - DCI – data consistency input-input link;
 - DCOI – data consistency output-input link;
 - DCO – data consistency output-output link.

There is another issue to consider: for the data exchange links (DE), there are two different situations: (a) moving data from the SD model to the PERT/CPM model and (b) moving data from the PERT/CPM model to the SD model. If this is considered to differentiate the DE links, then 96 categories of DE relationships can be generated, raising the overall number of potential types of data-links up to 160! In some cases, however, the formal relationship used to exchange data from one model to the other will also work in the opposite direction. For such bi-directional links the differentiation is not relevant. Whether a DE link is bi-directional or not, it

will depend on the specific formal relationship. The following notation will be used to identify unidirectional inks:

- data exchange from SD model to PERT/CPM model:
DEI.SD-PERT
DEOI.SD-PERT
- data exchange from PERT/CPM model to SD model:
DEI.PERT-SD
DEOI.PERT-SD

This differentiation is not considered explicitly in the matrix above but will be identified in the formal specification of each specific data-link.

The output-input consistency links can also be split into those that refer to the PERT/PCM model as the output and the ones where the output comes from the SD model. Where appropriate these will be identified as follows:

- DCOI.PERT-SD – the output from the PERT/PCM model consistent with the input from the SD model;
- DCOI.SD-PERT – the output from the SD model consistent with the input from the PERT/CPM model;

Again, this differentiation is not considered explicitly in the matrix above but will be specified in the formal definition of each specific data-link.

The complexity of this classification can also be increased if it is considered that some links may involve data from both past and future segments of the project in one of the models. For example, the expected completion date of a SD-Task may involve considering the schedules of PERT/CPM tasks of the past segment (already accomplished) and of the future segment (not started yet). In the matrix above, this situation is not considered explicitly. Fortunately, as it will be seen, this is not a major problem to understanding the nature of the links.

Dis-aggregation of data

The need to dis-aggregate data occurs mainly when data is transferred from the SD model to the PERT/CPM model.

Most variables representing data common to both models are defined at a higher level of aggregation in the SD model. This implies that transferring values from the SD model to the PERT/CPM model requires dis-aggregation. It also implies the need for a certain *criteria* of decomposition. In principle, this should be based upon the structural links. For example, if a certain SD-Task in the SD model has a planned budget of 100 person-day, then in order to transfer this value to the PERT/CPM model, the structural links should be used to identify which PERT/CPM tasks are mapped to that specific SD-Task and hence should receive *part* of that budget. The next step would be to distribute this budget among all the PERT/CPM tasks. This is where a formal decomposition criteria is required.

Dis-aggregation of data is a complex issue. A decomposition criteria can be determined in two ways:

- (1) deduced analytically from the two models and their structural links;
- (2) specified as a SYDPIM input by the user.

Unfortunately, the first possibility is not viable because there is no information available in the models and in their structural links that could lead to *logical deduction* of the *right* or a *valid* criteria. Back to the example, from the structural links we know that the 100 person-day should be distributed through PERT/CPM tasks “x”, “y” and “z”. But there is no other information available in the models that tells us how much should go to each of these three tasks. It is logical to wonder about what is needed to answer this question. Some breakthrough can be done: the profile of effort expenditure over-time of the SD-Task produced by the SD model can be used to deduce how much is spent in specific periods of time. So if the PERT/CPM tasks “x”, “y” and “z” are purely sequential, then the problem is resolved: the solution is analytically deduced and complies with the practical reality of the project. However, as far as there is some *overlap* between the tasks (i.e. tasks taking place simultaneously), there is again the problem of distributing that portion of the budget which is spent during those periods of time where work in more than one PERT/CPM task is underway. Fortunately, there is yet another breakthrough that can be considered: the breakdown of the effort expenditure among the *continuous activities* that take place within the SD-Task. In most SD project models, a few activities modelled as rates will characterise the work processes within a SD-Task.

It is therefore possible to know, for example, how the budget being spent is due to “development work”, due to “review work” and due to “rework”. Now, *if* the tasks in the PERT/CPM model are *classified* into categories *equivalent* to these types of SD-activities, the problem may again be solved: if the tasks that overlap in the PERT/CPM model are of different type, then from the breakdown of effort expenditure by activity within the SD-Tasks it is possible to know how much should go to each PERT/CPM task. For example, if in a certain month only the mapped PERT/CPM tasks “Design Interface” and “Review Interface Design” are underway, then the first task will receive the effort spent by the SD-activity “development” and the second the effort spent by the SD-activity “review”. This solution has a requirement: it imposes the important conditions that (1) the tasks of the PERT/CPM model are classified into work categories and (2) these categories are formally mapped to intra-task SD-activities. The disadvantage is that imposing a task classification scheme on the PERT/CPM model may not comply with many organisations’ practices and is clearly restrictive. The other limitation of this solution is that if two PERT/CPM tasks of the same type are underway simultaneously, something which is likely to happen, the problem of dis-aggregation persists once again. And this time, there is no further data available in the SD model to deduce how the budget is to be split. From these facts, it is reasonable to accept that a fully formal dis-aggregation algorithm is not viable and practical. The example above focused on decomposing a budget. The problem with decomposing the schedule can be even more complex.

The breakthroughs illustrated above are however valuable and will be used for the definition of consistency links. Under a “consistency perspective”, these links will not say “this % portion of the budget should go to task x”, but instead they will state “this budget should be distributed among these tasks in a way (decided by the user) that the resultant profile of effort expenditure over-time is the same as in the SD model. The “consistency approach” is formal in the conditions imposed, but it leaves the user with the freedom to decide about dis-aggregation in a way which corresponds to reality.

Let us now consider the second solution: the decomposition criteria is analytically specified by the user as an input to the SYDPIM process. Unfortunately, this does not help because of two reasons: first, formal decomposition rules are likely to be

hard to specify for the project manager, secondly even if they were defined, most likely they would not reflect what happens in the real world. Why? Because in the real world high-level budgets are decomposed down to the various PERT/CPM tasks according to various factors, most of which cannot be modelled formally. These factors include, for example, deciding on how to combine the specific staff to form teams, which will depend on the specific individuals available. Such process can hardly be modelled (and automated) through a formal rule. However, some breakthrough is possible and simplifications can be considered. For example, decomposing the budget in proportion to the duration of the PERT/CPM tasks mapped to the SD-Task. This type of simplifications also have their serious limitations: in this example one would be assuming the same level of cost per day in all tasks, which in reality would hardly be the case.

Decomposing aggregate data is not an original problem. Project managers have to do it all the time when they start decomposing the budget in approved bids to the project phases. The most formal way available to do this is to use past metrics as a *guidance*. This approach is used in empirical estimating tools like the COCOMO and the KnowledgePLAN packages in the software industry. From a database of past projects, it is observed, for example, that the design phase of this type of projects consumes 30% of the budget and so this factor is used to decompose the project budget. With all the reservations about extrapolation into future projects (and about the “scientific validity” of empirical regression analysis), this could be a reasonable solution for SYDPIM as well. In fact, the semi-formal integration of KnowledgePLAN with MS Project recently developed considers that high-level estimates are first produced using an empirical estimating engine and are then “automatically” decomposed down to the PERT/CPM tasks. This “automated” decomposition is based on two principles: (i) the user chooses up-front a pre-defined PERT/CPM network plan available within the tool, and (ii) because from hereon the tool will know which specific tasks are being mapped to the high-level phases, a database of past metrics is used to decide how much goes into which task. This approach imposes the use of a pre-defined PERT/CPM network plan, which in any case does not provide the operational detail required in practice. The further decomposition of this initial version of the PERT/CPM plan is then carried out “manually”, to consider all the specifics and uncertainties of the operational world within the project. This supports the idea advocated in SYDPIM that for this type of

problem, full automation needs to and should be compromised with human judgement.

In SYDIPM basic mode, no pre-defined network structure is considered for the PERT/CPM plan. This way, an automated decomposition process would be even more difficult to implement because the exact meaning of each PERT/CPM task in terms of work contents is not available. Furthermore, the possible use of empirical regression analysis to derive decomposition factors would require a metrics database with a meaningful number of sample projects. This is not available in most organisations (if in any at all), and even if it was, the process would not guarantee the “right” solution for this specific project and specific situation. In most cases, the result would always need to be “manually” readjusted by managers. Nevertheless, SYDPIM considers that this type of approach for automated dis-aggregation can be used, but that is an implementation issue left to be handled by the user.

Based on this discussion, in SYDIM basic mode it is assumed that work decomposition will not be automated in most cases. Automated decomposition will only be used in simple cases, as part of the overall process, and where the algorithm is logically valid.

Data links and the SYDPIM objects

Input and output data are stored in the models’ variables. The data links therefore establish relationships between these variables. In terms of implementation, the specific variables and their names will depend on the specific models used. To overcome this issue, the models were previously specified as abstract objects which contain the explicit identification of those variables that any model will have to have in order to be suitable for SYDPIM.

The structural links were previously specified as making reference to the sub-components of the objects that specify the models. Likewise, the data links will be formally specified as making reference to the variable names in the models’ objects, as suggested in the figure 7.13.

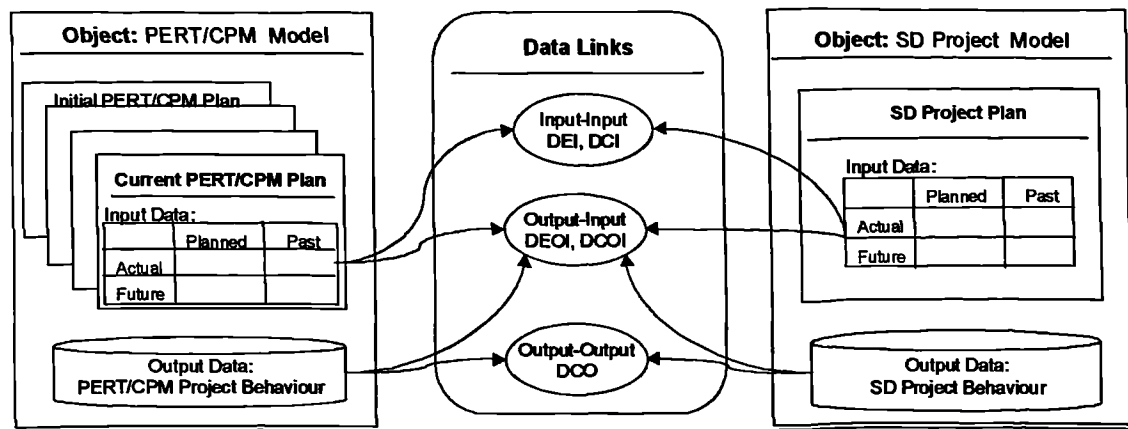


Figure 7.13 – The data links are specified through direct references to the variables in the models' objects

The data links and the SYDPIM process logic

As discussed above, overall 160 different types of data links can be established. It was also mentioned that many of these potential links are of no practical use. In order to understand the usefulness of the possible links it is important to consider their roles within the SYDPIM process. This will be better illustrated in the detailed description of the SYDPIM activities.

However, it is important at this stage to understand some of the relationship between the possible types of links and their role within the SYDPIM process. The data links will be used in three main situations:

- (1) when the models are set-up to represent the same initial project plan;
- (2) when the models are readjusted to represent the same new **current** plan for the remaining future of the project;
- (3) when the models are readjusted to represent the same **actual** project status up to present.

The following algorithm provides a simplified description of the SYDPIM process logic, where these three scenarios above are identified. For the sake of simplicity, it is assumed that the project plans are always developed first in the PERT/CPM model and are then tested and improved in the SD model:

Algorithm: {Simplified SYDPIM Project Control Process}

```
Develop initial plan in PERT/CPM model
Analyse initial plan in PERT/CPM model
Improve initial plan in PERT/CPM model
Transfer initial plan to SD model – scenario (1)
Analyse initial plan in SD model
Improve initial plan in SD model
Transfer initial plan to PERT/CPM model – scenario (1)
WHILE Project is not Finished DO
    Implement plan for the next control period
    Update PERT/CPM model with actuals regarding project status
    Analyse project future in PERT/CPM model
    Transfer project status to SD model – scenario (3)
    Analyse project future in SD model
    Develop new future plan in PERT/CPM model
    Transfer new future plan to SD model – scenario (2)
    Analyse future plan in SD model
    Improve future plan in SD model
    Transfer new future plan to PERT/CPM model – scenario (2)
ENDWHILE
```

One of the criteria used to classify the links was whether the type of data handled was planned or actual. It is now important to differentiate between planned data regarding the initial plan and planned data regarding the current (new) plan.

The initial plan is particularly important for the SD project model. This plan establishes the initial project targets which will affect the evolution of the simulated project. Ideally, in the SD model the project outcome would result from this initial plan and the endogenous management control actions. Without an initial plan, the SD model would not be able to simulate management decision-making throughout the project because there would be no targets. The initial plan is critical for the SD perspective of a project outcome. Because an initial plan is required by the SD model it was also considered that an initial plan would be available in the PERT/CPM model (as often is in reality).

Some of the data links to be established between the two models are aimed exclusively at transferring the project initial plan from one model to the other – scenario (1) above.

As the project is implemented new versions of the project plan are produced. A critical issue with the SD model is that once it has received the required input data to

represent the targets of the initial project plan, it never receives any further input data regarding the newly updated targets. Ideally, changes to the initial project targets should be generated endogenously within the model, with real-life management decisions being accurately simulated. In practice this is not so easy to achieve and changes to the targets may need to be “imposed” in the SD model. For this purpose, exogenous management decisions, which can occur at any stage in the project life-cycle, are considered as inputs to the SD model (see object specification). Under a validation perspective, exogenous inputs to the SD model are acceptable as far as the real causes of the events lay outside the SD model’s boundary; in these cases the SD model should not be “forced” to reproduce the desired outcome. So, if a new plan is produced by management in the PERT/PCM model then the new adjustments to the targets will not be an input to the SD model. Rather, they will be endogenously produced by the model as an output, eventually with the help of some exogenously decisions “imposed” as inputs. The same rationale applies to updating the SD model with the actual project outcome: the results are not imposed in the model as an input but rather they must be produced by the model as an output – again, exogenous management decisions may be required.

The consequence of all this is that because management actions which causes cannot be validly captured by the SD model may be required, exogenous management decisions will be required as inputs so that the SD model reproduces well both the project status and new project plan. Some of the data links to be established are aimed exclusively at generating exogenous decisions to the SD model. They will support the scenarios (2) and (3) above.

The data links that support the SYDPIM scenario (1) above handle data in both models that refer to the initial plan. It was assumed that both models will contain this plan – see object specification. On the other hand, the data links that support the SYDPIM scenarios (2) and (3) above handle data which in the SD model refer to the “SD plan” and which in the PERT/CPM model refer to the “current PERT/CPM plan”. Links used in scenario (2) handle mainly data in the future segment of this plan whereas the links used in scenario (3) handle data in the past segment of the plan.

In order to understand the usefulness of each data-link within the SYDPIM process it is important to identify which scenario the link is addressing. For the sake of clarity, this scenario will be identified in the specification of each link.

Summary overview of the data-links proposed

Overall 37 data-links are proposed in SYDPIM basic mode, as summarised in table 7.5. These links will be specified formally in the remaining of this section. Before that, it is useful to understand the overall picture.

The first column of table 7.5 identifies the three generic scenarios where the links are used. The second column identifies the specific “data item(s)” which will be transferred or consistency-checked across the two models. The third and fourth columns identify the data-exchange links that transfer the data-items, from the PERT/CPM model to the SD model and vice-versa. The fifth column identifies the data consistency links that check the consistency of the data-items between the two models. So, for each specific data-item there can be three types of links.

The fact that for each data-item there are both data-exchange and data-consistency links raises the question of what are the roles of each type of link. The critical issue in SYDPIM is that the process of adjusting one model to the other is a *semi-automated* process. It starts with an initial automated data-transfer, using the data-exchange links, and is then followed by an iterative manual process of model readjustment / re-calibration, guided by the consistency links. The consistency links are used as flags which identify the remaining inconsistencies between the two models. Once all consistency links are verified, the two models are representing the same project reality. It is important to note that throughout this iterative process, some consistency links may alternate between being verified and being violated – i.e. once link A becomes verified, further model changes to achieve verification of link B may lead to violation of link A. So the “flags” go up and down throughout the process until they are all eventually down.

Looking at table 7.5, the data-items considered are the data elements previously identified as being common to both models (see model assumptions underlying the object specification of the models). These data elements are as follows:

- profiles of resources availability for the whole project;
- project start date;
- start and finishing dates of SD-Tasks and project finishing date;
- budget / cost of SD-Tasks / project;
- scope of SD-Tasks / project;
- profiles of resource allocation to SD-Tasks / project.

Some of the data-items presented in table 7.5 consider specific occurrences of these data-elements – e.g. present level of resources allocated to a SD-Task. It can be seen that for all data-items there is one data-exchange link from the PERT/CPM model to the SD model, but the reverse is not always true. This highlights the problem of data dis-aggregation discussed above. Data-transfer links from the SD model to the PERT/CPM model refer only to data-items that stand at the same level of aggregation.

Another important issue is that, as it will be seen, the links that transfer the project status and the project future plan from the PERT/CPM model to the SD model will not generate the desired results or targets as inputs to the SD model. Instead, they will generate exogenous decisions which will contribute to the achievement of these results and targets.

For all data-items there is a data-consistency link. Therefore, all data that can be exchanged between the two models can also be checked for consistency. This is important given the semi-automated nature of the process described above.

The data links that refer to the budget breakdown will be specified, but cannot be implemented in SYDPIM basic mode – this will be clarified when these links are presented.

Transferring the project status from the SD model to the PERT/CPM model is not likely to be required in most cases, because according to the process logic of SYDPIM monitoring the project past is first updated in the PERT/CPM model and only after this it is transferred to the SD model.

Some data-exchange links are presented in both scenarios: transferring the project status and the project future plan. This happens for those links that adjust the project targets at the present moment. The present moment can be seen as belonging to both the project past and the project future segments. For example, after the past period has been updated in the PERT/CPM model, a new completion date is forecasted at the present moment for the project – this is the planned completion date if the future plan is not changed. This present planned completion date is transferred to the SD model as part of the project status. Still at the present moment, management produces a new PERT/CPM plan for the future and now the planned completion date may have changed. This new present planned date is now transferred to the SD model as part of the future segment. The same link is used for both cases.

The formal specification and detailed explanation of each of the data-links shown in table 7.5 is now presented. The actual use of the links will take place in the detailed description of the SYDPIM activities, but most of the underlying rationale will be described by then. The “formal” specification language to be used is proposed below.

Formal specification language

The formal specification of the data links will be implemented using the format below. The links will be grouped according to the five types identified above and according to the specific type of data being handled. A sequential number is used to distinguish between links of same type but which refer to different type of data. An informal definition of the links is provided followed by a brief description of its usage and by its formal specification, using the same type of language proposed for the objects. The links will be “called” (or implemented) within the SYDPIM activities. Their real names, based on the formal specification, tend to become very long and so short abbreviations are also proposed. Finally, the models’ variables referenced in the formal specification are identified using short abbreviations as well. These are mapped to their true long names in the object specification of the models.

<Link type>-n: <Link title description>

<Link type> : DEI | DEOI | DCI | | DCOI | DCO
n : serial integer number
<Link title description> : text description

Definition:

<Informal description of the links group and identification of the individual links>

Usage:

<informal identification of the scenario addressed: transfer initial plan, transfer project status or transfer current project plan>

Formal specification:

<Formal specification of the link using the same language used to specify the objects>

Short references:

<Definition of short references to each link used to "call" the link in the description of the SYDPIM activities >

Object variables:

<Mapping of the short-named models' variables used in the formal specification above, to the actual variables in the in the models' object >

DEI – data exchange input-input links

Overview

These links specify which input data can be exchanged between the two models. This type of data-transfer takes place when one model is being adjusted to reflect either the project plan or the actual project status in the other model.

This type of data exchange can happen in both directions, depending on from which model the project plan or project status is being converted. These two scenarios are analysed separately.

SD-input to PERT/CPM-input

This is a situation where dis-aggregation is likely to be required. Most input parameters that characterise the project in the SD model stand at a higher level of aggregation than in the PERT/CP model. From the previous discussion about dis-aggregation, for such input parameters only data-consistency checking is possible.

Therefore, only input parameters that stand at the same level of aggregation in the two models can potentially be transferred. In SYDPIM basic mode there are only two SD input variables in this situation:

- the variables that specify the initially planned profiles of *resources availability* over-time in the project – it was assumed that the models would consider this type of variables as an input, both for planned and actual profiles (see object specification);
- the variable that specifies the initial *date* of the project – it was assumed that in both models there would be a variable specifying the project's initial date (see object specification).

Regarding the first variable, in a PERT/CPM model these profiles generally aggregate resources into categories. For example, it is not too common to specify a profile of resource availability for the amount of "John's" available. Instead, a certain profile over-time of resource availability will specify the number of "Designers" available. These profiles are typically used in PERT/CPM tools to perform automated re-planning operations like *resource loading* and *resource levelling* (Nicholas 1990).

In this way, there can be variables in both models which specify profiles of resource availability at the same level of aggregation. This happens when a SD-Resource (which typically aggregates several individuals according to the type of work they perform), corresponds to a PERT/CPM resource category (which is used in the definition of the profiles of resource availability). In this case, both the SD-Resource and the PERT/CPM resource category stand at the same level of aggregation *and* incorporate the same elementary resources – an example of this match is the profile of resources availability which aggregates all the resources in the project. In these

cases, it is possible to establish a data exchange link that transfers the initially planned profiles of resource availability from one model to the other. This link is bi-directional since the transfer can also occur from the PERT/CPM model to the SD model. In SYDPIM basic model the DEI-1 link implements this formal relationship. The formal specification of this link is shown below (also in appendix D.2).

DEI-1: Initially Planned Profiles of Resources Availability

Definition:

For those profiles of resource availability defined in the PERT/CPM model at the same level of aggregation as in the SD model, and in both models these profiles also incorporate the same elementary PERT/CPM resources, then the following links can be established:

- transfer initially planned profile of resources availability from PERT/CPM to SD
- transfer initially planned profile of resources availability from SD to PERT/CPM

Usage: transfer initial project plan from one model to the other

Formal specification:

DEI-1(PERT/CPM Model, SD Model) =
 Relationships =
 SD_to_PERT =
 PERT/CPM.Init_Plan_Res_Avail[r_i][t] := SD.Init_Plan_Res_Avail[r_i][t]
 PERT_to_SD =
 SD.Init_Plan_Res_Avail[r_i][t] := PERT/CPM.Init_Plan_Res_Avail[r_i][t]

Where: r_i = SD-Resource[i]

Short references:

DEI-1.SD-PERT = DEI-1().Relationships.SD_to_PERT
 DEI-1.PERT-SD = DEI-1().Relationships.PERT_to_SD

Object variables:

PERT/CPM.Init_Plan_Res_Avail [r_i][t] =
PERT/CPM model.Initial plan.Plan_Resource_availability[r_i][t]
 SD.Init_Plan_Res_Avail [r_i][t] =
SD project model.SD plan.Initial plan.Resources.Project Resources[r_i][t]

The second SD input variable that can be transferred to the PERT/CPM model is the project initial date. The bi-directional DEI-2 link implements this relationship (see formal specification in appendix D.2).

It is important to note that these two links are only used the project initial plan is being transferred from one model to the other. In principle, this only happens once. After the models become consistent one another in representing the project initial plan, there will be no further need to transfer data regarding this plan and so these links will never be used again. Nevertheless, setting up one model to accurately represent the initial plan in the other model is one of the initial steps of the SYDPIM process and it is an important task.

PERT/CPM-input to SD-input

These links are used when a project plan or project status is being converted from the PERT/CPM model to the SD model. The links DEI-1 and DEI-2 already consider this type of transfer. The data inputs to the PERT/CPM model are directly transferred to the SD model. The potential inputs from the PERT/CPM model are the following:

- profiles of resources availability to the project (per resource category);
- project initial date
- schedules and duration per PERT/CPM task;
- budget / cost per PERT/CPM task;
- scope per PERT/CPM task;
- resource allocation per PERT/CPM task.

The first two inputs were already used to specify the bi-directional links DEI-1 and DEI-2 – the DEI-1 link considers the initially planned resource profiles. All the other inputs are defined at a lower level of aggregation and therefore cannot be *directly* transferred to the SD model. For example, the budget of a SD task will be equals to the sum of the budgets of the PERT/CPM tasks mapped to it (according to the structural links). However, this would not be a direct transfer because the sum of PERT/CPM budgets is not an input to the PERT/CPM model. Rather, it is an output of the model. So this data-relationship would be an output-input data exchange link (DEOI).

So, is there any other type of input-input data-transfer that can take place, other than the DE1-1 and DEI-2 links? As mentioned previously, when the project status

or project future plan are transferred from the PERT/CPM model to the SD model, it might be necessary to impose exogenous management decision in the SD model. These exogenous decisions are inputs to the SD model. If they can be derived from the inputs to the PERT/CPM model (which describe the project status and the project future plan), then a DEI links can be established. There is one link of this type considered in SYDPIM basic mode, which refers to planned profiles of resource availability.

The initially planned profiles of resource availability to the project are transferred between the models using the link DEI-1. By definition of “initially planned”, this refers to the profiles planned at the beginning of the project. However, the current PERT/CPM plan contains information about the actual profiles in the past segment and about the *currently planned* profiles for the future segment. Both are inputs to the PERT/CPM model and both may differ from the initially planned profiles – there are three types of profiles overall. If the current PERT/PCM plan is to be transferred to the SD model, then both the actual and the currently planned profiles will have to be reproduced accurately by the SD model. For that to happen, some exogenous decision may be required. Why? If both the actual profiles and the currently planned profiles (in the current PERT/CPM plan) differ from what was initially planned, then management has undertaken corrective re-planning decisions both in the past and in the present moment regarding the future. When this re-planning can only be partially reproduced endogenously by the SD model (at least in a valid manner), exogenous decisions are required. These exogenous decisions need to be imposed in the SD model as inputs. The DEI links that can be used to derive these decisions will consider the input from the PERT/CPM model as being the actual and currently planned profiles and the input to the SD model as being the exogenous decisions themselves (the movement of the data is of course from the PERT/CPM model to the SD model).

Let us consider first the transfer of the actual profiles in the past segment. The current PERT/CPM plan contains the actual profile of resources made available to the project. In order to represent this project status, the SD model must reproduce this profile in the past segment. As it will be seen, there is a DCOI link which will check this consistency. Ideally, the SD model should be re-calibrated to generate a past profile as close as possible to the target (while ensuring that the calibration is

“valid”). However, if after this re-calibration there are still significant gaps remaining at certain points in time, then an exogenous adjustment is appropriate to close these gaps. The first possible solution would be to simply consider that the exogenous adjustment would be equals the remaining gap over-time, throughout the past segment. Unfortunately, this is unlikely to be a good solution. Why? As the gap is eventually closed at time T by the exogenous adjustment at that time, there will be a new endogenous reaction for T+1 (different from the one if the new exogenous decision was not implemented at time T), and so the anticipated exogenous adjustment for T+1 might not longer generate the desired resource level – this is just a consequence of the SD model not being a linear model. Conceptually, several exogenous solutions are possible which, when combined with the endogenous reaction within the model, will generate the required profile. In practice, these solutions can be developed manually through a trial-and-error, iterative re-calibration process. There is one particular route which can be semi-automated through a DEI link.

First, let us consider the extent of the gap between the profile produced by the SD model and the one taken from the PERT/CPM model. In the most preferred SYDPIM scenario, the SD model is reproducing well the past segment up to the previous time-point, where the models were readjusted one another. In this case, only the present data-point eventually needs to be adjusted. In this case, if after careful re-calibration of the SD model, the endogenous solution does not produce a pattern close enough (the real cause possibly laying outside the model’s boundary), then an exogenous input solution can be generated to produce a fit for the present data point. In this case, this would consist of a single-point adjustment of the gap remaining. This is:

- exogenous adjustment of resource availability[present] = PERT/CPM resource availability level for present time-point – resource availability level produced by the SD model for the present time-point

Here, the exogenous adjustment is equals the gap between the present resource level as in the PERT/CPM current plan and the resource level being produced by the SD model. Adding this value to the resource level in the SD model eliminates the gap.

This relationship leads to the first DEI link. The exogenous adjustment is imposed on top of the endogenous decisions and will ensure that the profile of resources availability to the project will be reproduced accurately by the SD model for the present time-point. The use of this link is based on the assumption that the SD model is already reproducing well the actual profile for the remaining past segment of the project.

Let us now consider that the SD model was not updated regularly and is not reproducing well the profiles of resource availability in the past segment. There are two approaches to solve this problem. One is to follow an iterative process of combining exogenous decisions with the endogenous decision-making within the model – as mentioned previously, several exogenous solutions are possible. A structured way to implement this approach is to conduct a progressive calibration from the past to the present, eventually using a constant time-interval. Each time-point can be considered as a “local” present time-point. The link previously mentioned could then be used to ensure a good fit at each of these time-points, after a careful endogenous re-calibration. The overall exogenous solution achieved with this approach would consist of periodical exogenous adjustments “added on top” of the endogenous decisions. This approach is based on a “as endogenous as possible” solution. The other alternative approach is at the other extreme: an overall exogenous adjustment is generated, which when added on top of the initially planned profile will reproduce the actual profile accurately, thereby leaving no need for endogenous adjustments. The SD model would then be re-calibrated so that this exogenously imposed actual profile serves the needs of the project and hence, ideally, no endogenous adjustments would be generated within the model – this is a “as exogenous as possible” solution. In practice, some further re-calibration may be required. In order to implement this approach a data exchange link is required to derive the exogenous adjustment. This would consist of an over-time adjustment as follows:

- exogenous adjustment of resource availability[t] = PERT/CPM resource availability level for past segment – initially planned resource availability input to the SD model for the past segment

The exogenous adjustment is simply the over-time difference between the initially planned profile for the past segment and the actual resource availability level

occurred. Implementing this decision, adjusts the initially planned resource availability for what was really needed over the past segment.

In case only the present data point needs to be readjusted, then the first DEI link identified above solves the problem. In case, the whole or a large part of the past segment needs to be readjusted, then one of the two approached can be used. The first requires an iterative re-calibration process possibly using the first DEI link, whereas the second requires some re-calibration after the data-transfer. Which of the approaches is "better" is a matter of which one represents reality better. If the re-planning decision-making process of adjusting the resource availability level represented in the model is poor and captures little of what generated the decisions in the real world, then the second solution is a good approach. Otherwise, if the model structure can capture the real decisions endogenously, then the model should generate as much as possible the required adjustments endogenously. The best solution is of course the one that considers the nature of the changes one-by-one, evaluates whether their cause is captured within the model feedback structure, and then decides whether that particular change should be generated endogenously within the model or whether it should be imposed exogenously. So, the two DEI links proposed above may or not be used, and should be seen as tools within the SYDPIM process, as opposed to solutions. They help re-calibrating the SD model for the actual project outcome as represented in the past segment of the current PERT/CPM plan.

Let us now consider the transfer of the currently planned profiles regarding the future segment. The current PERT/CPM plan contains the profile of resource availability planned for the remaining future of the project. This profile may differ from the initially planned profile, or even from the profile re-planned in the previous corrective actions. This implies that new re-planning decisions were implemented for the future segment of the current PERT/CPM plan. For this new plan to be transferred to the SD model, it needs to reproduce the new profile of resource availability. Now, the SD model reproduces two profiles into the future: the initially planned profile (as planned in the beginning of the project) and the forecasted actual profile (the one the model suggests is likely to occur). It is the actual profile that will have to match the planned profile in the current PERT/CPM plan (note that in SYDPIM, the concept a SD simulation of a PERT/CPM plan implies a *steady*

behaviour where the targets are achieved). Like with the past profile, this actual future profile reproduced by the SD model results from the initially planned profile and from endogenous and exogenous decisions. Ideally the SD model would be able to reproduce endogenously the new re-planning decisions just as they were implemented by management in the current PERT/CPM plan. This way, a proper re-calibration would ensure the required fit. Once more, it is unlikely that the decision-making process structure within the SD model will capture all factors affecting these decisions. In particular, the re-planning decisions implemented in the current PERT/CPM plan imply the readjustment in the present moment of a whole profile over-time into the future. Most (if not all) SD project models will not consider such an over-time profile being dynamically and endogenously readjusted every time step, according to the needs of the project – in reality, this decision-making process is a fairly complex issue to model; to the author's experience, in real world models used in real projects this was often over-simplified. It is therefore acceptable that exogenous inputs are used to represent adjustment decisions, which lay outside the model's boundary. The approaches that can be followed are the same as the ones just discussed for the past segment: (i) progressive calibration based on endogenous decisions corrected at every time-point with an exogenous decision, or (ii) produce an overall exogenous decision and then re-calibrate to eliminate the need of endogenous decisions. While for the past segment, the second approach may imply the model's endogenous inability to reproduce actual scenarios, for the future segment the situation is different. Generating an exogenous decision to produce the currently planned profile of resource availability is somewhat similar to the exogenous decision "initially planned profile" input to the model at the beginning of the project. In a way, it is as if the project was going to start again and a pre-planned profile of resource availability into the future is made available to the model. Re-calibrating the SD model to eliminate the need for endogenous decisions results naturally from ensuring that the future segment is implemented successfully "as planned". In order to implement the first approach, the same DEI link mentioned above can be used. To implement the second approach a new link is required to produce the required exogenous decision:

- exogenous adjustment of resource availability[t] = current PERT/CPM plan resource availability for the future segment – (initially planned resource availability input to the SD model for the future segment + (actual resource

availability level produced by the SD model for the present time-point – initially planned resource availability input to the SD model for the present time-point))

This exogenous adjustment is basically the over-time difference between the currently planned resource availability in the future segment in the PERT/CPM model and the initially planned profile, added by the current gap between the initially planned level for the present moment and the actual level in the SD model. For example, if in the SD model at the present moment there are already 10 persons more than what was initially planned then, if no further staff adjustment takes place, the resource availability profile for the remaining future would be the initially planned profile plus these 10 persons. This “adjusted profile” is then subtracted from the currently planned profile in the PERT/CPM model, to generate the exogenous decision.

Overall a rationale was proposed for the specification of three DEI links. These links transfer input data from the PERT/CPM model to the SD model, as part of the process of adjusting the SD model to represent the current project plan in the PERT/CPM model – this process will be fully described in the description of the SYDPIM activities. In SYDPIM basic mode the DEI-3 link implements these relationships – see formal specification in appendix D.2.

In principle, this link will be used regularly within the SYDPIM process whenever the project status is updated in the PERT/CPM model and is then transferred to the SD model. It will also be used when a new project plan for the remaining future of the project is developed in the PERT/CPM model and is then transferred to the SD model.

DEOI – data exchange output-input links

Overview

These links specify the output data from one model that can be directly transferred to the inputs in the other model. Again, this will happen when a project plan or project status is being transferred from one model to the other. It can also happen in both directions.

PERT/CPM-output to SD-input

There are various inputs to the SD model that can be derived from the PERT/CPM model output. This transfer of data requires *aggregation*, which, unlike *dis-aggregation*, can be formally deduced from the structural links.

As mentioned previously there are two types of SD input parameters that can be updated automatically through the data links: (i) “initially planned” type of parameters, and (ii) exogenous re-planning decisions which adjust project targets. The parameters in (i) are derived from the initial PERT/CPM plan, while the parameters in (ii) are derived from the current PERT/CPM plan.

The SD input parameters in (i) are as follows:

- initially planned schedules of each SD-Task;
- initially planned budget of each SD-Task;
- initially planned profile over-time of resource allocation of each SD-Task;
- initially planned scope of each SD-Task.

If a budget breakdown is considered in the SD model, according to different types of work (e.g. QA, rework), and the PERT/CPM tasks are also classified according to the same criteria, then there is another SD input that can be derived from the PERT/CPM model:

- the initially planned budget breakdown of each SD-Task.

One data link can be established for each type of data. These five data-exchange links are not bi-directional, since they only work to transfer data from the PERT/CPM model to the SD model. They refer exclusively to “initially planned” type of data in the SD model.

The other type of links that can be established refers to exogenous decisions which correct the planned targets in the SD model at the present time-point. Except for the resource profiles, these targets refer exclusively to the expected future outcome of tasks which are currently underway or have not started yet. Regarding the resource allocation profiles, three links can be established, similarly to the DEI-3 links: one to

adjust exogenously the actual profiles, another to adjust the present resource level and another to adjust the future planned profile. Overall, the exogenous decisions that can be generated are as follows:

- adjustment of currently planned start date of SD-Tasks not started yet;
- adjustment of currently planned completion date of SD-Tasks no completed yet;
- adjustment of currently planned cost at completion of SD-Tasks not completed yet;
- adjustment of currently planned scope at completion of SD-Tasks not completed yet;
- adjustment of current level of resource allocation in the present moment (for each SD-Resource allocated to the task) of SD-Tasks underway;
- adjustment of actual resource allocation profile in the past segment of SD-Tasks completed or underway;;
- adjustment of currently planned resource allocation profile for the future (for each SD-Resource allocated to the task) of SD-Tasks underway.

Overall four data links can be established, one per data type. The link that refers to resource allocation will include three sub-links (the three last bullets). The two first links that refer to the schedules will be aggregated within a single link.

All these links need to be specified formally. Let us start with those that refer to initially planned type of data.

The initially planned scheduled start and completion dates of each SD-Task can be derived from the PERT/CPM model, according to the data-link DEOI-1 (see appendix D.2). The variables referenced in both models refer to the initial plan. This link includes a validity operator which ensures that the PERT/CPM tasks are mapped to the SD-Task according to the WBS link of structural correspondence.

It could be argued that the earlier start date and the later finish date of a set of PERT/CPM tasks are inputs to the PERT/CPM model rather than outputs. However, in many cases the *duration* of the tasks is the real input and the fields *start date* and *finishing date* will result from the tasks' duration and from the elementary finish-to-start dependencies among them. In this case, the dates will be an output from the model. In other cases, start dates and/or finishing dates are imposed as

inputs to the tasks and their duration becomes an output. Whatever the case, the calculations of the “minimum start date” and “maximum finishing date” are considered as operations that produce an output. Therefore, for the sake of simplicity, it is assumed that the derived start and finishing dates of a sub-network of tasks in the PERT/CPM model are an output from this model.

The initially planned budget of a SD task can be derived from the PERT/CPM model according to the DEOI-2 link (appendix D.2). Similarly, the initially planned scope of a SD task can be derived from the PERT/CPM model according to the DEOI-3 link (appendix D.2). Regarding the initially planned profiles of resource allocation overtime to a SD-Task, these can also be derived from the PERT/CPM model, for each SD-Resource type allocated to that task. This is specified in data link DEOI-4 (appendix D.2).

Finally, if a budget breakdown into work activities is considered in the SD model, then the initially planned breakdown can be derived from the PERT/CPM model, assuming that the tasks in the PERT/CPM model are also classified according to the same criteria. In this case, the DEOI-5 link can be established (see appendix D.2). *It is important to note that the object specification of the two models presented earlier does not consider the required assumptions for this link to be implemented: in the SD model object, the budget of a SD task is not decomposed into work activities and in the PERT/CPM model object the tasks are not classified by work type. This way, the second condition of the validity statement is in italics to highlight that this is not possible to implement with the current object specification of the PERT/CPM model – it assumes that the operator *Is_Type(<task>,<work activity>*) is available. In the same way, the mapping of the SD variable used in the link to the object variable is also in italics to highlight that such SD variable is not available – the object specified for the SD model decomposes the initial budget by SD-Task only and not by work activity (the second dimension). Nevertheless, this specification provides a guidance to the required changes that would have to be implemented in the models' object specification. It also illustrates how this link can be implemented.*

The links DEOI-1 to DEOI-5 specified above refer exclusively to transferring data regarding the project initial plan from the PERT/CPM model to the SD model. These

links will only be used when the SD model is being adjusted to represent the initial plan in the PERT/CPM model. This is only likely to happen once at the beginning of the SYDPIM project control process.

Let us now specify the links that refer to exogenous decisions imposed in the SD model.

The exogenous adjustment of the currently planned start and finishing dates of the SD-Tasks is specified in link DEOI-6 (appendix D.2). This link produces exogenous adjustments at the same time for both the currently planned start date and finishing date of SD-Tasks. It is important to note that validity conditions of this link ensure that the start date can only be adjusted for tasks not started yet and the finishing date can only be started for tasks not completed yet. Also, the PERT/CPM tasks from where the currently planned finishing date of a SD-Task is derived can either belong to the past or to the future segment. In most cases, the specific PERT/CPM task which imposes the completion date will be in the future segment, but if it is an an-going task then it will belong to the past segment. In this link it is also assumed that the “present” time-point belongs to the past segment – this assumption is generalised to the whole SYDPIM process logic and will also apply to future links.

The adjustment of the currently planned cost at completion of the SD-Tasks underway is specified in link DEOI-7 (appendix D.2). The validity condition of this link ensure that the SD-Task which CAC is being adjusted is not complete yet. Note that in most cases, the PERT/CPM task that imposed the completion date belongs to the future segment of the current PERT/CPM plan (hence not started yet), but in some cases it may belong to the past segment (i.e. underway). This is considered in the formula of the validity condition.

The adjustment of the currently planned scope at completion of the SD-Tasks underway is specified in link DEOI-8 (appendix D.2). This link is similar to the DEOI-7 link, with the currently planned scope at completion being adjusted for the present moment instead of the cost at completion.

The adjustment of the current resource allocation profiles is specified in the link DEOI-9 (appendix D.2). This link includes three sub-links which adjust, for each SD-

Task, (1) the actual profile, the (2) present resource level and (3) the future planned profile. The logic of the data transfer is based on the same rationale proposed for the DEI-3 links. There are only two differences: (i) the resource profiles refer to SD-Tasks instead of the whole project and (ii) the data transferred from the PERT/CPM model is output data instead of input data. The present moment is considered as belonging to the project past segment. The validity conditions impose that: (1) the link that adjusts the past actual profile is only applicable to SD-Tasks which have started (complete or underway), (2) the link that adjusts the resource level at the present moment applies only applies to SD-Tasks underway and (3) the link that adjusts the future planned profile only applies to SD-Tasks not complete yet (underway or not started). The exogenous decisions of (1) and (2) are implemented in the past segment and the decision in (3) is implemented in the future segment.

The links DEOI-6 to DEOI-8 refer exclusively to adjusting the SD model to reproduce at the present moment the currently planned targets in the PERT/CPM model. This will happen regularly throughout the SYDPIM process, whenever a new plan for the remaining future of the project is developed in the PERT/CPM model. The links DEOI-9.1 and DEOI-9.2 are also used to adjust the SD model to the currently planned targets in the PERT/CPM model and are used in the same situation. The DEOI-9.3 link is used to adjust the whole past segment of the SD model to the actual outcome updated in the PERT/CPM model, regarding the resource allocation profiles. This will only happen when the SD model was not updated (and used) with actual results during a considerable period of time and needs to be updated at the present moment.

SD-output to PERT/CPM-input

In SYDPIM basic mode the only SD output data that can be transferred to the PERT/CPM model as inputs refers to the profiles of resource availability of the whole project. Once more, all the other SD output-data (tasks' schedules, budgets / costs, scope and resources) is defined at a higher level of aggregation than the inputs of the PERT/CPM model. In order to be transferred to PERT/CPM model, dis-aggregation would be required. As discussed previously, a formal data link can only be established if a formal dis-aggregation rule was available (either derived from the structural links and from other data in the models, or from user input). As

mentioned previously, based on practical experience, the use of such automated dis-aggregation rules is not likely to be accurate enough and is not considered in SYDPIM basic mode.

The actual and currently planned profiles of resource availability in the project are an output of the SD model. These profiles result from the initially planned profiles (an input to the model) and from the various endogenous and exogenous adjustments throughout the project simulation. In case the project status is first updated in the SD model or a new plan is generated in the SD model, new actual or currently planned profiles can be generated. If the PERT/CPM model is then to be re-adjusted to the SD model, these profiles can be transferred directly throughout a DEOI link. This is specified in link DEOI-10 (appendix D.2). To an extent, the DEOI-10 sub-links are the opposite of the DEI-3 sub-links – they transfer the actual and currently planned profiles of resources availability in the opposite direction. The link DEOI-10.2 is unlikely to be used regularly throughout the SYDPIM process, because according to the SYDPIM monitoring process logic the first model to be updated with actual results is the PERT/CPM model. Therefore, the actual profile of resources availability is first updated in the PERT/CPM model and only after that is transferred to the SD model, eventually using the DEI-3.3 link. On the other hand, the DEOI-10.1 link is used whenever a new plan is generated in the SD model and is then transferred to the PERT/CPM model. This will happen regularly throughout the SYDPIM process.

DCI – data consistency input-input links

Overview

Within the SYDPIM process logic, data consistency links are primarily used to check whether the project plan or project status represented in the two models are the same – i.e. consistent one another. If they are not the same, these links will identify what are the inconsistencies.

Input-input data consistency links impose conditions between the input data of both models. This differs from data exchange links because these conditions do not necessarily specify how the input data from one model is derived from the input data

of the other model. Only when a data consistency link imposes an equality between two variables, on from each model, an “equivalent” data exchange link can be derived from transforming this equality condition into an attribution. This way, data exchange links can be seen as deriving from a subset of the data consistency links which impose an equality condition between input parameters of the two models.

In concept, a data-consistency link is specified through the definition of a mathematical condition involving variables of the two models. These conditions can be more or less complex, depending on the specific input parameters considered in the models. For example, consider that the input parameter “maximum degree of concurrency” is used in the PERT/CPM model to measure the maximum number of tasks allowed to be implemented in parallel. This parameter would probably impose restrictions to the shapes of the intra- and inter-task work dependencies among the tasks in the SD model. In this situation, the shapes of these curves could not be *directly* derived from this PERT/CPM parameter (so no data-transfer link could be established), but their shape would be restrained, somehow, by this PERT/CPM input parameter. This would be an input-input data-consistency link.

In SYDPIM basic mode, there are two main situations where the comparison between variables is required: single values and over-time profiles. For the first case, a precise equality will be imposed and the operator “==” will be used for this purpose. For comparing profiles over-time, a precise equality is not the best approach. This is because these profiles may have different granularities (i.e. no. of data-points). Furthermore, the profiles produced by the SD model tend to be continuous whereas the ones produced by the PERT/CPM model tend to be more discrete. Therefore, the SD model tends to generate smoothed patterns where the PERT/CPM model generates more stepped type of patterns. To cope with this type of problem, in System Dynamics, “goodness-of-fit” criteria are often used (Sterman 1984). The operator “~=” will be used for this purpose, meaning that a “goodness-of-fit” criteria is satisfied. The formal specification of this criteria will be discussed later in this chapter.

Links

In SYDPIM basic mode, the only input-input consistency links considered are the ones that refer to the equalities underlying the DEI-1 and DEI-2 links previously specified. These are specified as links DCI-1 and DCI-2 (see appendix D.2).

The role of the DCI-1 link is to verify whether the initially planned profiles of resources availability to the project are the same in the two models. The role of the DCI-2 link is to verify whether the actual or planned project start date is also the same in the two models. It is important to note that in both cases this type of data is transferred from one model to the other only once, when the receptor model is being adjusted for the first time to the initial project plan in the other model. Once the two models become consistent for the initial plan, this consistency will never be affected again.

However, it can also be argued that if the initial project plan has just been transferred from one model to the other using the exchange links DEI-1 and DEI-2, then the DCI-1 and DCI-2 links will automatically be verified. So, what is the usefulness of these links?

The answer to this question is simple, but important to understand the seem-automated nature of the SYDPIM integration. Transferring data from one model to the other using the data exchange links is only *part* of the process of transferring a project plan or project status. This data transfer is not sufficient to ensure that the two models will be representing the same reality. In most cases, further “manual” adjustments to the receptor model are required.

Once the data is transferred, the data consistency links are used to check what remains inconsistent between the two models – see SYDPIM operations (A2), (A3) and (A4) previously described (see section “SYDPIM operations: the elementary activities of model integration”). If the DEI-1 and DEI-3 links were used, then at this stage in the process the DCI-1 and DCI-2 links will be verified. However, other consistency links might not be verified (in particular the output-output consistency), and so manual changes need to be carried out in the receptor model (e.g. recalibrating the SD model, or readjusting the PERT/CPM network). These manual

changes are not done all at once. Rather, this will be an iterative process. As the initial changes cause some consistency links to become verified, other links previously consistent may now become violated (including the DCI-1 and DCI-2 links). For example, if more resources are added to ensure budget consistency, as a consequence the profile of resources availability may become inconsistent. The user will need the DCI-1 link to know about this. In this way, the DCI links are necessary within the iterative process of manual adjustment to ensure that in the end of the process the models are fully consistent one another. These consistency links will work as flags which identify possible inconsistencies still remaining and that need to be resolved. The use of the consistency links in this way will be illustrated in the detailed description of the individual activities of the SYDPIM process logic.

Unlike the DEI-1 and DEI-2, the DEI-3 link does not originate a DCI link. Why? The aim of the DEI-3 link is to transfer data from the PERT/PCM model to the SD model so that the later reproduces the same profiles of resource availability. However, this "transfer" consists in generating exogenous input decisions to the SD model so that the profiles it produces as outputs are as required. So, the transfer does not consist of an equality between input variables with the same meaning in the two models. Furthermore, the exogenous decisions generated are not the only solution to make the SD model reproduce the desired profiles and the use of the DEI-3 links is not mandatory. Therefore, it does not make sense to impose any type of condition between the input exogenous decisions in the SD model and the input profiles in the PERT/CPM model.

DCOI – data consistency output-input links

Overview

These data consistency links are also used to check whether the two models are representing the same plan or project status. According to these links, the output from one model must be consistent with the input from the other model. Some of these links can be generated directly from the DEOI links where equalities are imposed, replacing attributions between similar variables. As described for the DCI links, these links are used to support the process of adjusting one model to the other eventually just after the DEOI links have been used. Their usefulness is twofold: (1)

identify what is different between the project plans or project status represented in the two models, and (2) work as “flags” to guide the readjustment of one model to the project plan or project status of the other model.

More or less complex conditions can be considered depending on the specific input parameters and output data produced by the two models. In SYDPIM basic mode, most of the DCOI links considered are the ones that impose the equalities underlying the links DEOI-1 to DEOI-5 and DEOI-10 previously specified. The links DEOI-6 to DEOI-9 produce input exogenous decisions to the SD model and, as discussed for the DCI links, they do not originate a DCOI link.

The DCOI links of SYDPIM basic mode are specified in appendix D.2 – see links DCOI-1 to DCOI-5. Let us consider first the links which output comes from the PERT/PCM model.

PERT/CPM-output to SD-input

As discussed for the DEOI-5 link, the DCOI-5 links requires the explicit breakdown of the effort within the SD tasks into work activities. It also requires that the tasks in the PERT/CPM model are classified according to the same criteria. These assumptions have not been considered in the proposed object specification of the two models for SYDPIM basic mode.

All the five links consider that the output is from the PERT/CPM model and so, like the DEOI links, they are all PERT-SD links. They also refer exclusively to “initially planned” type of variables in the two models and so their role is to check whether the initial project plans (developed at the beginning of the project) are represented in the both models in a consistent manner. In principle, this will happen only once when the models are set-up to represent the same initial project plan developed in the beginning of the project. So these links will not be used regularly throughout the SYDPIM process.

SD-output to PERT/CPM-input

The only PERT/CPM input data that can be checked for direct consistency against SD output data refers to the profiles of resources availability for the whole project. Two outputs are produced by the SD model: the actual profiles as occurred and the currently planned profiles for the remaining future. These profiles are inputs to the current project plan in the PERT/CPM model. The DCOI link that checks this consistency can be considered as originated from the DEOI-10 link previously specified. This data exchange link transfers the profiles produced by the SD model to the inputs of the PERT/CPM model. This is implemented in link DCOI-6 (see appendix D.2). This data consistency link is used whenever the models are readjusted so that they represent the same project past as occurred and the same current future plan. This will happen whenever a new future plan is developed in one of the models and needs to be transferred to the other model and also when the project actual outcome is updated in one model and needs to be transferred to the other model. This will happen regularly throughout the SYDPIM process.

DCO – data consistency output-output links

Overview

These links impose conditions between the output data produced by the two models. They are important indicators of whether the two models are representing the same reality. Their underlying concept is based on an intuitive relationship between the two models: if both are representing the same project, then the results that they produce must be the same (e.g. final duration and final cost) (Rodrigues 1994a). For example, the duration of the critical path of the current network plan in the PERT/CPM model must be the same as the value of the variable “scheduled completion date” in the SD model (at the end of the project simulation).

The output data considered in the DCO links includes both currently planned and actual data. This implies that for the models to represent the same reality, then both the current work plan and the results of implementing such plan must be the same.

The DCO links are used in the SYDPIM planning and monitoring activities. In monitoring they are used in activity (M6), where the SD model is re-calibrated to represent the project status as described in the PERT/CPM model. In planning, these links are used in activity (P3a), where the PERT/CPM model is readjusted to represent the plan of the SD model, and in activity (P4b) where the SD model is re-calibrated to represent the project plan of the PERT/CPM model.

Links

Output-output data consistency checking can only be implemented considering output data which is common to both models. In SYDPIM basic mode, the output data considered for this purpose refers to profiles over-time. In the PERT/CPM model, these profiles are the ones specified in the objects "PERT/CPM past behaviour" and "PERT/CPM future behaviour". In the SD model, these profiles are the ones specified in its sub-component: "Project behaviour". For the DCO links the following profiles of output data are considered in SYDPIM basic mode:

-
- (1) Schedules
 - (1.1) Whole project
 - (1.1.1) Finish date (SAC)
 - (1.2) Per SD-Task
 - (1.2.1) Start date
 - (1.2.2) Finish date (SAC)
 - (2) Effort – whole project and per SD-Task
 - (2.1) ACWP – actual cost of work performed
 - (2.1.1) engineering effort
 - (2.1.2) management effort
 - (2.2) BCWP – budgeted cost of work performed
 - (2.3) BCWS – budgeted cost of work scheduled
 - (2.4) CTC – cost to complete
 - (2.5) CAC – cost at completion
 - (3) Resources (allocation) – whole project and per SD-Resource category
 - (3.1) ASP – actual staff profile
 - (3.2) PSP – initially planned staff profile
 - (3.3) CASP – cumulative actual staff profile
 - (3.4) CPSP – cumulative planned staff profile
 - (3.5) CSPAC – cumulative staff profile at completion
 - (4) Scope – whole project and per SD-Task
 - (4.1) SCAC – scope at completion
 - (4.2) CSCC – cumulative scope changes
 - (4.3) ASCWP – actual scope of work performed
 - (4.4) SCTC – scope to complete
-

Output profiles of over-time data considered in SYDPIM for the DCO links

The outputs from the PERT/CPM model are not a direct output from a specific network plan taken from the model (e.g. initial or current plan). Instead, these outputs are taken from the objects “PERT/CPM past behaviour” and “PERT/CPM future behaviour” and in some cases result from data taken from a set of PERT/CPM plans stored in the model – the SAC[t] is a good example. So, how are these PERT/CPM output patterns generated? The algorithm will be described within the SYDPIM monitoring and planning activities where these patterns are extracted from the PERT/CPM model.

For each of these outputs, a DCO link is established – a total of 19 links. These are grouped into four main links, according to the type of data they handle (schedule, effort, resources and scope, as numbered above). Each link checks the consistency of all output variables within the group. These links are specified in appendix D.2 – see links DCO-1 to DCO-4.

Within each link various patterns are considered. For example, regarding the effort patterns the following indices were considered: ACWP, BCWS, BCWP, CTC, and CAC. In principle, one separate sub-link could be considered for each individual pattern. However, as it will be seen, for the purpose of the SYDPIM process there is no need to consider such detailed breakdown. As specified above, whenever one of the four DCO links is implemented all patterns within the link will be checked at the same time. The only relevant breakdown considered is the split between past and future segment. This is because the consistency of the patterns is checked at different stages within the SYDPIM process regarding the past and the future segments.

The DCO links are based on the fundamental principle that the dynamic patterns of a project's behaviour typically produced by a SD project model are also implicit in a PERT/CPM model and can be extracted from it. If the two models are representing the same project reality, then these patterns must be the same. The DCO links are very important to support the implementation of the whole SYDPIM process and will be used regularly throughout this process.

Data-structural links

Overview

As the project is simulated in the SD model and the project schedule eventually slips by, wouldn't the project manager want to know exactly how the PERT/CPM network will evolve? – i.e. a trip into to the future at the operational level. As the project manager readjusts the project plan in the SD model so that the targets achieved, wouldn't the project manager like to have immediately available the correspondent PERT/CPM network plan? (a considerable amount of effort would be saved). This is the ideal scenario for integrating the two models. Data-structural links would be required to provide the project manager with these features. In concept, the usefulness of these links is of great magnitude. However, in practice they are not so easy to implement.

Data-structural links establish relationships between the structure of one model and the data in the other model. Similarly to the data links, there are two types of relationships that can be considered:

- (1) *consistency* – if the data in one model has certain values, then the structure in the other model will have to have certain characteristics, and vice-versa.
- (2) *adjustment or generation* – the structure of one model generates data for the other model, or that the data in one model can be used to generate structural changes to the other model.

The concept of data-structural links results, in great part, from the models standing at different levels of aggregation. Structural aggregation often implies converting structure into data. Therefore, it can be expected that some of the data in the SD model corresponds to structural characteristics of the PERT/CPM model.

An example of this concept is the forecasted completion date produced by the SD model (of the whole project or of a SD-Task). Typically, this consists of an output variable, which evolves throughout the simulation. A varying pattern implies the implementation of re-planning decisions or the occurrence of risks, both of which will change the expected completion date. At the PERT/CPM level this will most likely correspond to structural changes in the logical network of the project plan, with tasks

and dependencies being added or removed. More generally, the simulation of the project implementation in the SD model, represented through changes in data (i.e. over-time patterns), corresponds in the PERT/CPM model to a “network evolution” which is characterised by on-going structural changes. This way, data in one model represents, more or less explicitly, structure in the other model. In SYDPIM, the data-structural links address these relationships.

It is important to understand the difference between data-structural links and data-links which are based upon structural links. In the data-structural links, the data generated from the structure of one model is based *only* on its structure. Most likely this data will refer to structural issues (e.g. no. of tasks running in parallel). Some data-links make use of the structural links and this may look like that the structure of one model is generating data for the other model. For example, the budget of a SD-Task being equal to the sum of the budgets of the PERT/CPM tasks mapped to it requires the SC-WBS link. However, the relationship being established is between data, and the structural link is only used to support the specification of that relationship. An example of a data-structural link is a link that counts the number of dependencies in the PERT/CPM model and generates an index of overall parallelism. This index would be a data-input (or output) to the SD model.

The formal specification of data-structural links also requires a clear definition of what is the structure of each model. At a detailed level of specification, the structure of the models can be considered as data. For example, a PERT/CPM dependency can be seen as a set of two numbers, each corresponding to the numerical identifiers of the predecessor and successor tasks. However, the perspective in SYDPIM is that the structure of a model is a collection of inter-related elements (or objects) as follows:

- (1) *structure of the SD model* – a collection of SD-Tasks, inter- and intra-tasks dependencies, and resources;
- (2) *structure of the PERT/CPM model* – a collection of PERT/CPM tasks, tasks dependencies, and resources.

The data-structural links of SYDPIM basic mode are based on these assumptions.

Application in the SYDPIM process

The usefulness of these links is to support the *automation* of two crucial SYDPIM operations, or elementary activities, which are required several times throughout the SYDPIM process logic. These were previously described:

(A4)*structural-data consistency checking* – the structure of one model must respect a certain condition depending on the input and/or output data in the other model. Data-structural links can be used to specify these formal conditions.

(A5)*structural readjustment* – the structure of one model and/or the structural links between the two models are readjusted so that both structural and data consistency are achieved. This operation assumes that, prior to being implemented, the two models are not consistent in terms of structure and/or data. Data-structural links can be used to specify formally how the readjustment takes place based on the data values.

In practice, the activity (A5) of structural readjustment is the more critical one. Throughout the SYDPIM process, the PERT/CPM model will have to be readjusted frequently to represent new project plans developed in the SD model. In the specification of the structural consistency links some algorithms were proposed to support the automation of this activity. For the data-structural links similar algorithms can also be proposed as part of the specification of generation / readjustment links. These will be based on data-structural consistency links.

Links of SYDPIM basic mode

The specific data-structural links that can be established between the two models depend on the model's characteristics: structure and data. For SYDPIM basic mode, this consists of the object specification previously proposed for the two models.

The first issue to consider is that no specific structural type of data (e.g. no of tasks running parallel, average number of predecessors) is considered explicitly in any of the models. The presence of this type of data in the SD model would raise the possibility of links being established with the structure of the PERT/CPM model. For that to happen, the SD model would have to contain a numerical description about

the structure of the PERT/CPM network. This could be a good direction to take in order to better integrate the two models, but it requires a specific orientation in developing the SD project model. The SD model would have to be “PERT/CPM oriented”, explicitly trying to capture and quantify the structural evolution of the PERT/CPM network. Such a specialised SD project model is not considered in SYDPIM basic mode. The other possibility would be the PERT/CPM model containing data that refers to the structural characteristics of the SD model. This is unlikely to be very useful for two main reasons: (i) most of the PERT/CPM data stands at a lower level of aggregation hence it will not aggregate structural characteristics of the SD model, and (ii) the standard use of a PERT/CPM model does not contemplate the presence of a SD model within the project control process; this would require a SD-specialised PERT/CPM model and could also be a direction to take in improving the PERT/CPM models for a better integration. Again, the PERT/CPM model considered in SYDPIM basic mode is “standard” and so data-structural links of this type cannot be established.

Developing specialised SD and PERT/CPM models, closer to each other, each containing data referring explicitly to the structure of the other model, is a good direction towards the development of more sophisticated SYDPIM modes of implementation. Since this type of data is not considered explicitly in any of the models of SYDPIM basic mode, is there any possibility to establish data-structural links? The possible will have to be based on “unintended” relationships between the structure and the data of the “standard” models. These relationships would result from the simple fact that both models are representing common aspects of the project.

There are only two data-structural links considered in SYDPIM basic model. These are based on the following relationships:

- the shapes of the intra-task dependencies in the SD model can be derived from output data of the PERT/CPM model;
- the shapes of the inter-task dependencies in the SD model can be derived from output data of the PERT/CPM model;

The SD-dependencies themselves are a structural element (or object) of the SD model. These elements contain data within which specifies the dynamic shape

of the dependency. The perspective assumed in SYDPIM is that this data describes structure. The data within the SD dependencies describes a “progress curve”. For the inter-task dependencies this curve specifies how much work can be accomplished in the successor task given the amount of work already accomplished in the predecessor task. For the intra-task dependency, this curve describes how much work can be started given the amount of work already accomplished within the task. Changing this data is a structural change in the SD model. Therefore, deriving this data from PERT/CPM output data is a data-structural link.

The “progress curves” of the SD dependencies are considered in SYDPIM as being part of the project plan. This is because the “progress curves”, which are typically related with product development issues, impose a dependency which may or not be respected as the work is carried out within the task. For example, in certain conditions extreme schedule pressure, more work can be started within the SD-Task than what the dependency would allow. While doing so is likely to have negative impacts on quality (the specific impacts considered will depend on the specific SD model at hand), it can happen.

The links here proposed are used to transfer the project initial plan or the new current plan from the PERT/CPM model to the SD model. Therefore, the progress curves of the SD-dependencies are derived only for the initial plan or for the current plan in the SD model.

The “progress curves” are derived from the shape of the cumulative scope planned to be accomplished in the SD-Task, according to the project plan in the PERT/CPM model. This shape is an output from this model, which can be obtained from the object “PERT/CPM future behaviour” – ASCWP[t] variable. It can be obtained for both past and future segment of the project. When the progress curves are derived for the new current plan transferred to the SD model, the “PERT/CPM future behaviour” is used as the source. When the progress curves are derived for the initial plan, the “PERT/CPM future behaviour” will also be the source, assuming that the project has not started yet (hence the future behaviour covers the full project life-cycle). If the project has already started and the SD model was not updated at day zero with the initial PERT/CPM plan, then at the present moment the SD model will have to be calibrated for the past and future segments. Regarding the past segment,

the behaviour to be produced by the SD model refers to the past and current PERT/CPM plan. However, regarding the future segment, the future behaviour to be produced must correspond to the initial PERT/CPM plan. Otherwise, the initially planned progress curves for the project past segment cannot be derived, and hence this link cannot be implemented.

The final question is: how to derive the shape of an intra- and inter-task SD-dependency “progress curve” from the PERT/CPM model? Regarding the intra-task dependency, the key assumption is that the planned cumulative scope accomplishment of a SD-Task, as extracted from the PERT/CPM plan, reflects the shape of the intra-task “progress curve”. This is because typically project plans are compressed “as much as possible” to cope with tight deadlines. This “as much as possible” generally reflects how much the project manager is willing to compress the work within a task, up to the limit of technical feasibility. However, in some cases the project manager can feel so pressured that even further compression is carried out for the remaining future of the project. In other cases, the project manager may have free time to add as contingency to the tasks. The algorithms here proposed for these links are based on the assumption that the limit of the technical feasibility is being considered in the PERT/CPM project plan. If that is not the case for a specific SD-Task, then the “progress curve” produced by the algorithm can be adjusted manually, being used as a good starting point.

The following algorithm derives the “progress curve” for the intra-task dependency of a SD-Task. This algorithm assumes that there is an average delay of work accomplishment within the task – input variable DELAY. This represents how long, in average, it takes for an elementary work task to be accomplished once started. This delay encapsulates several characteristics of the product development process modelled within the SD-Task.

Algorithm DSR-1-1: {derivation of progress curve for intra-task dependency}

Input parameters:

UT := <time unit assumed in the SD simulation>
DELAY := <expected average delay of work accomplishment within the task>
ASCWP[t] := <planned cumulative scope accomplishment from PERT/CPM>
Planned SCAC[t] := <planned scope at completion from PERT/CPM>

Output parameters:

Progress_Curve[x%] ==
<% scope that can be started given x% already accomplished>

ASCWP%[t] := ASCWP[t] / Planned SCAC[0]
Scope_That_Can_Start[t] :=
 IF ASCWP%[t] == 0 THEN ASCWP%[DELAY * UT]
 ELSE ASCWP%[t + DELAY * UT]
LET x% : ASCWP%[T] == x%
Progress_Curve[x%] := Scope_That_Can_Start[T]

The input parameters ASCWP[t] and “Planned SCAC[t]” are taken as output from the PERT/CPM model. This output is produced when the PERT/CPM planned behaviour is extracted from the PERT/CPM model (in the appropriate activity of the SYDPIM process logic). In this process, the pattern ASCWP[t] will be generated considering the PERT/CPM dependencies mapped to the SD-Task.

The SYDPIM analytical link DSR-1 derives the SD-intra-task dependencies from the PERT/CPM future behaviour. This is specified in appendix D.3.

The algorithm used for the inter-task dependencies between two SD-Tasks is similar. For the same reasons, it is also assumed that the tasks are being overlapped up to the limit of technical feasibility. The “progress” curve produced by the algorithm can be adjusted manually if appropriate. The algorithm is as follows:

Algorithm DSR-2-1: {derivation of progress curve for inter-task dependency}

Input parameters:

UT := <time unit assumed in the SD simulation>
DELAY := <average delay of work accomplishment in successor task>
ASCWP_P[t] := <planned cumulative scope of predecessor task >
ASCWP_S[t] := <planned cumulative scope of successor task >
Planned SCAC_P[t] := <planned scope at completion of predecessor task>
Planned SCAC_S[t] := <planned scope at completion of successor task>

Output parameters:

Progress_Curve[x%] ==
<% scope that can be started in successor task given
x% of scope already accomplished in predecessor task>

ASCWP_P%[t] := ASCWP_P[t] / Planned SCAC_P[0]
ASCWP_S%[t] := ASCWP_S[t] / Planned SCAC_S[0]
Scope_That_Can_Start[t] :=
IF ASCWP_P%[t] == 0 THEN 0
ELSE ASCWP_S%[t + DELAY * UT]
LET x% : ASCWP_P%[T] == x%
Progress_Curve[x%] := Scope_That_Can_Start[T]

Again, the patterns ASCWP_P[t] and ASCWP_S[t] will be generated when the PERT/CPM planned behaviour is extracted from the PERT/CPM model in the appropriate activity of the SYDPIM process logic. The pattern ASCWP_S[t] will consider only the PERT/CPM dependencies mapped to the inter-task SD-Dependency.

The SYDPIM analytical link DSR-2 derives the SD-inter-task dependencies from the PERT/CPM future behaviour. This is specified in appendix D.3.

These two data-structural links are links of structural readjustment, where the structure of the SD model is readjusted based on output data from the PERT/CPM model. Similar links can be established to check the consistency of these relationships. The SYDPIM links DSC-1 and DSC-2 implemented this process, and are formally specified in appendix D.3. These links are used when a project plan is transferred from the PERT/CPM model to the SD model.

Other data-structural links that would be very useful for the automation of the SYDPIM process would readjust the structure of the PERT/CPM model based on

data from the SD model. Such links would be used to transfer new project plans developed in the SD model. As suggested by SYDPIM, this is the preferred way to develop new plans. Developing new project plans in the SD model requires less effort (less amount and detail of input data) and provides more reliable plans (as they are developed their outcome is tested). However, establishing structural-data links of this type of links is difficult because of the problem of dis-aggregating data, as previously discussed for the data links. In this case, data at a more aggregate level in the SD model would have to be dis-aggregated into structure in the PERT/CPM model. The difficulty in automating this process is the same as for data: there are various possible solutions at the detailed level and the real-world factors that determine the "right" solution are difficult to be captured in any of the models and are difficult to be specified formally in a link. Any automated solution of this type is likely to generate structural readjustments that require too much further rework to be worth implementing.

Structural readjustments to the SD model from PERT/CPM data, other than changing the shapes of the dependencies, are unlikely to be successful as well. These would include adding SD-Tasks, SD-Dependencies or SD-Resource categories. The data considered available from the PERT/CPM model in SYDPIM basic mode does not provide any basis to derive these adjustments. Tasks, dependencies and resource categories are introduced in the SD model when the model is developed first time and are eventually readjusted when the structural links of work correspondence are developed and updated. In general, this type of structural readjustment in the SD model is not common and even desired while the project is underway – a SD structure changing frequently would consume a considerable amount of effort and could cause calibration problems. These structural changes are likely to be justified only when major changes in the project occur, and not from changes in the data of the PERT/CPM model.

Summary

In this section the formal analytical links of SYDPIM basic model were proposed. This included structural links, data links and data-structural links. A summary of these links is presented in table 7.6. These links will be used within the activities of the SYDPIM process logic. They need to be created and updated throughout the

SYDPIM process. The ideal of the SYDPIM process is to be as automated as possible in the practical implementation of the links, although human judgement will always be (and should be) required. In order to support this automation, some algorithms were also proposed along with the analytical links. These are also summarised in table 7.7.

7.4.4 The SYDPIM activities

This section describes in detail each of the activities of the SYDPIM process logic described in section 3.1. There are two main components of this process logic: SYDPIM planning and SYDPIM monitoring. Within this process logic, the SYDPIM activities develop, update and make use of the SYDPIM objects described in section 3.2. They also develop, update and invoke the analytical links specified in the previous section.

The description of this activities in the following sections will make reference to a series of actions that the “project manager” will have to perform. This is not intended to imply that in practice the actual person carrying out these actions will be *the* project manager. Depending on the complexity of the organisation, the project manager typically has a team of analysts who perform all the PERT/CPM monitoring and planning actions. Likewise, these same persons will perform the required actions within the SYDPIM activities.

SYDPIM planning

The planning process logic has three main entry points: diagnosis of project past from monitoring, development of initial plan in PERT/CPM model and development of initial plan in SD model. Other than in the beginning of the project, the entry point will always be the diagnosis from monitoring. The SYDPIM activities have been numbered sequentially from this entry point. Various paths can be followed within the process logic, reflecting different policies for the planning process. The recommended path in SYDPIM is based on the idea that a reliable high-level plan can be developed first in the SD model and only then translated into the operational level in the PERT/CPM model. This leads to the following sequence of activities:

- (P1b) SD forecast and analysis of project future – SYDPIM feature;
- (P2b) Re-plan project future in SD model – SYDPIM feature;
- (P3b) Readjust PERT/CPM model to SD plan – SYDPIM feature;
- (P5) Update analytical links if necessary – SYDPIM feature.
- (P1a) PERT/CPM forecast and analysis of project future – conventional feature;
- (P2a) Re-plan project future in PERT/CPM model – conventional feature;
- (P3a) Extract future behaviour from PERT/CPM model – SYDPIM feature;
- (P4a) Re-calibrate SD model to PERT/CPM plan – SYDPIM feature;
- (P6) Update target metrics – SYDPIM feature

This path gives the major emphasis to the use of the SD model to develop project plans, and where the PERT/CPM model is only used to sort out the operational details. This view is not familiar to most project managers. The traditional view is the one where the PERT/CPM model is at the core of the re-planning process, which can be enhanced by other more sophisticated tools or techniques. For the sake of an easier understanding, the following sequence will be used to describe the SYDPIM activities.

- (P1a) PERT/CPM forecast and analysis of project future – conventional feature;
- (P2a) Re-plan project future in PERT/CPM model – conventional feature;
- (P5) Update analytical links if necessary – SYDPIM feature.
- (P3a) Extract future behaviour from PERT/CPM model – SYDPIM feature;
- (P4a) Re-calibrate SD model to PERT/CPM plan – SYDPIM feature;
- (P1b) SD forecast and analysis of project future – SYDPIM feature;
- (P2b) Re-plan project future in SD model – SYDPIM feature;
- (P3b) Readjust PERT/CPM model to SD plan – SYDPIM feature;
- (P6) Update target metrics – SYDPIM feature

This path gives more emphasis to the traditional use of the PERT/CPM model to develop or readjust the project plan, and introduces the use of the SD model as a complement to test and enhance this plan.

Out of the nine activities, only two are a conventional feature of the traditional project control framework. This highlights the added value of the SYDPIM steps to the process. Each of these activities is now described separately.

The SYDPIM process logic also contains some decision-points which determine the alternative paths that can be followed. The SYDPIM analytical links can be used to help the project manager in tasking the right decision in these branches. This is discussed after the description of the SYDPIM activities. The main output of the SYDPIM planning process is a project plan. This plan is represented in both the PERT/CPM and SD models. However, while the two models contain similar information they also contain planning information not present in the other model, or present in a different level of detail. Therefore, the SYDPIM project plan consists of all the planning information present in both models. This integrative view is also discussed.

(P1a) PERT/CPM forecast and analysis of project future – conventional feature

This activity can be entered in three main scenarios: (a) beginning of the project, (b) from monitoring or (c) from activity (3b). In all cases there is PERT/CPM plan available for the future. The purpose of this activity is to forecast the project likely outcome based on this plan and assess the plan under the conventional critical path perspective.

If we are at the beginning of the project, there is no project past and so when this activity is entered only a proposed PERT/CPM initial plan is available. If this activity is entered from monitoring then the PERT/CPM current plan is updated with the actual results of the project past and the future plan available is the one from the previous control cycle (i.e. no corrective action implemented yet). The PERT/CPM model will forecast what will happen if nothing is done. Finally, if this activity is entered from activity (3b), then the future plan available correspond to a tentative plan already developed in the SD model. In this case, the PERT/CPM model is used to provide a more detailed critical-path based perspective of the SD plan – for example, what appeared a low risk plan in the SD model may result in short floats and high number of near critical paths.

This activity is a conventional feature. The analyses and forecasts produced are based on the traditional Critical Path Analysis (CPA). The critical path and the critical activities are identified. The tasks' floats are analysed and the near critical

paths are also identified. This analysis may identify serious risks (e.g. a high number of near critical paths) and suggest changes to the plan (e.g. transferring resources from activities with a high float to the near critical activities).

This activity is illustrated in figure 7.14 below. The shaded area of the input/output data represents data about the project past (when the activity is entered from monitoring). With the current plan, prior to any corrective re-planning, the future outcome of the project is forecasted by the PERTC/PM model.

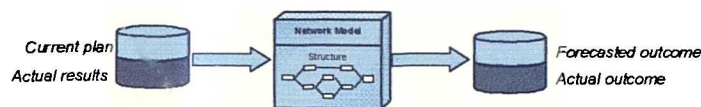


Figure 7.14 – Activity (P1a): PERT/CPM analysis and forecast of project future

(P2a) Re-plan project future in PERT/CPM model – conventional future

This activity is entered either when an existing project future plan has proven unsatisfactory (by PERT/CPM analysis and/or SD analysis), or when an initial project plan is to be developed. In the first case, there is a current PERT/CPM plan available with the past segment updated and with a future segment available. In the second case there is no plan at all.

If an initial project plan is to be developed, initial estimates will probably be available (e.g. using SLIM or KnowledgePLAN) for the major project milestones. The purpose of this activity is then is to develop a PERT/CPM network plan specifying the detailed tasks, and meeting the targets.

If there is a future plan available that needs to be changed or improved, the project manager will use the Critical Path Analysis (CPA) features, identifying the critical path, “crashing” activities, carrying out PERT analysis and do other types of CPA analysis (Nicholas 1990), until the plan is perceived to be satisfactory.

This activity is at the core of the conventional project planning process and is shown in figure 7.15 below. Based on the forecasted critical-path based outcome, a new plan is developed which ensures the desired objectives.

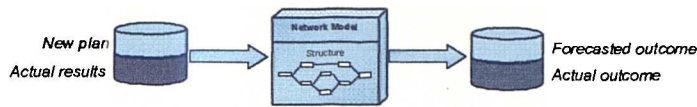


Figure 7.15 – Activity (P2a): PERT/CPM re-planning of project future

(P5) Update analytical links if necessary – SYDPIM feature

This activity is a non-conventional feature and is the core step to integrated the PERT/CPM model with the SD model. The purpose of this activity is to develop or update the analytical links between the two models.

This activity is entered when a new plan has been developed in the PERT/PM model. This can happen in two situations: (i) there is already a version of the analytical links available, and (ii) there are no analytical links available. In the first case, the SD model has already been used to simulate the project. As a new plan has been developed in the PERT/CPM model, newly added or modified tasks, dependencies and resources may require some re-mapping to the SD model. This may require that some links need to be updated. In the second case, a project plan is being transferred for the first time to the SD model and the analytical links need to be created.

The set of links proposed for SYDPIM basic mode includes three types of: structural links, data links and data-structural links. The last two types are based on the structural links and do not need to be created or updated. Their purpose is to ensure that the two models are representing the same project plan. They are automatically established and changed with the structural links. The only alternative that the project manager has is to remove some of these links proposed for SYDPIM basic mode or create new ones. This should be done when the analytical links are being developed for the first time.

The structural links establish the mapping between the constituent elements of the two models: tasks, dependencies and resources. Their purpose is not to ensure that the models are representing the same plan but to identify what corresponds to what between the two models. As the PERT/CPM plan is changed, this mapping may need to be updated. There are two types of structural links: structural correspondence links and structural consistency. The structural consistency links are used to check whether there is any inconsistency among the three mappings of the structural correspondence links.

In this activity, the links of structural correspondence are created or updated and the links of structural consistency are invoked to check whether there is any inconsistency.

The first step is to establish or update the links of structural correspondence. This should be done in the following sequence:

- (1) SC-WBS – PERT/CPM tasks are mapped or re-mapped to the SD-Tasks;
- (2) SC-OBS – PERT/CPM resources are mapped or re-mapped to the SD-Resources;
- (3) SC-WD – PERT/CPM dependencies are mapped or re-mapped to SD-Dependencies (both intra- and inter-task). The two algorithms proposed, SC-WD-1 and SC-WD-2, are available to automatically generate this mapping, which can then be readjusted manually by the project manager;

In case the analytical links already exist, how does the project manager know that an update is required? First, if an update is required, at least one of the validity conditions specified for each of these three links (see formal specification) is violated. For example, if a newly added PERT/CPM task has not been mapped to any SD-Task, then the validity condition PERT/CPM mapping of the SC-WBS link will be violated. Another possibility where the update of the structural links is required is when these have become inconsistent one another. For this purpose, the structural consistency links are called:

- (4) SCN-RA – checks whether the SC-WBS and SC-OBS links are consistent one another regarding resource allocation. The algorithm SCN-RA-1 proposed in the specification of this link can be used to automatically check this consistency and proposes possible solutions to eliminate the inconsistency;

- (5) SCN-WD – checks whether the SC-WBS and the SC-WD links are consistent one another. Again, two algorithms can be used: SCN-WD-1 and SCN-WD-2, which describe an automated process to check this consistency for intra-task and inter-task SD-Dependencies, respectively. In case inconsistencies are detected, both algorithms suggest possible solutions.

In case some inconsistencies are detected, corrections to the links of structural correspondence should be carried out.

Finally, if the analytical links are being developed for the first time, the project manager may choose to remove or create data-links or data-structural links, regarding the default set proposed for SYDPIM basic mode.

Once the analytical links have been created or updated as required the next activity in the SYDPIM process logic can be implemented.

(P3a) Extract future behaviour from PERT/CPM model – SYDPIM feature

Once a new project plan has been developed for the project future, the following steps in SYDPIM planning are aimed at allowing this plan to be tested in the SD model. The purpose of this activity is to extract the project future behaviour from the PERT/CPM plan. This behaviour consists of a set of dynamic patterns over-time which will have to be reproduced by the SD model. At this stage in the SYDPIM planning process, this set of patterns will be stored in the object “PERT/CPM future behaviour”.

While the conventional PERT/CPM tools produce automatically some project behaviour patterns (e.g. ACWP vs. BCWP), the concept of project behaviour as comprising a whole variety of dynamic patterns was introduced by the System Dynamics approach (Roberts 1964; Richardson and Pugh 1981). The concept of extracting the dynamic project behaviour from a PERT/CPM plan was introduced by the SYDPIM approach (Rodrigues and Williams 1997, Rodrigues 1997).

The specific set of patterns to be extracted in this activity was proposed in the formal specification of the SYDPIM object “PERT/CPM future behaviour”, where these

patterns will be stored. The technical algorithms required to calculate these patterns do not present any major difficulty and will not be presented here in detail. However, there are some particular issues worth clarifying.

Tables 7.8 through 7.11 in appendix B, present all the patterns, grouped per data type – table 7.8 is shown for illustrative purposes. In the second column the calculation process is described informally, along with some notes. In the last column the requirements to implement the calculations are identified. This includes the PERT/CPM plans required and the analytical links. The specific object variables of the “PERT/CPM future behaviour” to be updated are not specified in these tables, but their identification is straightforward. For example, the object variables to be updated with the pattern SAC[t], for each SD-Task and for the whole project, are as follows:

- PERT/CPM future behaviour.Schedule.Project schedule.Finish date[t]
- PERT/CPM future behaviour.Schedule.Task schedule.Finish date[i][t]

Except for the planned start date, all the other patterns are calculated for each SD-Task and for the whole project. The patterns that refer to resources are also calculated per SD resource category. The patterns that refer to final project targets have constant value over-time and hence their shape will consist of an horizontal line. This is because a PERT/CPM plan for the project future establishes fixed targets at the present moment and does not consider that these may change over-time – if the plan were to be implemented successfully, this is how the project would behave over-time. The target patterns considered are: planned start date, planned completion date (SAC), estimated cost at completion (CAC), estimated scope at completion (SCAC), and estimated cumulative resources applied at completion (CSPAC). The pattern of cumulative scope changes is also nil throughout the project, as a consequence of the SCAC target being constant. This implies that the PERT/CPM plan does not consider explicitly the emergence of further scope changes into the future.

The patterns for a SD-Task are extracted from the numerical inputs of all the individual PERT/CPM tasks which are mapped to it, according to the structural link SC-WBS. The patterns related to resource allocation also require the SC-OBS links to identify the PERT/CPM resources mapped to each SD-Resource. In some cases,

numerical fields of the tasks need to be summed-up for all mapped tasks. The SUM operator is used. Similarly, the SUMT operator is used to add profiles over-time – this operator returns an over-time pattern. The MIN and MAX operators are used to select minimum and maximum values of a set. Finally, the operator CUMULATIVE is used to calculate the cumulative value of an over-time series – this operator returns an over-time pattern.

In most cases the mapping of the tasks refers to tasks in the future segment of the current PERT/CPM plan. For some cases however, data is required from tasks in the initial PERT/CPM plan. For example, the BCWS pattern requires data only from this plan. The use of the initial plan requires that analytical links for this plan are available but the current version of the links refers the current PERT/CPM plan. It is therefore assumed that the initial version of the links is available – it was assumed previously that the SYDPIM object “Analytical links” will store all the past versions of the links. The tasks considered in the initial plan refer to the project future. Although the initial plan is not explicitly split into past and future segments in the object “PERT/CPM model”, it is assumed that it is easy to identify the tasks in this plan which fall the project future (i.e. the ones which start and/or finish date is beyond the present moment).

The calculation of the BCWP pattern requires data from tasks in both the initial and the current PERT/CPM plans. This is because the budgeted cost of the work performed in a task is equals the initially planned budget times the actual % scope progress. The initially budget is taken from the initial PERT/CPM plan while the % of scope progress is derived from the current PERT/CPM plan. Furthermore, this calculation only applies to the PERT/CPM tasks which are present in both the initial plan and the future segment of the current plan. So the extraction of this pattern from the PERT/CPM model requires the availability of both plans.

Many of these patterns are cumulative and so they build upon the present value of the specific variable. For example, the actual cost of work performed (ACWP) is equals the present value plus the planned effort expenditure in the future segment of the current PERT/CPM plan. If the project is just starting, the present value of these variables is nil, otherwise the present value is available from the patterns in the objects “PERT/CPM past behaviour” and SYMDB.

Pattern	Calculation	Requirements
Schedule		
SAC[t]	For the whole project and per SD-Task. Constant value – steady behaviour. SD-Task: = MAX planned finishing date of all tasks in the PERT/CPM current plan mapped to the SD-Task. Project: = Max SAC[t] of all SD-Tasks.	Current PERT/CPM plan SC-WBS link
Start Date[t]	For each SD-Task. Constant value – steady behaviour. SD-Task: = MIN planned start date of all tasks in the PERT/CPM current plan mapped to the SD-Task.	Current PERT/CPM plan SC-WBS link

Table 7.8 – Calculation and requirements of the schedule patterns of future behaviour extracted from the PERT/CPM model.

A simplified example of how some of these patterns would look like after being extracted from the PERT/CPM model is shown in figure 7.16 below (a software project is considered). The data collected from the PERT/CPM model tends to be discrete and so the patterns will consist of data-points which can be linked through smoothed lines to produce the continuous patterns.

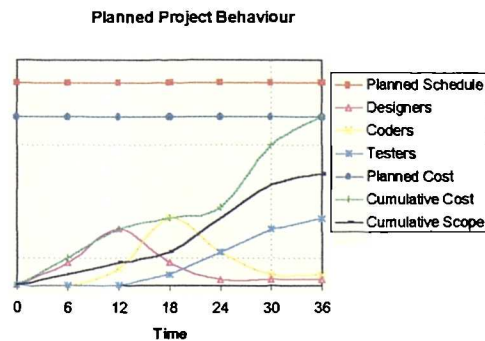


Figure 7.16 – Example of dynamic patterns of a planned project future behaviour extracted from the PERT/CPM model

As specified in the object “PERT/CPM future behaviour”, in SYDPIM basic mode there is a global set of patters that describe the behaviour of the whole project and an individual set of patterns for each SD-Task. In both cases, the resource related patterns are split by SD-Resource.

As shown in figure 7.17 below, at the end of this SYDPIM planning activity, the set of patterns proposed by SYDPIM is extracted from the PERT/CPM model and is stored in the “PERT/CPM future behaviour” object, using the SYDPIM structural links of work correspondence. These patterns describe the project future behaviour implicitly portrayed by the PERT/CPM plan – if the plan is to be implemented successfully, the project will exhibit this dynamic behaviour.

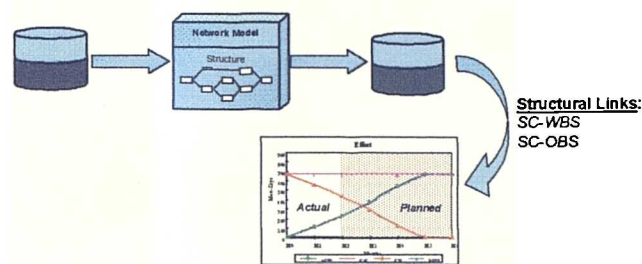


Figure 7.17 – Activity (P3a): extraction of PERT/CPM future behaviour

The analytical links used to extract the patterns of the PERT/CPM future behaviour are the SC-WBS and the SC-OBS, both actual and initial versions. The PERT/CPM plans required are the initial plan and the current plan. The SC-WBS links are used to aggregate the data from the elementary PERT/CPM tasks into the patterns of the SD-Tasks. The SC-OBS links are used to aggregate the data from the elementary PERT/CPM resources into the SD-Resources.

(P4a) Re-calibrate SD model to PERT/CPM plan – SYDPIM feature

The purpose of this activity is to readjust the SD model so that it represents the project future plan developed in the PERT/CPM model. In other words, the PERT/CPM plan is to be transferred to the SD model. This is basically a SD calibration exercise.

For the SD model to represent a project plan, two conditions must be verified: (i) the project outcome produced by the models must be the same, and (ii) the calibration must be consistent with the assumptions underlying the project plan. Condition (i) implies that the “PERT/CPM future behaviour” extracted from the PERT/CPM model in the previous activity is reproduced by the SD model within an acceptable degree of accuracy. Condition (ii) implies that while various SD calibrations may satisfy

condition (i), only the ones based on the correct assumptions are accepted – i.e. reproduction of the right behaviour for the right reasons.

The ideal of this activity would be the automated calibration of the SD model. Unfortunately, algorithms for complex models are not available. The route to follow in this direction would be optimisation (Coyle 1996). While useful, for complex models this technique only allows for initial solutions to be refined into a better fit. A major barrier to a fully automated process is the need to reproduce the desired behaviour for the “right reasons” – here meaning that the assumptions underlying the project behaviour are the same in the both SD and PERT/CPM models. In the SD models, these underlying assumptions are made explicit through the quantification of certain parameters (e.g. productivity, impact of learning on quality, etc.). The problem is that in the PERT/CPM model most of these assumptions are not quantified and remain implicit in the mind of the persons who developed the plan. Therefore, a fully automated process which would ensure the “right reasons” is not possible to implement because required inputs are missing from one model. A longer further discussion about this topic could be presented here, but it falls outside the scope of this research. Equally, it should be stressed that it is totally beyond the scope of this research to propose automated algorithms for model calibration, regarding this or any other SD purpose. Instead, the SYDPIM perspective is that this process should be iterative and based on human judgement, which will always be required. The process can be enhanced with the support of tools or algorithms which automate some of its steps. This is the purpose of various analytical links proposed.

When this activity is entered the structural links are updated and validly established between the two models (activity (P5)). The process within the activity consists mainly of transferring data and readjusting the SD model calibration. SYDPIM proposes that this is based on the following principles:

- (1) in an initial step, some planning data can be automatically transferred from the PERT/CPM model to the SD model, through the implementation of data-exchange links and data-structural links;
- (2) the project behaviour produced by the SD model can be automatically checked for accuracy against the “PERT/CPM future behaviour”, through the implementation of data-consistency links;

- (3) calibrations produced manually for the SD model can be automatically checked for validity against some of the planning assumptions in the PERT/CPM model, through the implementation of data-consistency links and data-structural consistency links.

Based on these principles, a structured process can be proposed wherein the various analytical links are employed. The following algorithm describes the basic flow of this process:

Algorithm: { calibration of SD model for PERT/CPM future plan }

1. Transfer planning data from PERT/CPM model to SD model
2. Readjust SD structure (dependencies) from data in PERT/CPM model
3. Simulate project in SD model
4. Check if SD project behaviour fits "PERT/CPM future behaviour"
5. IF SD outcome is satisfactory END
6. Diagnose inconsistencies
7. Re-calibrate SD model manually (update analytical links if structural changes)
8. Check consistency of new calibration against project plan in PERT/CPM model
9. IF calibration is not satisfactory THEN diagnose inconsistencies AND GOTO 7
10. GOTO 3

The process starts with the transfer of some basic data that characterises the project future plan in the PERT/CPM model. A structural readjustment is also carried out in the SD model, which proposes an initial shape for the SD dynamic intra- and inter-task dependencies in the SD model (which do not have to be the final ones). The project is then simulated in the SD model and the resultant project outcome is checked against the PERT/CPM future behaviour. In most cases, an acceptable fit will not be achieved. This is because the automated data transfer and structural readjustment of steps 1 and 2 do not update many other relevant SD parameters responsible for the project outcome (e.g. productivity, defect generation rate, etc.) – not only they affect the outcome in the SD model as they also do in the real world, but are not quantified explicitly in the PERT/CPM model. Therefore, after diagnosing the inconsistencies in the project outcome, the modeller will have to adjust these parameters manually – if structural changes are required to the SD model (which is unlikely), the analytical links should be updated as required. In order to calibrate the process parameters of the SD model, the modeller can find

support in the process metrics of the project past stored in the SYMDB, in case this database is being implemented (see specification in appendix C.3). However, the modeller cannot just change the model's parameters arbitrarily – the SD model will have to reproduce the desired behaviour for the right reasons. Ideally, one would like to check all possible violations to the assumptions in the plan. As previously mentioned, this is not possible. But there are some parameters that can be checked and these refer to the targets of the initial plan. These should not be changed, even though that could lead to the desired behaviour. For example, the currently planned completion date of an on-going task might now be different according to the PERT/CPM plan. A solution would be to change in the SD model the initial planned completion date. However, that does not reflect reality, because the date was adjusted at the present moment and not in the beginning of the project. Preferably, this re-planning decision should be generated endogenously within the model for the present moment. Alternatively, an exogenous decision can also be generated as an *adjustment* to the initially planned completion date. Assuming that the new calibration does not change the initially planned targets, the project is simulated once again and the process is repeated until the SD model is reproducing accurately the "PERT/CPM future" behaviour, most likely for the right reasons.

In order to implement this process, various SYDPIM analytical links are used. This usage consists of "invoking" the links as if they were software routines or applications which would implement their equations. A more detailed description of this algorithm is presented, identifying the use of the analytical links.

Algorithm: { calibration of SD model for PERT/CPM future plan }

1. Transfer planning data from PERT/CPM model to SD model
 { IF beginning of the project THEN invoke links:
 DEI-1.PERT-SD : Profiles of Project Resources Availability
 DEI-2.PERT-SD : Project Start Date
 DEOI-1.PERT-SD : Start and Finish Dates of SD-Tasks
 DEOI-2.PERT-SD : Budget of SD-Tasks
 DEOI-3.PERT-SD : Scope of SD-Tasks
 DEOI-4.PERT-SD : Profiles of Resource Allocation to SD-Task
 DEOI-5.PERT-SD : Budget Breakdown of SD-Tasks
 ELSE invoke links:
 DEI-3.2.PERT-SD : Present Level of Project Resources Availability
 DEOI-9.2.PERT-SD : Present Level of Resource Allocation to SD-Task
 DEOI-6.PERT-SD : Present Planned Start / Finish Dates of SD-Tasks
 DEOI-7.PERT-SD : Present Planned CAC of SD-Tasks
 DEOI-8.PERT-SD : Present Planned SCAC of SD-Tasks
 <Simulate project in SD model>
 DEI-3.1.PERT-SD : Profiles of Project Resources Availability
 DEOI-9.1.PERT-SD : Profiles of Resource Allocation to SD-Task
 Note: the modeller may choose not to invoke these links }
 }
 2. Readjust SD structure (dependencies) from data in PERT/CPM model
 { Invoke links:
 DSR-1.PERT-SD: adjust intra-task SD-Dependencies
 DSR-2.PERT-SD: adjust inter-task SD-Dependencies
 Note: the modeller may choose not to invoke these links }
 }
 3. Simulate project in SD model
 4. Check SD project outcome against "PERT/CPM future behaviour"
 { Invoke links:
 DCOI-6.1.SD-PERT : Planned Project Resources Availability
 DCO-1.1 : Planned Start / Finish Dates of SD-Tasks
 DCO-2.1 : Planned Effort of SD-Tasks
 DCO-4.1 : Planned Scope of SD-Tasks
 DCO-3.1 : Planned Resource Allocation to SD-Tasks }
 }
 5. IF SD outcome is satisfactory END
 { If none of the consistency links in step 4 is violated }
 6. Diagnose inconsistencies
 { Check which consistency links and respective patterns are violated }
 7. Re-calibrate SD model manually
 { Change the SD input parameters which are likely to eliminate the
 inconsistencies above. Consult process metrics in SYMDB if possible. If structural
 changes are made to the SD model, adjust analytical links as required }
 8. Check consistency of new calibration with project plan in PERT/CPM model
 { Invoke following links to check that parameters of the initial plan were not
 modified:
 DCI-1 : Profiles of Project Resources Availability
 DCI-2 : Project Start Date
 DCOI-1.PERT-SD : Start and Finish Dates of SD-Tasks
 DCOI-2.PERT-SD : Budget of SD-Tasks
 DCOI-3.PERT-SD : Scope of SD-Tasks
 DCOI-4.PERT-SD : Profiles of Resource Allocation to SD-Task
 DCOI-5.PERT-SD : Budget Breakdown of SD-Tasks
 Invoke the following links to check whether intra- and inter-tasks
 SD-Dependencies were modified:
 DSC-1 : check intra-task SD-Dependencies
 DSC-2 : check inter-task SD-Dependencies }
 }
 9. IF calibration is not satisfactory THEN diagnose inconsistencies AND GOTO 7
 { Check which consistency links are violated. Note that modifications to the
 parameters that describe the initial plan should not be allowed, whereas
 changes to the SD-Dependencies are allowed at the modeller's explicit
 preference. }
 10. GOTO 3

In step (1) the data exchange links of type “input-input” and “output-input” are invoked. Depending on which plan is being transferred, the initial or the current plan, a different set of links is invoked. If we are the beginning of the project, then the initial plan is transferred. In this case, the links invoked transfer the targets (schedule, scope, effort, and resources) for the whole project and for each SD-Task, from the PERT/CPM model to the SD model. If we are half-way through the project, then the current plan is transferred to the SD model. In this case, the links invoked generate exogenous decisions in the SD model, which will adjust the project planned targets at the present moment. It is important to note that these will be exogenous adjustments to the endogenous decisions generated within the SD model. Since these adjustments are not the only “valid” way to calibrate the SD model for the current plan (e.g. other parametric changes may be able to generate these same decisions endogenously), invoking these links is not mandatory. However, SYDPIM recommends these links are invoked when there is in reality a major cause for the decisions which is not captured in the SD model. In case the modeller chooses this route, there is an intermediate simulation required. This takes place after the exogenous decisions at the present moment being introduced in the model and prior to adjusting the planned future profiles of resources availability and allocation. The reason for this is that the resource decisions introduced in the SD model at the present moment affect the present level of the resources, which in turn affects the exogenous decisions to be generated regarding the future profiles. So the new present level of resources should be generated (via simulation) before links DEI-3.1 and DEOI-9.1 are invoked. It is important to note that this intermediate simulation is not aimed at reproducing a desired future behaviour.

In step (2) the data-structural links which adjust the SD-Dependencies are invoked. These links produce initial shapes for the progress curves of the dependencies, which are consistent with the planned scope progress over-time in the PERT/CPM plan. These shapes are not necessarily the “valid” ones, because scope progress is not just a consequence of these dependencies, and hence may need to be readjusted by the modeller. However, they will be a good starting point. Alternatively, the modeller may choose not to invoke these links.

This data transfer and structural readjustment generate an initial calibration for the SD model in an attempt to reproduce the “PERT/CPM future behaviour”. As the SD model is simulated in step (3), the next step is to check whether the produced project outcome fits into this behaviour and if not make the necessary readjustments in the SD model.

This takes place in steps (4) through (10). In step (4) data consistency links are invoked to check whether the SD model is reproducing the required patterns. DCO links play an important roles here since they check the consistency of the output data produced by both models. Each link checks various patterns of project behaviour. For example, the link DCO-2.1 checks all patterns related with the project effort, which account for a total of 14 patterns – see specification of the links. A critical issue at this stage is the criteria used within the links to assess whether the output from both models are “close enough” one another. In this particular case, to assess how close the patterns from SD model are to the patterns extracted from the PERT/CPM model. This leads to the more general issue in System Dynamics of quantitative tests for “goodness of fit”. While it is not the purpose of this section to discuss this topic in great detail, it is important to provide some basic elements upon which the data consistency links can be implemented.

Assessing how close the SD model output is to certain desired patterns, calls for quantitative techniques and was already the subject of study. Sterman (1984) proposes quantitative indices based on Theil’s statistics of “goodness of fit”. These are as follows:

- RMSPE (Root of the Mean of the Squared Percent Errors) – this is an overall index measure in percentage. The calculation is applied to the two time series, the model output and the pattern to be reproduced (the two series must have the same number of data points). The result gives an overall measure how far the model output is from the target series (0% meaning a perfect fit). The causes of this gap are then divided across the following three indices calculated as percentage.
- UM (bias) – this index indicates how much of the gap is due to a general bias in the time series. This bias is detected through a difference in the means of the two series and visually shows as a vertical shift in the model output.

- US (trend) – this index indicates how much of the gap is due to a general difference in the trend of the time series. The difference in the trend is detected through a difference in the standard deviations of the two series and visually shows as a “rotation” in the model output.
- UC (point prediction) – this index indicates how much of the gap is due to a difference in the series “point by point”. This difference is detected through the value of the correlation coefficient and the standard deviations of the two series. Visually it literally shows a lack of overlap of the curves point by point.

Once these statistics have been generally accepted as adequate for SD models (Barlas), they are used in SYDPIM. In addition there are other indices suggested in SYDPIM:

- *correlation coefficient* – this is a classic statistic which assesses whether and how two series are correlated. A good fit should give a value close to 1 (the maximum of this statistic);
- *absolute and percentage deviations of cumulative results* – these indices are aimed at assessing whether the cumulative result of a certain pattern is close to reality, regardless of how accurately the model reproduced how it evolved over time. A good example are the effort expenditures. In some cases, a simplified model structure may miss some irregular shapes of the evolution of these curves over time, but the final result may still be accurate. This may be satisfactory for the model purpose. SYDPIM suggests this analysis for some relevant patterns, namely: (i) schedule, (ii) effort, (iii) scope, (iv) defects and (v) process metrics (e.g. productivity, defect generation), these two indices should be calculated.

Tables 7.12 and 7.13 present an example of the application of these indices taken from the case-study undertaken during this research (the KDCOM Project). Table 7.12 presents the correlation coefficient and indices proposed by Sterman (1984), where seven patterns are tested for goodness of fit. A correlation coefficient and a low RMSPE indicate a good fit of the SD output into the desired patterns. For the CTC patterns, where the RMSPE is higher this is mostly due to a “point-by-point” error whereas the average (Bias) and the trend of the patterns is good. These indices focus on the shape of the patterns. The indices presented on the table 7.13 focus on the cumulative final results at the end of the simulation. It should be noted

that the accuracy of the final cumulative results is very important in life-cycle type of systems such as a project – such final results will dictate important objectives like the project cost and product quality.

"Goodness of fit" of SD project behaviour (summary)

Metric	Correlation	RMSPE	Bias	Trend	Point Prediction
1. Schedule	1,00	0,10%	7,69%	92,31%	0,00%
2. Effort					
2.1 ACWP	1,00	0,17%	36,95%	45,78%	17,27%
2.2 CAC	1,00	0,20%	27,24%	72,76%	0,00%
2.3 BCWP	1,00	0,15%	0,42%	1,11%	98,47%
2.4 CTC	1,00	28,91%	0,42%	1,11%	98,47%
3. Staff	1,00	0,91%	6,72%	7,77%	116,27%
4. Defects	0,95	37,92%	59,95%	0,33%	52,05%

Table 7.12 – Example of quantitative assessment of “goodness of fit”: the correlation coefficient and Theil’s statistics proposed by Sterman.

Accuracy of final results

Metric	Actual	Simulated	Deviation	
			Absolute	Percentage
1. Schedule	540,00	538,00	-2,00	-0,37%
2. Effort				
2.1 Development	352,20	349,87	-2,33	-0,66%
2.2 QA	117,40	117,82	0,42	0,36%
2.3 Rework	117,40	113,23	-4,17	-3,55%
2.4 Total	587,00	587,90	0,90	0,15%
3. Defects (index per KLOC)				
3.1 Reworked	34,43	34,01	-0,42	-1,21%
3.2 Generated	57,38	53,67	-3,71	-6,46%
4. Productivity (LOC/man-day)				
4.1 Overall	32,49	32,44	-0,05	-0,15%
4.2 Development	54,14	54,50	0,36	0,67%

Table 7.13 – Example of quantitative assessment of “goodness of fit”: accuracy of final cumulative results.

These quantitative indices provide an objective measure of how close the model is in fitting the desired patterns. Based on the values achieved, the project manager will decide in step (5) whether the SD outcome is satisfactory. There is no single value which will determine whether the required “goodness of fit” will be achieved. An acceptance criteria requires the diagnosis of the values of all the indices and the likely reasons for the deviations. Depending on the specific model at hand and on many other factors specific to the project, human judgement and experience is required to establish this criteria (see Sterman (1984) for a discussion).

If some patterns are not being reproduced with the required accuracy, then some data-consistency links will fail. In step (6) the project manager will identify these inconsistencies and will try to eliminate these through re-calibration of the SD model in step (7). To support this re-calibration process, the project manager should consult the process metrics regarding the project past stored in the SYMDB (if in use). This will help validating the calibration for the project future. If structural changes are required in the SD model (which is unlikely) the analytical links should be updated as required. In a project model, a possible way of achieving an effective calibration in terms of “goodness of fit” is to change the initially planned targets. This is particularly effective for the patterns that represent the project targets, like the expected completion date of tasks. This type of parametric changes should not be allowed because they do not reflect reality: the initially planned target dates were decided at the beginning of the project and not half-way. In order to ensure that this type of re-calibration is not implemented, in step (8) data consistency links are invoked to check whether the initial plan in the SD model is still consistent with the initial PERT/CPM plan. In this step the DSC links are also invoked to let the user know whether changes to the progress curves of the intra- and inter-task SD-Dependencies were made during re-calibration, having become inconsistent with the scope progress planned in the current PERT/CPM plan. While these changes are allowed (scope progress is not just a function of these dependencies), it is important to ensure that this is an explicit calibration option. If the new calibration is not satisfactory then it should be reviewed again in step (7). This verification takes place in step (9). If the new calibration is satisfactory, then the project is simulated once again and the process cycles back to step (3). If this new valid calibration reproduces well the PERT/CPM future behaviour portrayed by the current PERT/CPM plan, then this SYDPIM activity is complete.

During the re-calibration process of steps (4) to (10), the indices of “goodness of fit” proposed above will be used to check whether the SD model is reproducing well the desired behaviour. However, it should be noted that this is not the only requirement for a valid calibration. “Goodness of fit” is not the same as ensuring that the model is reproducing the desired behaviour for the right reasons. Achieving a valid calibration is critical because the reliability of the forecasts and diagnosis produced by the SD model depend upon it – and thereby the usefulness of the whole SYDPIM process. Establishing an objective or even a quantitative acceptance criteria for the

The way in which the SD model is used also depends on the scenario within which the activity is entered. This activity can proceed from SYDPIM monitoring function or from activity (P4a), where the SD model was just re-calibrated to reproduce the future segment of the PERT/CPM current plan.

If this activity is entered from monitoring, and the project past went just as planned then the future behaviour produced by the SD model will be *steady* and equals the one shown by the PERT/CPM model. However, in most cases deviations would have occurred and the project future behaviour produce by the SD model will be *unsteady*. This means that the SD simulation is showing a future where the project targets will change, like the completion date slipping by. In this scenario it is also very likely that the project future will differ from the one shown by the PERT/CPM model. Once the to models have just been updated with the project past, because they consider different factors in different detail they will probably forecast a different future.

Once the project future is known the next step is to diagnose why the unsteady outcome is likely to be as forecasted. This diagnosis is based on the analysis of the individual factors that affect the performance drivers (effects on productivity and defect generation) and on causal analysis (i.e. which specific feedback loops are driving the project outcome). On top of this, the project manager may wish to assess the compounding impacts of risks over the already problematic project outcome. This is done through the introduction of risks without changing the plan. It is important to note that in this activity no re-planning decision will be implemented in the SD model – this will be done later. Since the outcome is unlikely to be satisfactory and inconsistent with the one produced by the PETRT/CPM model, the project manager will have to decide whether the project plan is to be improved in the PERT/CPM model or in the SD model.

In the other possible scenario, this activity was entered from activity (P4a). A new project plan was just develop in the PERT/CPM model and the SD model was re-calibrated to reproduce this plan. Hence, it will produce a *steady* future behaviour, which is the same as the one extracted from the future segment of the current PERT/CPM plan. Therefore, in this scenario there is no need to use the model to forecast the project future (purpose (a) above). Once the SD model was “forced” to

reproduce the PERT/CPM plan, it will be used mainly to uncover assumptions underlying this plan and to identify and assess its sensitivity to risks (i.e. is the plan realistic? what are the relevant risks? is the plan robust?)

Uncovering assumptions in the project plan is based on the fact that the SD calibration requires the explicit specification of the expected values of various performance or process-related metrics – e.g. development productivity (function-point/man-hour), defect generation rate (defect/function-point), effort to detect defects (man-hour/defect), and effort to fix a defect (man-hour/defect). A PERT/CPM plan does not force these assumptions to be made explicit. These metrics can be assessed for feasibility in three ways: (1) they can be compared against standard metrics available in the literature, (2) against the calibration of a same or similar SD model for a past similar projects, or (3) against expert judgement. The comparison against standard metrics requires that the metrics have similar definitions. The use of expert judgement also requires that the definition of the metrics is well understood by the “expert”. Finally, the comparison against past calibrations is the more powerful means of sanity-checking. However, this also requires validation: the variables under comparison must have a similar definition and the past project must be similar. This means of sanity-checking the project plan becomes more practical and easier if the organisation develops and maintains a database of past applications (this can be done while applying SYDPIM to various projects) – this database would include the SD models used, the calibration metrics, and the results. Although it requires time and effort, the development of such database is strongly encouraged by SYDPIM. Apart from sanity-checking future plans, it can also be used as the basis to evaluate the performance of an on-going project, and supports the model development through re-usage of full models or model components.

Where any of the metrics used to calibrate the model for the PERT/CPM plan proves unrealistic (e.g. error generation rate too low), this indicates an invalid assumption – these can be considered as “planning risks”. The SD model can be used to investigate “what-if” the unrealistic metrics are changed to realistic values. If in these conditions the project outcome does not meet the targets, the now unsteady plan needs to be changed. If the project still meets the targets it may well be at the expense of a less robust plan now more sensitive to external risks.

Risks can be identified and the plan's sensitivity can be assessed in the SD model by changing conditions (internal or external) over which management has no direct control. The SD project model should incorporate the required variables that represent changes in these conditions. For example, there may be a variable to quantify the number of requirements changes over time requested by the Client. Depending on the plan's robustness, the outcome may become unsteady with the targets slipping by. In this way, the project manager can identify important risks (this quantitative assessment of risks can also be used as input to the existing project risk management framework eventually in place; e.g. a risk register). This will be useful information for re-planning. As with the previous scenario, the SD model is also used to diagnose the impacts of risks and of more realistic assumptions. The project manager will now have to decide whether the project plan is to be improved in the PERT/CPM model or in the SD model.

This SYDPIM activity takes place only within the SD model and consists of a "standard" SD analysis. The links with the PERT/CPM model are not used. The aim is to forecast the likely project outcome, identify the underlying causes, uncover unrealistic planning assumptions, and identify the risks and assess its impacts on the project future. The SYDPIM planning process, after the implementation of this activity, is shown in figure 7.19 below.

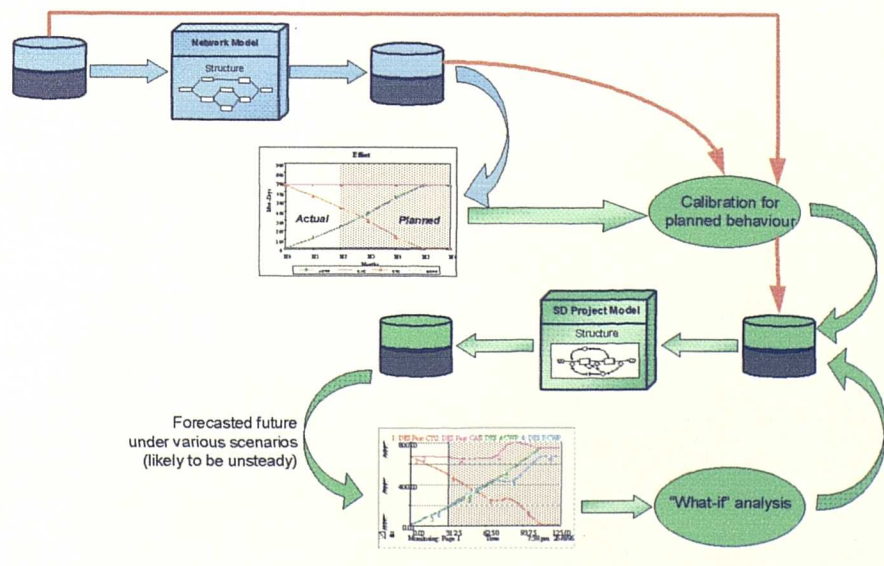


Figure 7.19 – Activity (P1b): SD analysis of the project future

(P2b) Re-plan project future in SD model – SYDPIM feature

The purpose of this activity is to develop a realistic and robust plan for the project future. It can be entered in three different scenarios: (a) development of an initial plan for the project, (b) readjustment of the current plan, and (c) to improve a new future plan developed in the PERT/CPM model. In all cases, the main output from this activity will be a future plan, which exhibits a steady behaviour in the SD model – no project manager will want to move ahead with a plan which is forecasted to over-run and hence “doomed to failure”. Another important output will be a list of the most relevant risks and correspondent mitigating actions.

The development of an initial plan will happen only once at the beginning of the project. At this moment, the project manager may wish to use the SD model to develop high-level estimates and a high-level plan for the project before developing the detailed PERT/CPM plan. SYDPIM considers this possibility. Here the SD model is used not only as a planning tool but also as a front-end estimating tool to produce and high-level estimates or verify whether given estimates are realistic. Past applications show that in some cases, a SD project model can be used in this way (Abdel-Hamid 1990), like the empirical estimating tools widely used in the software industry (e.g. COCOMO, SLIM, and KnowledgePLAN). However, a SD model should only be used for front-end estimating if valid metrics are available for calibration. The source for these “valid metrics” would be past applications of SYDPIM, using the same SD project model, and in similar projects. Another possible source of valid metrics would be the project past history (if the project is half-way through). However, in most applications of front-end estimating, there is no project past available to calibrate the model. Another potential problem is the way in which the SD estimates would compare with estimates produced empirical estimating tools – this can be controversial. The accuracy of the estimates depends on the number of past projects available in the databases, the factors considered explicitly in each of the two types of models, and the “level of tailoring” allowed in each of the models for the specific project. Overall, most organisations do not have a significant database of past applications of SD project models, and in particular clusters of similar projects. Therefore, in these cases, the use of the SD model as a pure front-end estimating tool in SYDPIM is not recommended (nevertheless, any

organisation considering the use of SYDPIM should feel encouraged to develop such database; it will be useful not only to support a SD-based front-end estimating but also the implementation of SYDPIM in planning and monitoring). An alternative way of using a SD project model as part of front-end estimating is to verify whether estimates produced by conventional estimating tools are realistic. This is similar to how the SD model is used to assess the PERT/CPM plan in the previous activity. Used in conjunction with the estimating tool the SD model is used to refine the estimates and develop an initial plan.

In the other two scenarios, the SD model is has been updated with the project past from monitoring and is showing a project future which by some reason is not satisfactory. It may be that given the project past, the project future is now showing delays and overruns and so the future plan needs to be changed. Or it may be that if a new plan was already developed in the PERT/CPM model which appears to ensure the objectives, the SD model reveals that the underlying assumptions are unrealistic or that the plan is too sensitive to risks. In any case the SD model contains a future plan which needs to be changed and improved.

Changing the project plan in the SD model will consist of changes in the schedules, budgets, scope, resource allocation (as in the PERT/CPM model), changes in other planning parameters not explicitly considered in the PERT/CPM model (e.g. QA level), more radical changes to the product development process (e.g. from the classic life-cycle to incremental development), and changes in various control policies which are also not considered explicitly in the PERT/CPM model (i.e. how progress is monitored and re-active actions generated). This last type of change is important because the SD model assumes that the project outcome depends only on the specified targets and operational work plan but also on the way in which the plan is going to be controlled. These changes are made and tested through simulation. The process consists of “trial-and-assess” what-if analysis. At the end of the process a project plan with a steady and satisfactory outcome should have been achieved.

Another very important purpose of this SYDPIM activity is, given a project plan, identify the relevant risks and correspondent mitigating actions. In order to achieve this, a list of likely risks is tested in the model to assess their impacts. Mitigating

actions are tested on top of these risks to assess their effectiveness in countering the disruptive impacts. The SD model will also provide an estimate for the cost of these mitigating actions. Used in this way, the SD model provides an excellent tool for pro-active risk management. This is one of the major benefits of the SYDPIM methodology.

This SYDPIM activity takes place within the SD model with no linkage to the PERT/CPM model. It consists of a “standard” SD analysis to improve the system performance. At the end of this activity the project manager will have developed a realistic and robust future plan for the project, which exhibits a steady behaviour. The project manager will also have identified and ranked the relevant risks, identified the most effective mitigating actions, and imposed some of these actions over the plan (possibly the ones against the more likely risks). The SYDPIM planning process after the implementation of this activity is shown in figure 7.20 below.

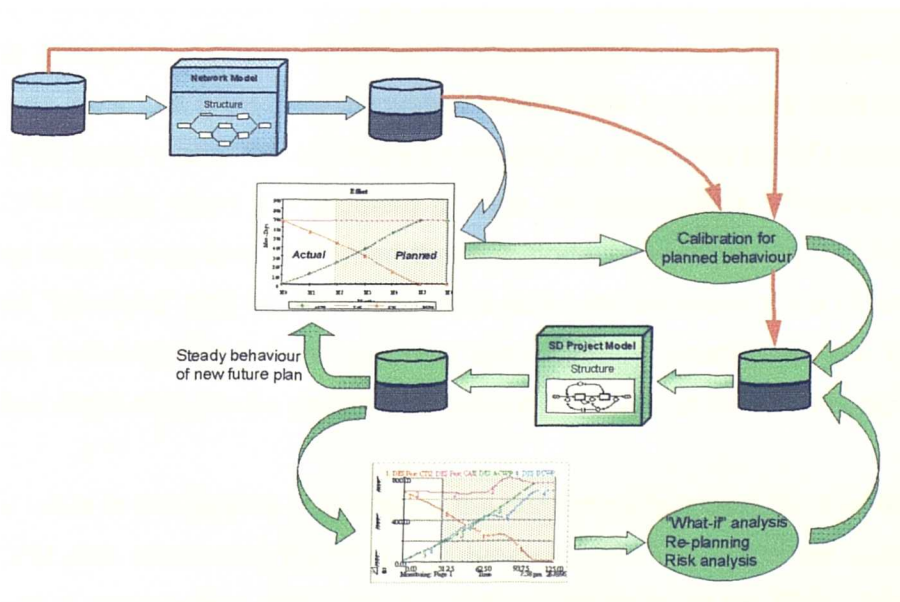


Figure 7.20 – Activity (P2b): SD re-planning and risk analysis

(P3b) Readjust PERT/CPM model to SD plan – SYDPIM feature

The purpose of this activity is to readjust the PERT/CPM model to the new future plan developed in the SD model, in the previous activity. Basically, this consists of

“transferring” this SD future plan to the PERT/CPM model. As expected, the SYDPIM analytical links will play an important role in this operation.

There are two different scenarios in which this activity is performed: (a) beginning of the project and (b) the project is underway. In the first scenario, there can be two situations: (a) there is already a PERT/CPM plan developed and this needs to be readjusted for the SD plan, or (b) the PERT/CPM model plan needs to be developed from scratch. In the second scenario there will always be a future PERT/CPM plan available. In the first scenario the PERT/CPM plan to be readjusted or developed is the initial plan, which is considered as the current plan at the beginning of the project. In the second scenario, the plan to be readjusted or developed is the future segment of the current PERT/CPM plan.

The ideal of this activity would be the automated readjustment of the PERT/CPM plan. In the situation where there is no version of the PERT/CPM plan available, this would consist of an automated generation of a new plan. In SYDPIM this would be done through the establishment of data-exchange links and of data-structural adjustment links. As previously discussed when these type of links were proposed for SYDPIM basic mode, the automated exchange of data from the SD model to the PERT/CPM model faces the problem of data dis-aggregation. Regarding data-structural links, it was also concluded that the models considered in SYDPIM basic mode are “standard” and do not contain data about the structure of the other model. The data exchange and data structural links already established are the only automated steps that can be performed within this activity, in SYDPIM basic mode.

A critical issue in this activity is to specify the conditions that determine whether the PERT/CPM plan represents the SD plan. Since the PERT/CPM model stands at a lower level of aggregation, there can be various “solutions” at the PERT/CPM level that represent the SD plan. In SYDPIM, it is considered that there is no single formal criteria that can dictate the “right” solution. Instead, the plans in the two models are checked for consistency, using the SYDPIM consistency links, and based on this information human judgement is required to determine whether the PERT/CPM solution is satisfactory or not.

The following principles are proposed to readjust the PERT/CPM plan to the SD plan:

- (1) in an initial step, some planning data can be automatically transferred from the SD model to the PERT/CPM model, through the implementation of data-exchange links. Before this is done, the structural links between the two models must be updated as necessary (by calling activity (P5));
- (2) whenever the PERT/CPM plan is changed, there are two operations that must be performed: (i) the analytical links are updated as necessary (by calling activity (P5)) and (ii) the "PERT/CPM future behaviour" should be re-extracted from the PERT/CPM model (same process as in activity (P3a));
- (3) the consistency of the PERT/CPM plan being readjusted with the SD plan can be checked on three different ways:
 - (3.1) if we are at the beginning of the project then the initial PERT/CPM plan can be checked against the targets of the initial plan in the SD model through the implementation of data-consistency links;
 - (3.2) the "PERT/CPM future behaviour" extracted from the PERT/CPM current plan can be automatically checked for accuracy against the project behaviour produced by the SD model, through the implementation of data-consistency links;
 - (3.3) the PERTCPM plan can be checked for consistency against the SD dependencies through the implementation of data-structural consistency links. However, this consistency is not mandatory (see discussion in activity (P4a)).

The need to extract the "PERT/CPM future" behaviour whenever the PERT/CPM plan is changed, results from the fact that the changed plan will most likely portray a different behaviour which needs to be checked against the SD behaviour. This process of extracting the PERT/CPM behaviour corresponds to the simulation in the SD model.

The need to update the analytical links at the beginning of the process results from the fact that the new SD plan developed in activity (P2b), which is now to be transferred to the PERT/CPM model, may incorporate new SD-Tasks, SD-Resources and SD-Dependencies, or some of these may have been removed, as part of re-planning. In these cases, it is necessary to update the mapping of tasks, resources and dependencies between the two models. The need to update the

analytical links after the PERT/CPM plan has been changed also results from the fact that these changes may include adding or removing tasks resources and dependencies in the PERT/CPM model.

Based on these principles, a structured process can be proposed as described in the following algorithm:

Algorithm: { readjustment of PERT/CPM model to SD future plan }

1. IF beginning of the project AND there is no PERT/CPM plan available GOTO 10
2. Call activity (P5) to update structural links as necessary
3. Transfer planning data from SD model to PERT/CPM model
4. Extract "PERT/CPM future behaviour" from PERT/CPM model
5. IF beginning of the project THEN check consistency of initial plan
6. Check "PERT/CPM future behaviour" against behaviour produced by SD model
7. Check consistency of SD dependencies with PERT/CPM plan
8. IF consistency-checking is satisfactory END
9. Diagnose inconsistencies
10. Re-adjust or develop from scratch manually the PERT/CPM future plan
11. Call activity (P5) to update structural links as necessary
12. GOTO 4

The process basically consists of a loop where the PER/CPM plan is manually readjusted and checked for consistency against the initial plan in the SD model, against the future behaviour produced by the SD model and against the SD-dependencies. Whenever a new version of the PERT/CPM plan is developed, the analytical links are updated as required.

In order to implement this process, various SYDPIM analytical links are "invoked". A more detailed description of this algorithm is presented, identifying the use of the analytical links.

Algorithm: { readjustment of PERT/CPM model to SD future plan }

1. IF beginning of the project AND there is no PERT/CPM plan available GOTO 10
2. Call activity (P5) to update structural links as necessary
3. Transfer planning data from SD model to PERT/CPM model
 { IF beginning of the project THEN invoke links:
 DEI-1.SD-PERT : Profiles of Project Resources Availability
 DEI-2.SD-PERT : Project Start Date
 ELSE invoke links:
 DEOI-10.1.SD-PERT : Profiles of Project Resources Availability }
4. Extract "PERT/CPM future behaviour" from PERT/CPM model
 { This process is the same as the one previously described in activity (P3a). In case the initial plan is being re-adjusted / developed, it is assumed that this is at the same time the future segment of the current PERT/CPM plan, from where the project behaviour is extracted. }
5. IF beginning of the project THEN check consistency of initial plan
 { Invoke following links:
 DCI-1 : Profiles of Project Resources Availability
 DCI-2 : Project Start Date
 DCOI-1.PERT-SD : Start and Finish Dates of SD-Tasks
 DCOI-2.PERT-SD : Budget of SD-Tasks
 DCOI-3.PERT-SD : Scope of SD-Tasks
 DCOI-4.PERT-SD : Profiles of Resource Allocation to SD-Task
 DCOI-5.PERT-SD : Budget Breakdown of SD-Tasks }
6. Check "PERT/CPM future behaviour" against SD project outcome
 { Invoke links:
 DCOI-6.1.SD-PERT : Planned Project Resources Availability
 DCO-1.1 : Planned Start / Finish Dates of SD-Tasks
 DCO-2.1 : Planned Effort of SD-Tasks
 DCO-4.1 : Planned Scope of SD-Tasks
 DCO-3.1 : Planned Resource Allocation to SD-Tasks }
7. Check PERT/CPM plan against SD structure (dependencies)
 { Invoke links:
 DSC-1 : check consistency of intra-task SD-Dependencies
 DSC-2 : check consistency of inter-task SD-Dependencies
 Note: the modeller may choose not to invoke these links }
8. IF consistency-checking is satisfactory END
 { If any of the consistency links in steps 4 to 7 is not violated }
9. Diagnose inconsistencies
 { Check which consistency links and respective patterns are violated }
10. Re-adjust / develop from scratch manually the PERT/CPM future plan
 { Change the input data and structure of the PERT/CPM network, which are likely to eliminate the inconsistencies above. }
11. Call activity (P5) to update structural links as necessary
12. GOTO 4

The transfer of the data from the SD model to the PERT/CPM model is limited to the project start date (in case the project has not started yet) and to the profiles of resources availability to the project. As discussed when these links were specified, this is the only data that stands at the same level of aggregation in the two models. Therefore, no update is made to the data-fields in the individual PERT/CPM tasks and no readjustment is made to the logical network.

In step (4), the future behaviour portrayed by the PERT/CPM plan is extracted from the PERT/CPM model. This process was already described in activity (P3a). It should be noted that this behaviour patterns are extracted from the future segment of the current PERT/CPM plan. However, if we are at the beginning of the project it is the initial PERT/CPM plan that is being developed or readjusted. As previously mentioned, at the beginning of the project the initial plan and the current plan are the same.

In steps (5), (6) and (7) the PERT/CPM plan is checked for consistency against the plan in the SD model. If we are at the beginning of the project, then the targets of the PERT/CPM plan are checked against the targets of the initial plan in the SD model. The behaviour extracted from the PERT/CPM plan is then checked for consistency against the future behaviour produced by the SD model – it is assumed that a simulation of the SD plan is available from activity (P2b). Finally, the PERT/CPM plan is checked for consistency against the SD-Dependencies in the SD model – unlike in steps (5) and (5), while this consistency can be a good indicator of whether the PERT/CPM plan is representing well the SD plan, this is not mandatory.

In most cases the PERT/CPM plan will not be consistent with the SD plan and hence it needs to be readjusted manually by the project manager. While this readjustment is not carried out in an automated manner by the SYDPIM links, the inconsistencies identified in the previous steps are good indicators of what needs to be changed in the PERT/CPM plan. Just as two different project managers would develop different PERT/CPM networks to represent a same project, various solutions are possible to achieve consistency with the SD model. The principle behind SYDPIM is that if all the consistency links specified in steps (5), (6) and (7) are verified, then most likely the PERT/CPM plan provides an acceptable representation of the SD plan.

Once the PERT/CPM plan is modified manually, the structural links are updated as necessary – new tasks, resources or dependencies may have been added or removed and need to be remapped to the SD model. The process then cycles back to extracting the new project behaviour portrayed by the PERT/CPM plan and repeats until a satisfactory plan consistent with the plan in the SD model is achieved. Figure 7.21 shows the SYDPIM planning process after the

implementation of this activity, identifying the data exchange links used. This activity closes the re-planning cycle of the SYDPIM process logic, which was initiated with the PERT/CPM analysis and re-planning of the project future (activities (P1a) and (P2a)).

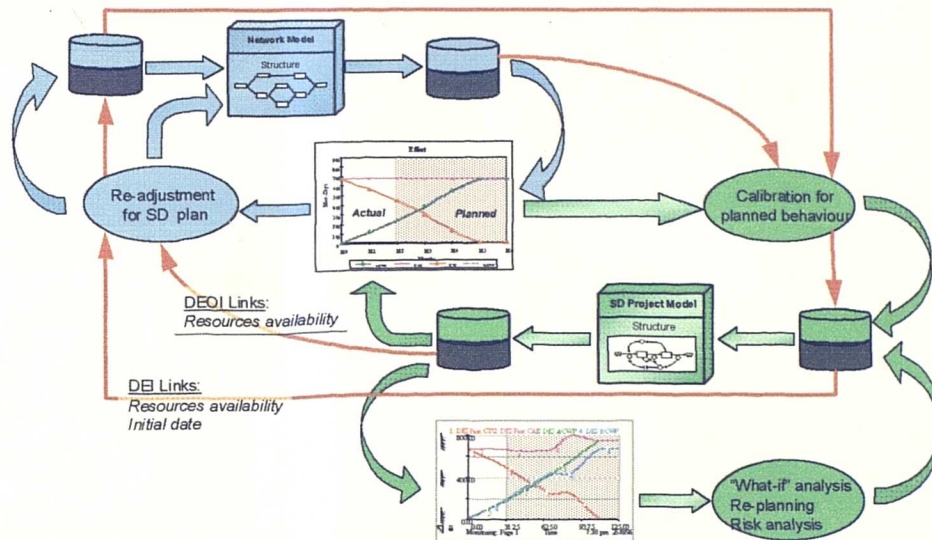


Figure 7.21 – Activity (P3b): readjusting the PERT/CPM model to the new SD plan

The next step in the SYDPIM planning process logic is to analyse the project future under a critical path perspective, in the PERT/CPM model, to verify whether it is satisfactory. This takes place in activity (P1a) already described and closes the loop of the SYDPIM planning process logic. If the plan is satisfactory then it can be implemented. Otherwise re-planning can take place once again, either in the PERT/CPM model (activity (P2a)) or in the SD model (activity (P2b)). If a final plan is ready to be implemented, then the present value of the various project target metrics needs to be updated in the SYDPIM objects. This is done in the final planning activity (P6).

(P6) Update target metrics – SYDPIM feature

This final activity in SYDPIM planning consists simply in an update of the present value of the target metrics which are eventually being recorded in the following SYDPIM objects: SYMDB, “PERT/CPM past behaviour” and “Project past

behaviour". The SYMDB will be updated if this database is in use. The other two objects will be updated in case they are being "preserved" throughout the project (as opposed to being re-generated whenever monitoring takes place).

The present value of the target metrics depends on both actual results and on the future plan. In SYDPIM, the update of the various metrics in the objects, in both monitoring and (re)planning, takes place in the same time-point (i.e. the present moment). Therefore, target metrics may have two different values in the present moment: one at the end of monitoring (when the plan is updated with actual results), and the other at the end of planning (when the new plan produced). For example, in the plan of last month the design phase may have been scheduled to finish in day 100. However, given the current progress updated in monitoring, the new indicated completion date may now be day 110 (i.e. the critical path shifted 10 days to the right). This new value is based on actual results and assumes that the plan for the remaining of the project is kept as per last month. The SYDPIM process then moves to planning, where control decisions are implemented and a new plan is produced. This new plan may now indicate a completion date of day 105 (the critical path was "crashed" 5 days). This new present (and final) value of this target metric is updated at the end of planning, in this activity (P6). This applies to all target metrics considered.

For the purpose of updating the target metrics, two elements need to be specified in this SYDPIM activity: the specific metrics to be updated and the calculation process required. The target metrics are common to the three objects and are identified in table 7.14. The second column identifies the condition for the metric to be updated – for example, the completion date of a SD-Task already completed does not need to be updated because it is not affected by changes in the future plan.

The object variables that correspond to each of these metrics in the objects SYMDB, "PERT/CPM past behaviour" and "Project past behaviour" are straightforward to identify. For example, for the project SAC, the variable in each object is as follows:

- SYMDB.PERT/CPM derived metrics.Schedule.Project schedule.Finish date[present]
- PERT/CPM Past behaviour.Schedule.Project schedule.Finish date[present]
- Project past behaviour.Schedule.Project schedule.Finish date[present]

Target metric	Condition
(1) Schedules	
(1.1) Planned completion dates (SAC)	For the whole project and for each SD-Task not completed
(1.2) Planned start date	For each SD-Task not started yet
(2) Effort	
(2.1) Cost at completion (CAC)	For the whole project and for each SD-Task not completed
(2.2) Cost to complete (CTC)	For the whole project and for each SD-Task not completed
(3) Scope	
(3.1) Scope at completion (SCAC)	For the whole project and for each SD-Task not completed
(3.2) Scope to complete (SCTC)	For the whole project and for each SD-Task not completed
(4) Resources	
(4.2) Cum. staff profile at completion (CSPAC)	For the whole project and for each SD-Task not completed

Table 7.14 – Target metrics to be updated in planning activity (P6)

The calculation process required for these metrics is the same as the one use to derive the steady patterns of these variables in the “PERT/CPM future behaviour” – see description of activity (P3a); tables 7.8 through 7.11. These calculations are the same for the three objects (since the metrics are the same), and assume the availability of a current PERT/CPM plan, with a future segment just updated in planning.

This activity consists of a simple metrics update process. Once this has been done in the three objects, the future plan is used to guide implementation.

The branches of the SYDPIM planning process logic

The SYDPIM planning process logic can be entered in three different points: from monitoring, development of initial plan in the PERT/CPM model and development of initial plan in the SD model. Once entered, it can follow various paths depending on the preferences of the project manager in using the models for re-planning. Once the project future is forecasted in both models, the project manager may perceived this either as satisfactory or unsatisfactory. If unsatisfactory, the project manager can either follow the path of re-planning in the SD model and then transfer the SD plan to the PERT/CPM model, or the path re-planning in the PERT/CPM model and

test this plan in the SD model. This generates two decision-points within the process logic, which are shown in yellow boxes with the following questions:

(Q1) : "Is outcome satisfactory and plans consistent?"

(Q2) : "Re-plan in SD model or PERT/CPM model?"

Regarding the second question, SYDPIM recommends that if the process was just entered from monitoring then re-planning should be implemented in the SD model. There are two main reasons for this: (i) it is generally a good idea to change a project plan first at higher level of aggregation and only then refine into a more detailed level; and, more important, (ii) the SD model is conceptually a richer model, incorporating a wider range of factors and giving a strong importance to the project past (e.g. amount of undiscovered defects). The SD model therefore provides a more robust and realistic plan than the PERT/CPM model. However, (Q2) can be reached after the project has been re-planned in the SD model and then still found unsatisfactory in the PERT/CPM model perhaps due to detailed critical path and resources issues. In this case, the required changes in the project plan do not tend to be of great magnitude and should be implemented first in the PERT/CPM model and then transferred to the SD model. But if the required changes are perceived of great magnitude then the SD model should be used first once more.

Regarding the first question, there are two separate issues: the outcome being satisfactory and the plans in the two models being consistent. The first issue results from comparing the forecasted outcome to the project objectives. It is entirely of the Project Manager's responsibility to decide whether the outcome is "good enough" or not. The second issue is very important in terms of SYDPIM process. It results from the fact that the SYDPIM planning process logic should only be exited if the project plans in the two model are the same. If (Q2) is entered after re-planning has been carried out in one of the models then the plans will always be consistent one another – they were readjusted in activities (P3b) or (P4a). Otherwise, the question is entered just after SYDPIM planning has been entered from monitoring. In this case, the project future forecasted by both models may be different and consistency needs to be verified. This condition has one immediate implication which is that the future project behaviour produced by the SD model *must be steady*. This is because a PERT/CPM plan, by nature, always portrays a steady behaviour. Once the SD behaviour is steady, the next step is to check whether it represents the same

future plan as the PERT/CPM model. In order to do this, two conditions must be verified: (i) the future project behaviours produced by both models must be the same, and (ii) and this has to be for the right reasons. In order to verify these two conditions the SYDPIM consistency links are invoked. This consistency-checking process is similar to the one in activities (P3b) and (P4a). The links help to ensure that these two conditions are verified but should not be considered as sufficient, and human judgement is required. The consistency links should be invoked as described in the following algorithm:

Algorithm: { Check if project future plans are consistent in both models }

1. Check if initial plan is consistent
{ Invoke following links:
 - DCI-1 : Profiles of Project Resources Availability
 - DCI-2 : Project Start Date
 - DCOI-1.PERT-SD : Start and Finish Dates of SD-Tasks
 - DCOI-2.PERT-SD : Budget of SD-Tasks
 - DCOI-3.PERT-SD : Scope of SD-Tasks
 - DCOI-4.PERT-SD : Profiles of Resource Allocation to SD-Task
 - DCOI-5.PERT-SD : Budget Breakdown of SD-TasksIf any of these links fail then probably the initial plan has been changed during the update of the models in monitoring – this should not happen. }
2. Check SD-Dependencies of future plan
{ Invoke links:
 - DSC-1 : check consistency of intra-task SD-Dependencies
 - DSC-2 : check consistency of inter-task SD-DependenciesNote: the modeller may choose not to invoke these links
If any of these links fail then the SD dependencies of the future plan are not consistent with the planned scope progress in the PERT/CPM future plan. This is indicative of some possible mismatch between the two plans but is not a necessary conditions to fail the consistency. }
3. Extract "PERT/CPM future" behaviour
{ This process is the same as the one described in activity (P3a). }
4. Check that future behaviours are "close enough"
{ Invoke links:
 - DCOI-6.1.SD-PERT : Planned Project Resources Availability
 - DCO-1.1 : Planned Start / Finish Dates of SD-Tasks
 - DCO-2.1 : Planned Effort of SD-Tasks
 - DCO-4.1 : Planned Scope of SD-Tasks
 - DCO-3.1 : Planned Resource Allocation to SD-Tasks }If any of these links fail then the forecasted behaviours of the models are not "close enough" one another, indicating that the future plans are not the same. }

In order to consistency-check the future project behaviour produced by the two models, it is necessary to extract the "PERT/CPM future behaviour" from the PERT/CPM model. This is done in step (3).

If this consistency-checking fails then the two models are not representing the same future plan and hence SYDPIM planning should not be exited – it does not make sense to exit with two different plans, one in each model. The process should proceed to activities (2a) or (2b) in order to readjust one of the plans.

The SYDPM project plan: an integrated view

Once the SYDPIM planning cycle is complete, the next step is to implement the improved project plan. This “project plan” includes not only what is specified in the PERT/CPM network, but also other type of information and decisions specified in the SD plan. For example, in a software project the SD analysis may have suggested that, even in case of delays, unskilled staff should not be hired until the coding phase is complete (perhaps to prevent the damages of low quality components being delivered to integration). This is a planning “control policy”, which should be implemented along with what is specified in the PERT/CPM network. While some of the planning information is present in both models (e.g. project schedules), the SD model also adds new information to the project plan which not considered within the scope of the PERT/CPM model.

The SYDPIM methodology proposes the use of an integrated project plan combining the planning information contained in both models. The concept of the SYDPIM integrated plan is represented in figure 7.22. The “SD component” of the project plan contains mostly strategic information, like the management control policies, and a high level specification of the schedules, budgets, scope and resources. The “PERT/CPM component” contains operational information, specifying in great detail the scheduling of work tasks, resources and scope. The PERT/CPM component also contains information not considered in the SD plan, like the elementary precedence relationships between the detailed tasks.

Once the SYDPIM project plan has been implemented, the next step is to collect result metrics, monitor progress, assess performance and identify deviations. This is done in SYDPIM monitoring.

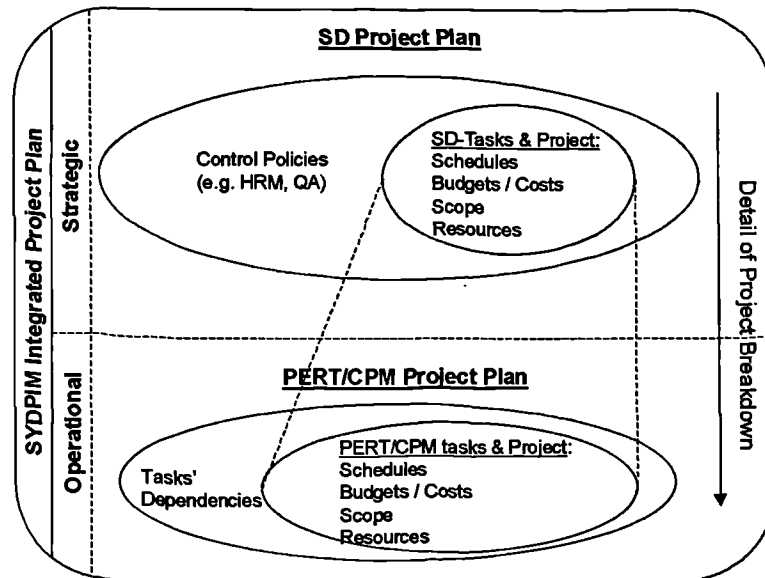


Figure 7.22 – The SYDPIM Project Plan: an integrated view

SYDPIM monitoring

Project control consists of a closed loop of re-planning, implementation of the plan and monitoring of the project status. The only entry-point of the SYDPIM monitoring process logic is after the implementation of the project plan – see figure 7.6. The SYDPIM activities have been numbered sequentially from this point. There are various possible paths within the process logic, which reflect different scenarios. First, it is considered that there might not be a PERT/CPM current plan available. For example, this can happen in a less structured management environment, where SYDPIM is going to be implemented in a project half-way through its life-cycle. The best solution would be to update the PERT/CPM model for the whole past that remains to be updated. However, even if this is not possible, it is fundamental to calibrate the SD model for the project past. The second branch considers that, in addition to the PERT/CPM model not being updated, the SYDPIM metrics plan (SYMP) and database (SYMDB) might not be in use. This can happen for various reasons, and it implies that the monitoring metrics proposed by SYMP are not being collected. In case the PERT/CPM model is being updated every control cycle, the SYMDB might be in use, or not. For the sake of simplicity, this branch is not considered explicitly in the SYDPIM monitoring process logic shown in figure 7.6, but it will be considered in the description of the individual activities.

The recommended path in SYDPIM monitoring is based on the principle that a PERT/CPM model is being updated every control cycle and that the SYMDB is being implemented. The role of the SD model in monitoring is to diagnose the project past, to explain why it has evolved as observed, and to uncover further information about the project status in the form of quantitative metrics – e.g. current amount of undetected defects. This scenario leads to the following sequence of activities:

- (M1a) collect project result metrics – conventional feature and SYDPIM feature;
- (M2a) update PERT/CPM model and SYMDB with result metrics – conventional and SYDPIM feature;
- (M3a) update analytical links if necessary – SYDPIM feature;
- (M4) extract project past behaviour from PERT/CPM model – SYDPIM feature;
- (M5) specify project past behaviour – SYDPIM feature;
- (M6) re-calibrate SD model for past behaviour – SYDPIM feature;
- (M7) SD diagnosis of project past – SYDPIM feature;
- (M8) update SYMDB with uncovered SD metrics – SYDPIM feature.

It is important to note that activity (M3a), where the analytical links are updated, should be considered as implemented in parallel with the activity (M2a). This is because of two reasons: the update of the PERT/CPM model may imply structural changes to the current PERT/CPM plan, and the update of the SYMDB requires the use of the analytical links. So activity (M3a) is “called” within (M2a), while the PERT/CPM model is being updated and if necessary.

Out of the eight activities only the first two are a conventional feature. They consist of collecting data and update the PERT/CPM plan with actual results: resource allocation and availability, schedules, costs and scope. Once this is done, the traditional process follows immediately to the critical-path analysis of the project past in the re-planning function. In SYDPIM a series of other monitoring steps are followed, leading to the SD diagnosis and uncovering of project status – this is the main added value of SYDPIM; it should be noted that the diagnosis of the project past is based on retrospective “what-if” analyses, which can feed important information into process improvement activities within the organisation.

In case a current PERT/CPM plan is not available but SYMDB is being implemented, then the SYDPIM sequence of activities is as follows:

- (M1b) collected project result metrics – SYDPIM feature;
- (M2b) update SYMDB with result metrics – SYDPIM feature;
- (M5) specify project past behaviour – SYDPIM feature;
- (M6) re-calibrate SD model for past behaviour – SYDPIM feature;
- (M7) SD diagnosis of project past – SYDPIM feature;
- (M8) update SYMDB with uncovered SD metrics – SYDPIM feature.

As it will be seen, the set of metrics to collect in activity (M1b) differs from the one in activity (M1a) – in (M1b) only the metrics specified in SYMP are being collected. The process then jumps into activity (M5), skipping the steps which required the PERT/CPM plan. Implemented in this way, the monitoring process becomes a “stand-alone” mode of the use of the SD model. Whenever monitoring is implemented in this way, there is no connection with the PERT/CPM model.

In case the SYMDB is not being implemented, the SYDPIM sequence of activities is as follows:

- (M1c) collect information from expert judgement;
- (M5) specify project past behaviour – SYDPIM feature;
- (M6) re-calibrate SD model for past behaviour – SYDPIM feature;
- (M7) SD diagnosis of project past – SYDPIM feature.

In this scenario, no quantitative metrics are collected to help deriving the project past behaviour to be reproduced by the SD model. Instead, management expert judgement is used to derive these patterns. All that is related with the PERT/CPM model is skipped. Since SYMDB is not being implemented, the extra status metrics uncovered by the SD model are not stored in the database. This is the lead preferred scenario, since the derived project past behaviour is likely to be less accurate and thus all the subsequent SD analysis. However, SYDPIM is also intended to be flexible and thus it provides the project manager with an answer and alternative routes of action when necessary.

Each of the SYDPIM activities is now described separately in more detail.

(M1a) collect project result metrics – conventional and SYDPIM feature

This activity assumes that there is a current PERT/CPM plan available and that the SYMDB is being implemented. The purpose of this activity is to collect the necessary progress data so that the PERT/CPM current plan and the SYMDB are updated with actual results in the following activity.

In practice, the data to be collected and updated in a PERT/CPM plan depends on how the PERT/CPM model is used within the organisation. In SYDPIM it is assumed that certain tasks' data-fields will always be updated in the PERT/CPM model – see object specification.

Metrics to be updated in the SYMDB are only collected if the database is in use. Otherwise the collection of these metrics is skipped in this activity. The specific metrics to be updated in the SYMDB were identified in the formal object specification. Some of these metrics will not be collected directly from the project in this activity, as they can be derived from other metrics or from data in the current PERT/CPM plan.

Table 7.15 shows the result metrics to be collected in this SYDPIM activity. The PERT/CPM metrics refer to the data-fields in the current plan which store actual data. Most of these fields are task specific and thus these metrics will be collected for each task in the PERT/CPM current plan. This includes new tasks created during implementation, which will be added to the current plan – however, in general tasks should be preferably added to the plan in the re-planning function. The second column identifies the necessary condition for the metric to be collected. For example, the actual start date of task should only be updated when the work in the task was already started. The three metrics (2.1), (2.2) and (2.3) are inter-related and one can be derived or must be consistent with the other two. Collecting these metrics to update the PERT/CPM plan is part of the conventional project control framework.

The SYMDB metrics refer to defect detection and rework. These metrics are collected for each SD-Task considered in the SD model. The reason why these metrics are collected at this stage is that they cannot be derived from a PERT/CPM

plan. The standard use of the PERT/CPM model assumed in SYDPIM does not consider this type of data. Again, these metrics are only collected if the SYMDB is in use.

Metric	Condition
PERT/CPM metrics	
(1) Project	
(1.1) initial date	If project started
(1.2) resource availability	If project started
(2) Tasks	
(2.1) actual start date	If task started
(2.2) actual completion date	If task finished
(2.3) actual duration	If task finished = (2.2) - (2.1)
(2.4) actual effort spent to date	If task started
(2.5) actual scope to date	If task started
(2.6) actual resource allocation	If task started
SYMDB metrics	
(3) SD-Tasks	
(3.1) defects detected	If SD-Task started
(3.2) defects reworked	
(3.3) cumulative defects detected	
(3.4) cumulative defects reworked	
(3.5) defects awaiting rework	

Table 7.15 – Metrics to be collected in SYDPIM activity (M1a)

When this type of metrics is collected from a project, these are typically recorded in some kind of periodical metrics report. The next step is to update the PERT/CPM current plan and the SYMDB. This takes place in the following activity (M2a).

(M2a) update PERT/CPM model and SYMDB with result metrics – conventional and SYDPIM feature:

This activity updates the PERT/CPM model with the result metrics collected in the previous activity, just as in the conventional framework, and also updates the SYMDB with further result metrics (a SYDPIM feature).

The SYMDB is only updated in this activity if this database is in use. The required metrics are generated in four different ways:

- (a) collected – directly from the project, in the previous activity;
- (b) derived from PERT/CPM model – from metrics stored in the PERT/CPM model;
- (c) calculated – from other metrics already stored in the SYMDB database;

(d) uncovered – from the SD model, once calibrated to reproduce the project past.

This differentiation is emphasised in the formal specification of the database – see SYMDB object specification. In this SYDPIM activity, only the first three types of metrics are updated – uncovered metrics are updated in activity (M8).

The process of updating the PERT/CPM model and the SYMDB follows a logical sequence of three main steps:

- (1) update the PERT/CPM current plan and SYMDB with the result metrics collected in activity (M1a);
- (2) update the SYMDB with metrics derived from the PERT/CPM model;
- (3) update the SYMDB with metrics calculated from other metrics already stored in the database.

The second part of step (1) and steps (2) and (3) are only implemented if the SYMDB is in use. Otherwise, monitoring follows to the next activity. In order to implement these steps, there are two elements that need to be formally specified in this activity: (i) the object variables to be updated with the metrics, and (ii) any calculations required to generate the metrics.

The first step consists of feeding directly the result metrics collected in the previous activity (M1a), onto the PERT/CPM current plan and SYMDB. This process does not require any calculation. These metrics were identified in table 7.15 (data-fields of the current PERT/CPM plan and defect metrics in SYMDB), and the correspondent object variables to be updated in this activity are identified in table 7.16.

In the PERT/CPM model the result metrics are stored in the data-fields of the current PERT/CPM plan. Most of these are specified at the task level.

In the SYMDB, the metrics are specified at both project level and at the SD-Task level, and are stored as over-time data-points. Whenever the database is updated, the present data-point of each metric is updated. In the object specification above, the “[i]” entry refers to the specific task and the entry “[present]” refers to the present data-point.

Metric	Object variable
Update of PERT/CPM model	
(1) Project	
(1.1) initial date	<u>PERT/CPM model</u> .Initial date
(1.2) resource availability	<u>PERT/CPM model</u> .Current PERT/CPM plan.Past segment.Actual_Resource-availability[SD-Resource]
(2) Tasks	
(2.1) actual start date	<u>PERT/CPM model</u> .Current PERT/CPM plan.Past segment.Tasks[i].Actual start date
(2.2) actual completion date	<u>PERT/CPM model</u> .Current PERT/CPM plan.Past segment.Tasks[i].Actual completion date
(2.3) actual duration	<u>PERT/CPM model</u> .Current PERT/CPM plan.Past segment.Tasks[i].Actual duration
(2.4) actual effort spent to date	<u>PERT/CPM model</u> .Current PERT/CPM plan.Past segment.Tasks[i].Actual effort spent to date
(2.5) actual scope to date	<u>PERT/CPM model</u> .Current PERT/CPM plan.Past segment.Tasks[i].Actual scope to date
(2.6) actual resource allocation	<u>PERT/CPM model</u> .Current PERT/CPM plan.Past segment.Tasks[i].Actual resource allocation
Update of SYMDB Collected metrics	
(3) SD-Tasks	
(3.1) defects detected	<u>SYMDB</u> .Collected metrics.Quality.Task Quality.Defects detected[i][present]
(3.2) defects reworked	<u>SYMDB</u> .Collected metrics.Quality.Task Quality.Defects detected[i][present]
(3.3) cumulative defects detected	<u>SYMDB</u> .Collected metrics.Quality.Task Quality.Cum defects detected[i][present]
(3.4) cumulative defects reworked	<u>SYMDB</u> .Collected metrics.Quality.Task Quality.Cum defects reworked[i][present]
(3.5) defects awaiting rework	<u>SYMDB</u> .Collected metrics.Quality.Task Quality.Defects awaiting rework[i][present]

Table 7.16 – PERT/CPM and SYMDB object variables to be updated in SYDPIM activity (M2a) from data collected in activity (M1a)

At this stage, and before moving onto the second step, if any structural changes occurred in the current PERT/CPM plan while being updated, then activity (M3a)

must be called to update the analytical links. By structural changes it is meant any adding or removal of tasks, resources and dependencies from the network.

Once the analytical links have been updated in activity (M3a) as required, the second step consists of updating the SYMDB with metrics derived from the PERT/CPM model. This process requires calculations and consists mainly in aggregating the result metrics just entered in the PERT/CPM model to the SD level in the SYMDB. The result metrics to be derived from the PERT/CPM model refer schedules, effort, scope and resources. The correspondent SYMDB object variables to be updated are identified in the table 7.17. Since there is a large number of metrics, for the sake of simplicity these were summarised just to show how the object variables can be identified – see SYMDB object specification for the full set. Again, the “[i]” entry specifies the SD-Task and “[present]” specifies the present moment. The calculation processes for all variables are shown in tables 7.18 through 7.22 (appendix B). The second column explains informally the metric value is generated from the data stored in the PERT/CPM model. Since in many cases this will be an aggregation process (tasks to SD-Tasks; resources to SD-Resources), the SYDPIM analytical links are required to implement the mapping – as shown in see diagram of figure 7.6, which describes the SYDPIM monitoring process logic. The third column identifies the SYDPIM elements that will be required to implement the calculation process. This includes the analytical links and the PERT/CPM plans within the PERT/CPM model. Most of the metrics refer to past occurrences and so they are generated solely from data in the past segment of the current PERT/CPM plan. However, other metrics refer to current targets for the project (e.g. estimated cost at completion of on-going tasks), and as such their value depends upon the future plan – marked with (*). Therefore, the process of calculating the value of these target metrics also requires the future segment of the current PERT/CPM plan. Finally, some metrics reflect what was specified in the initial plan and so they required the initial plan.

The third and final step in this activity is to update metrics in the SYMDB from other metrics just entered in the database. There are three types of metrics in this situation:

- (1) project-wide results – like total cost spent in the project;
- (2) performance indices – like the *earned value*;

(3) process metrics – like the productivity and the defect rates.

Project-wide results can be derived from the PERT/CPM model, just as the results at the SD-Task level were in the previous step. However, the process would be more laborious and for most of them it is possible and easier to derive their values from the SD-Task level results. For example, the cost spent to date in the project (ACWP) is the same as the sum of the ACWP of all SD-Tasks. If derived from the PERT/CPM model it would be the sum of the effort spent to date in all tasks in the PERT/CPM current plan. However, not all project-wide results can be derived from the SD-Tasks results. This is the case of the results which refer to project resources availability, and were updated in the previous step as shown in table 7.18. Table 7.23 (appendix B) identifies the project results that can be derived from the SD-Task level results already updated in the database. The second column identifies the SYMDB object variables and the third column describes informally the calculation process – except for the completion date, this is simply the sum of the results at the SD-Task level. The project-wide defect metrics could not have been derived from the PERT/CPM model but could have also been derived from the metrics collected in the first step.

The performance indices and the process metrics are shown in tables 7.24 and 7.25 respectively (appendix B). In both cases, these metrics are calculated at both SD-Task and project levels. The object variables are identified in the second column and the calculation process is described in the third column. The variables used in the proposed formulas refer to the SD-Task and project results stored in the database, as specified in tables 7.18 through 7.23. For example, the index CPI (Cost Performance Index) at the project level is calculated from the ACWP and BCWP metrics at the project level specified in table 7.20.

An important issue in this activity is the update of the present data-points of some metrics in the SYMDB, which may need to be updated later at the end of re-planning. These particular metrics refer to project *targets* which depend not only on what just happened in the project (i.e. the recent past), but also on the project future plan. For example, the cost at completion (CAC) is equals what was actually spent in the project so far plus what is the planned remaining expenditure. Once SYDPIM monitoring is entered from implementation and actual results are updated in

SYMDB, these target metrics will be calculated based on the existing current PERT/CPM plan. Therefore, they will reflect what is likely to happen given the recent past and if no re-planning takes place. For example, say the CAC in the previous control cycle was 100 person-month. During the last period, there was an excessive expenditure of 5 person-month. Once the SYMDB is updated, the new CAC will be 105 person-month the project manager may have found solutions to reduce the CAC down to 102 person-month. So the value of the present data-point of this metric may change from 105 in monitoring to 102 at the end of re-planning. This change takes place within the present moment, because for the purpose of updating the SYMDB the time elapsed between monitoring and planning is not considered. The target variables that may need to be updated at the end of planning in activity (P6) are marked with a (*) in tables 7.18 through 7.22

Another important issue in this activity is that the analytical links may need to be updated. If necessary, activity (M3a) is called for this purpose

As shown in figure 7.23 below at the end of this activity both the PERT/CPM model and the SYMDB have been updated with actual results. Some of the metrics stored in the SYMDB are derived from the PERT/CPM model.

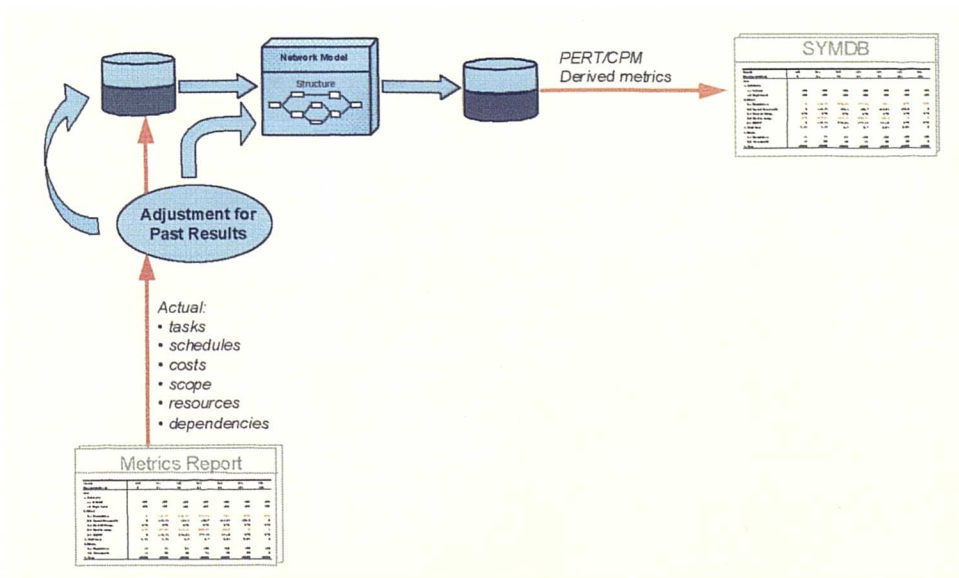


Figure 7.23 – Activity (M2a): update of PERT/CPM model and SYMDB with actual results

(M3a) update analytical links if necessary – SYDPIM feature

This activity is entered when structural changes took place in the current PERT/CPM plan while being updated with actual results. Newly added or removed tasks, dependencies and resources may require some re-mapping to the SD model.

The process within this activity is similar to activity (P5), in planning. The links that may need to be updated are the structural links. There are two types of structural links: structural correspondence links and structural consistency. In this activity, the links of structural correspondence are updated and the links of structural consistency are invoked to check whether there is any inconsistency.

The first step is to update the links of structural correspondence. As in activity (P5) This should be done in the following sequence:

- (1) SC-WBS – PERT/CPM tasks are re-mapped to the SD-Tasks, as necessary;
- (2) SC-OBS – PERT/CPM resources re-mapped to the SD-Resources, as necessary;
- (3) SC-WD – PERT/CPM dependencies re-mapped to SD-Dependencies (both intra- and inter-task). The two algorithms proposed, SC-WD-1 and SC-WD-2, are available to automatically generate this mapping, which can then be readjusted manually by the project manager;

How does the project manager know that an update is required? First, at least one of the validity conditions specified for each of these three links (see formal specification) will be violated. For example, if a newly added PERT/CPM task has not been mapped to any SD-Task, then the validity condition PERT/CPM mapping of the SC-WBS link will be violated. Another possibility where the update of the structural links is required is when these have become inconsistent one another. For this purpose, the structural consistency links are invoked:

- (4) SCN-RA – checks whether the SC-WBS and SC-OBS links are consistent one another regarding resource allocation. The algorithm proposed in the specification of this link, SCN-RA-1, is available to automatically check this consistency and proposes possible solution to eliminate the inconsistency;
- (5) SCN-WD – checks whether the SC-WBS and the SC-WD links are consistent one another. Again, two algorithms are available: SCN-WD-1 and SCN-WD-2,

which describe an automated process to check this consistency for intra-task and inter-task SD-Dependencies, respectively. In case inconsistencies are detected, both algorithms suggest possible solutions.

In case some inconsistencies are detected, corrections to the links of structural correspondence should be implemented as appropriate. Once the analytical links have been updated as required, the next activity is to extract the project past behaviour from the PERT/CPM model.

(M4) extract project past behaviour from PERT/CPM model – SYDPIM feature

Once actual results have been collected from the project and updated in the PERT/CPM model and SYMDB, the following steps are aimed at allowing the project past behaviour to be diagnosed in the SD model. The project past behaviour will be derived from the PERT/CPM model and from the SYMDB if this database is in use.

In case the SYMDB is being implemented, all the patterns of past behaviour that can be extracted from the PERT/CPM model can also be derived from the metrics stored in this database – in fact, some of these metrics were already derived from the PERT/CPM model. So in this case, the SYMDB could be used as the only source of information to specify the project past behaviour to be reproduced by the SD model in activity (M7). However, the calibration of the SD model in this activity is based on the analytical links established with the PERT/CPM model and this requires that an updated version of the “PERT/CPM past behaviour” object is available. Therefore, it is recommended in SYDPIM that this object is always updated in this activity (M4).

The specific set of patterns to be extracted is identified in the formal specification of the SYDPIM object “PERT/CPM past behaviour”, where the patterns will be stored. These patterns are the same as the ones specified for the “PERT/CPM future behaviour”, which is updated in activity (P3) already described. Like with the future behaviour, the technical algorithms required for the past behaviour patterns do not present any major difficulty. However, the calculation process is slightly more complex than with the future behaviour, in particular regarding those patterns that refer to project targets.

Ideally, in SYDPIM the SD model is calibrated to reproduce the whole project past progressively, as re-calibration for the last period takes place in activity (M7) of every control cycle. This way, the PERT/CPM past behaviour can also be specified progressively. In this scenario, only the control period just elapsed needs to be specified in the present activity (M4) (i.e. from the previous control point up to present). However, there can be situations where the PERT/CPM past behaviour has not been updated regularly or when SYDPIM is introduced in half-way in the project. In other cases, the user may wish not to preserve the “PERT/CPM past behaviour” object and so the whole past behaviour needs to be regenerated. The calculations presented in tables 7.26 through 7.29 refer to the generation of the patterns for the whole project past – table 7.26 is shown for illustrative purposes. From here, the calculation of the present data-point is straightforward.

The patterns are grouped per data type. In the second column the calculation process is described informally. In the last column the requirements to implement the calculations are identified. This includes the PERT/CPM plans required and the analytical links. Patterns that require the initial plan and other past versions of the PERT/CPM plan also require the correspondent version of the analytical links.

The “start date” is the only pattern that applies to the SD-Tasks only. All the other patterns apply to the SD-Tasks and to the whole project. The patterns that refer to resources are also calculated per SD resource category. The patterns for a SD-Task are extracted from the numerical inputs of all the individual PERT/CPM tasks which are mapped to it, according to the structural link SC-WBS. The patterns related to resources also require the SC-OBS links to identify the PERT/CPM resources mapped to each SD-Resource. The operators used are the same as in activity (P3a). The reasoning behind the calculation processes is not the same as for the “PERT/CPM future behaviour” for all patterns (specially for the target patterns), but the underlying principles are the same.

Regarding the patterns of project targets (e.g. completion date), these tend to change throughout the past segment of the project. To each value of the target corresponds an individual PERT/CPM plan. This is because a single PERT/CPM plan considers that the targets are fixed over-time, portraying a successful

implementation of the work. For example, the completion date is imposed by the critical path in the PERT/CPM network, which is not expected to change. Whenever the project targets change, the PERT/CPM network is modified and a new plan is produced. Therefore, patterns that describe project targets need to be extracted from the whole set of past PERT/CPM plans stored in the PERT/CPM object.

Non-target patterns (e.g. cumulative effort spent) are extracted from the current PERT/CPM plan. Patterns that describe what should have happened according to the initial project plan (e.g. BCWS) are extracted from the initial PERT/CPM plan.

Pattern	Calculation	Requirements
Schedule		
SAC[t] (*)	For the whole project and per SD-Task. SD-Task: = for each time-point MAX planned finishing date of all tasks in the PERT/CPM plan of that time-point mapped to the SD-Task. Project: = for each data-point Max SAC[t] of all SD-Tasks.	Set of past versions of PERT/CPM plan (incl. Initial plan) Current PERT/CPM plan SC-WBS link SC-WBS link (past ver.)
Start Date[t] (*)	For each SD-Task. SD-Task: = for each time-point MIN planned start date of all tasks in the PERT/CPM plan of that time-point mapped to the SD-Task.	Set of past versions of PERT/CPM plan (incl. Initial plan) Current PERT/CPM plan SC-WBS link SC-WBS link (past ver.)

Table 7.26 – Calculation and requirements of the schedule patterns of past behaviour extracted from the PERT/CPM model.

Again, if the user is preserving the “PERT/CPM past behaviour” object within the project control framework, the target variables marked with (*) must be updated at the end of planning.

Once generated, these patterns will be stored in the “PERT/CPM past behaviour”. The specific object variables are not presented in these tables, but their identification is straightforward. For example, the object variables to be updated with the pattern SAC[t], for each SD-Task and for the whole project, are as follows:

- PERT/CPM past behaviour.Schedule.Project schedule.Finish date[t]
- PERT/CPM past behaviour.Schedule.Task schedule.Finish date[i][t]

The next step in SYDPIM monitoring is to complement this set of patterns with other patterns derived from the SYMDB (if in use) and from expert judgement. The final

set of patterns will form “project past behaviour” which the SD model will have to reproduce in activity (M7). The specification of the “project past behaviour” is implemented in activity (M5). However, this activity can be entered from the current activity (M4) but also from other two alternative paths in the SYDPIM monitoring process logic: (i) when the PERT/CPM model has not been updated, and (ii) when in addition the SYMDB is also not in use. Therefore, before describing activity (M5), the activities of these other two alternative paths are now described.

Figure 7.24 below shows the state of the SYDPIM monitoring process after this activity has been implemented.

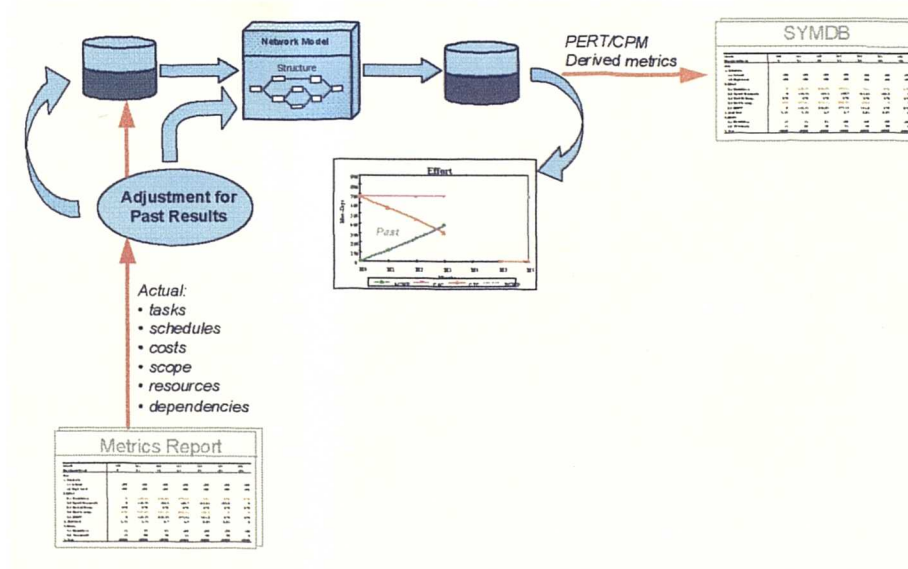


Figure 7.24 – Activity (M4): extraction of project past behaviour from PERT/CPM model

(M1b) collect project result metrics – SYDPIM feature

This activity is entered when the PERT/CPM model is not being updated regularly but SYMDB is being implemented. There is not current PERT/CPM plan available and so this will not be updated in monitoring. In this scenario the SYMDB will be used to derive the full project past behaviour in activity (M5).

Some of the metrics to be updated in the SYMDB would be derived from the PERT/CPM model. These metrics will now have to be collected directly from the

project. Also, metrics that would be collected at the PERT/CPM task level, will now be collected at the SD-Task level.

For the target-type of variables, the “collection” process involves some form of estimating. For example, collecting the metric ACWP instead of deriving it from the PERT/CPM model may simply consist of going around in the project and ask staff about their expenditures – in any case, a procedure similar to collecting this metrics at the PERT/CPM task level. However, for a target variable like CAC (Cost at Completion), the project manager needs, somehow, to estimate how much the SD-Task or project is likely to cost given the actual cost (ACWP) and the “future plan” (even if a PERT/CPM network is not available) – if a current PERT/CPM plan was available, the PERT/CPM model itself would provide this estimate, as in activity (M2a). Table 7.30 below presents the set of metrics to be collected in this activity. The second column identifies the condition for each metrics to be collected.

Metric	Condition
SYMDB Metrics	
(1) Project	
(1.1) Resources	If project started
ASP	
...	
CSPAC	
(2) SD-Tasks	
(2.1) Schedule	
Start date	If SD-Task started
Finish date	If SD-Task finished
(2.2) Effort	
ACWP	If SD-Task started
...	
CAC	
(2.3) Scope	
SCAC	If SD-Task started
...	
SCTC	
(2.4) Resources	
ASP	If SD-Task started
...	
CSPAC	
(2.5) Quality	
Defects detected	If SD-Task started
Defects reworked	
Cumulative defects detected	
Cumulative defects reworked	
Defects awaiting rework	

Table 7.30 – Metrics to be collected in SYDPIM activity (M1b)

Basically, the profiles of project resources availability are collected together with the schedule, effort, scope, and resource allocation metrics (all of which would be derived from the PERT/CPM model), and also the quality metrics, for each SD-Task. Once collected, the next step is to update the SYMDB in activity (M2b).

(M2b) update SYMDB with result metrics – SYDPIM feature

This activity is similar to activity (M2a) except that no metrics will be derived from the PERT/CPM model to update the SYMDB. The steps to be taken in this activity are as follows:

- (1) update the SYMDB with the result metrics collected in activity (M1b);
- (2) update the SYMDB with metrics calculated from other metrics already stored in the database.

The first step does not involve any calculation and consists simply in storing the metrics collected in the previous activity into the appropriate object variables of the SYMDB. Table 7.31 identifies these variables.

The first column identifies the same metrics as in table 7.30 and the second column identifies the formal specification of SYMDB object variables.

The second step is similar to step (3) in activity (M2a). Metrics in the SYMDB will be derived from the metrics just entered in the previous step. Again, there are three types of metrics in this situation:

- (1) project-wide results – like total cost spent in the project;
- (2) performance indices – like the *earned value*;
- (3) process metrics – like the productivity and the defect rates.

In order to update these metrics it is necessary to know the object variables in the SYMDB object and the calculation processes. Both are exactly the same as for activity (M2a), and were already described in tables 7.23 and 7.25.

Metric	Object variable
Update of SYMDB Collected metrics	
(1) Project	
(1.1) Resources	
ASP	SYMDB.PERT/CPM derived metrics.Resources. Project Resources.ASP[present]
...	...
CSPAC (*)	SYMDB.PERT/CPM derived metrics.Resources. Project Resources.CSPAC[present]
(2) SD-Tasks	
(2.1) Schedule	
Start date (*)	SYMDB.PERT/CPM derived metrics.Schedule. Task Schedule.Start date[i][present]
Finish date (*)	SYMDB.PERT/CPM derived metrics.Schedule. Task Schedule.Finish date[i][present]
(2.2) Effort	
ACWP	SYMDB.PERT/CPM derived metrics.Effort. Task Effort.ACWP[i][present]
...	...
CTC (*)	SYMDB.PERT/CPM derived metrics.Effort. Task Effort.CTC[i][present]
CAC (*)	SYMDB.PERT/CPM derived metrics.Effort. Task Effort.CAC[i][present]
(2.3) Scope	
SCAC (*)	SYMDB.PERT/CPM derived metrics.Scope. Task Scope.SCAC[i][present]
...	...
SCTC (*)	SYMDB.PERT/CPM derived metrics.Scope. Task Scope.SCTC[i][present]
(2.4) Resources	
ASP	SYMDB.PERT/CPM derived metrics.Resources. Task Resources.ASP[i][present]
...	...
CSPAC (*)	SYMDB.PERT/CPM derived metrics.Resources. Task Resources.CSPAC[i][present]
(2.5) Quality	
Defects detected	SYMDB.Collected metrics.Quality. Task Quality.Defects detected[i][present]
Defects reworked	SYMDB.Collected metrics.Quality. Task Quality.Defects detected[i][present]
Cumulative defects detected	SYMDB.Collected metrics.Quality. Task Quality.Cum defects detected[i][present]
Cumulative defects reworked	SYMDB.Collected metrics.Quality. Task Quality.Cum defects reworked[i][present]
Defects awaiting rework	SYMDB.Collected metrics.Quality. Task Quality.Defects awaiting rework[i][present]

Table 7.31 – SYMDB object variables to be updated in SYDPIM activity (M2b) from data collected in activity (M1b)

As also discussed in activity (M2a), the present data-point of target-metrics in the SYMDB, which was estimated in the previous activity (M1b), may need to be updated later at the end of re-planning – if the future plan is changed, the present targets will also change. These target-variables are marked with a (*) in table 7.31 above and their update will take place in SYDPIM planning activity (P6).

Since in this scenario the PERT/CPM model is not used and updated, there is no need to update the analytical links. If later in the project a current PERT/CPM plan is created, the monitoring process will then be entered in activity (M1a) and from there both the PERT/CPM plan and the analytical links will be updated as required in activity (M2a).

(M1c) collect information from expert judgement

Monitoring is entered through this activity in the less structure and formal SYDPIM scenario: neither a PERT/CPM plan is available nor the SYMDB is being implemented. As in the other previous scenarios, the aim is still to specify the patterns of behaviour for the project past which the SD model will have to reproduce and diagnose. Since no quantitative data is being collected and recorded, the only available source of information to derive these patterns is management expert judgement.

In practice, the process of extracting behaviour patterns from expert judgement can be done in a more or less structure way. From informal interviews, brain storming session or use of formal structure methods, like the Delphi method (Wright 1985, Kerzner 1998), the required set of patterns are derived as sets of data-points over-time.

The complete set of patterns to be derived are identified in the object specification of the “Project past behaviour”. This consists of all SYMDB metrics except for the “Uncovered” type of metrics, and plus the expert judgement patterns “risk-related” and “intangible issues” as perceived relevant at this stage in the project life-cycle. In practice, one cannot expect all of these patterns to be extracted. The more the better, and preferably the more representative ones per data-type (i.e. schedules,

effort, scope and resources) – this will be a matter of practical feasibility within the organisation.

Deriving the “risk-related” and “intangibles issues” patterns from expert judgement is also common to the other two monitoring paths that lead to activity (M5). Therefore, it is suggested that this is implemented in the following activity (M5).

(M5) specify project past behaviour – SYDPIM feature

The purpose of this activity is to produce a specification of the whole project past in the form of a set of quantified behaviour patterns over-time, which the SD model will have to reproduce. This is an activity which is always carried out, regardless of whether the PERT/CPM plan is being updated or not, and of whether the SYMDB is in use. If a SD project model is to be used in project monitoring, the specification of the project past is always required. Once the SD model is calibrated to reproduce this behaviour, it will then be used to diagnose the project past and to forecast the future. The accuracy of the description of the project past developed in this activity is critical to the usefulness of the SD model.

With this concern in mind, a structured metrics plan is proposed in SYDPIM – the SYMP. The implementation of this plan leads to an exhaustive collection and recording of the project past behaviour in a database – the SYMDB.

The project past behaviour to be specified in this activity was formally described as a SYDPIM object (see appendix C). This object identifies the specific patterns that need to be generated. Overall, the “ideal” set of patterns that describe the project past behaviour include:

- (a) all the metrics specified in the SYMDB;
- (b) other patterns to be derived from expert judgement, which are considered as relevant for the project at the present stage (e.g. amount of scope changes recently requested by the Client).

Three scenarios were considered in this specification:

- (1) the SYMDB is available and updated with past results;
- (2) the SYMDB is not available and a PERT/CPM model is available and updated;

- (3) both the SYMDB and PERT/CPM model are not available and only expert judgement is available from activity (M1c).

The actions to take in this SYDPIM activity will also depend on these scenarios as follows:

(1) If SYMDB is available:

- Extract from expert judgement the following type of patterns, as perceived relevant: (i) risk-related, and (ii) intangible issues (e.g. staff fatigue). It is important to note that the SD project model must contain variables within its structure which produce these patterns;
- Merge these patterns with the ones stored in the SYMDB and generate the "Project past behaviour" object;

(2) If only PERT/CPM model available and updated:

- Extract from expert judgement the following patterns:
 - (i) quality patterns (per SD-Task and whole project);
 - (ii) performance indices (per SD-Task and whole project);
 - (iii) process metrics (per SD-Task and whole project).For the specific patterns see specification of SYMDB. Collect as many patterns as possible.
- Extract from expert judgement the following type of patterns, as perceived relevant: (i) risk-related, and (ii) intangible issues (e.g. staff fatigue). It is important to note that the SD project model must contain variables within its structure which produce these patterns;
- Merge these patterns with the ones stored in the "PERT/CPM past behaviour" object and generate the "Project past behaviour" object;

(3) If PERT/CPM model and SYMDB not available:

- Extract from expert judgement the following type of patterns, as perceived relevant: (i) risk-related, and (ii) intangible issues (e.g. staff fatigue). It is important to note that the SD project model must contain variables within its structure which produce these patterns;
- merge these patterns with the ones already extracted in activity (M1c) and generate the "Project past behaviour" object.

Once the appropriate actions are implemented, the final set patterns of past behaviour are available. In practice, in what form are these patterns stored? While

SYDPIM proposes generic formal objects, there is no intention to impose any particular technological platform to implement them. Since the object “Project past behaviour” consists of a set of patterns (which in turn are sets of data-points), either a standard database, a spreadsheet, or a specialised application can be used to implement this object. The only requirement is that these data-patterns can be accessed for comparison with the patterns produced by the SD model. The same condition applies to the objects “PERT/CPM past behaviour” and “PERT/CPM future behaviour”.

Another important implementation issue is whether the patterns of this object are re-generated from the beginning of the project, whenever this activity is entered, or whether this object is “preserved” and thereby only the present data-point needs to be updated for each pattern. Both approaches are viable in SYDPIM. The appropriate one depends primarily in the effort required to regenerate the patterns from the beginning of the project. If the SYMDB is being implemented, and the patterns can be automatically regenerated from an application linked to this database, then there is no apparent need to “preserve” this object all the time, because it can be created almost instantaneously. On the other hand, if this object is being updated by hand in the form of a spreadsheet, then it is probably a good idea to preserve it throughout the whole SYDPIM control cycle. An issue of particular importance are the expert-judgement derived patterns: “Risk related” and “Intangible issues”, which may be considered only temporarily in the project life-cycle. In principle, these will not be stored in the SYMDB even if this database is being implemented. Therefore, if this object is “erased” at the end of monitoring, then next time they will not be available unless they have been recorded elsewhere – a possible solution to this problem is to incorporate them in the SYMDB. Whatever the “technological” solution adopted, the key requirement is that the necessary patterns of project past behaviour are produced in this activity within an acceptable level of effort.

As with the “PERT/CPM past behaviour”, if this object is preserved throughout the SYDPIM control framework, then the variables that represent project targets will need to be updated at the end of planning (activity P6).

As shown in figure 7.25 below, at the end of this activity the project past behaviour is specified mainly with data from the PERT/CPM model and from the SYMDB.

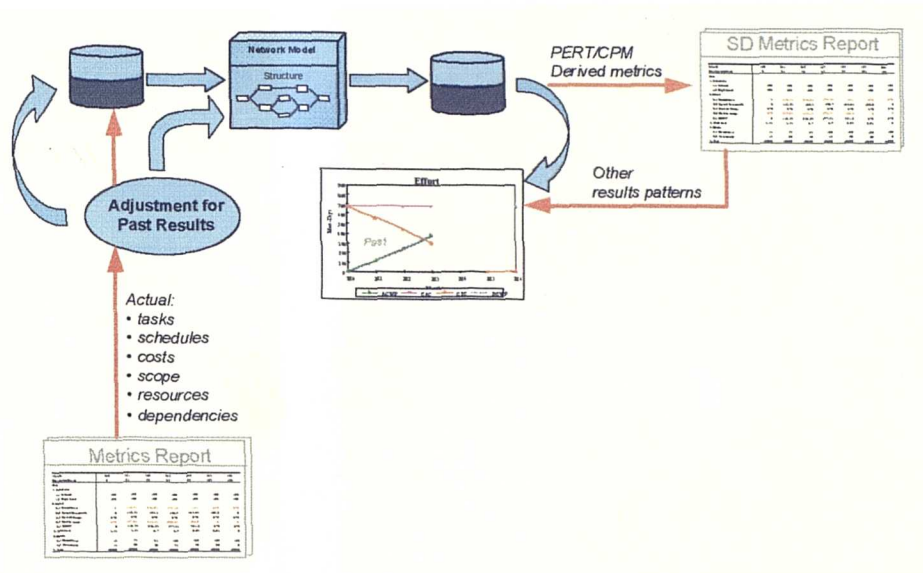


Figure 7.25 – Activity (M5): specification of project past behaviour from PERT/CPM model and SYMDB

Once the newly updated project past behaviour is available, the next step is to calibrate the SD model to reproduce this behaviour.

(M6) re-calibrate SD model for past behaviour – SYDPIM feature

This activity consists basically of a SD calibration exercise, similar to the one described in the planning activity (P4a). The SD model will have to reproduce the project past behaviour, within an acceptable degree of accuracy and for the right reasons.

The SD model will have to reproduce all the behaviour patterns stored in the "Project past behaviour" object. The calibration must be consistent with the real causes underlying these behaviour patterns. As discussed in activity (P4a), while the ideal would be an automated calibration process, this is not available. The calibration of the model to reproduce the actual outcome is highly based on the characteristics of the specific project model being used and requires human judgement. Since the SD model is essentially a causal model, calibrating the model consists in changing what affects what and by how much so that it matches

the observed reality. Without human judgement not only this proves extremely difficult, as important lessons and conclusions would remain uncovered.

Possibly, more than one calibration will generate the desired behaviour patterns and the one that reflects closer past reality should be selected. In the ideal implementation of SYDPIM, both analytical links and the SYMDB will be used to support this exercise. This will depend on the availability of an updated PERT/CPM model and on whether the SYMDB is being implemented.

If the PERT/CPM model is available, the analytical links can be used in this activity for the following purposes:

- (1) impose exogenous decisions in the SD model which will help ensuring that resource profiles and present project targets, as described by the current PERT/CPM plan, are reproduced by the SD model;
- (2) check whether the SD model is reproducing accurately the project behaviour patterns in the "PERT/CPM past behaviour";
- (3) check whether manual re-calibrations change the initial plan.

The SYMDB provides special metrics that help ensuring that the assumptions underlying a particular calibration are valid. These are the "process" type of metrics:

- (a) net productivity (includes QA and rework activities);
- (b) gross productivity (development only);
- (c) defect detection index (defects detected per scope accomplished);
- (d) cost to detect defects;
- (e) cost to rework defects.

A good SD project model must incorporate input variables that represent these process variables, or similar ones (most likely "nominal" values and "learning curves"). This is because any product development process is primarily driven by these type of metrics. The ones above are measurable but there are others which are intangible, like the "defect generation" and the "defect escape" indices. The SYMDB provides not only the present value for these metrics but also they way in which they have been changing over-time in the past (i.e. their shape). Since the calibration of the related SD input variables has a strong impact on the behaviour produced by the model, by forcing these inputs to be consistent with the metrics in

the SYMDB, the “validity” of the calibration is improved. For example, either a low “cost to detect defects” or a “high defect generation” may have similar visible impacts on the project outcome: a higher number of defects being detected and a higher overall rework cost. However, if the SYMDB suggests a low and decreasing cost to detect, only this scenario is likely to be valid. Finally, it is important to note that these process metrics constitute themselves output behaviour patterns that the SD model will should reproduce. This reinforces the validity of the calibration being achieved.

Assuming the preferred SYDPIM scenario, where both an updated PERT/CPM model and the SYMDB are available, the following algorithm describes the process to be implemented in this SYDPIM activity.

First, the SD model is simulated. In the second step, the patterns produced by the model are compared for “goodness-of-fit” against the patterns in the “Project past behaviour”. The behaviour patterns derived from the PERT/CPM model and stored in the “PERT/CPM past behaviour” object are compared by invoking the set of data-consistency links identified in the algorithm. Since the analytical links establish relationships only between data from the PERT/CPM model, the other non-PERT/CPM patterns present in the “Project past behaviour” (e.g. defect patterns) are not compared by the analytical links. Therefore they need to be compared in a conventional manner. In SYDPIM, it is suggested that the same “goodness-of-fit” criteria and quantitative statistics/indices proposed for the analytical links, as described in activity (P4a), are used. These non-PERT/CPM patterns include the quality patterns (i.e. defects), process metrics, performance indices and other patterns derived from expert judgement in activity (M5). It is also suggested that relevant changes to the metrics uncovered from the SD model, using previous calibration, are identified. In principle, the new calibration being carried out should not cause major changes (note that the previous calibration was subjected to thorough validation).

Algorithm: { calibration of SD model for "project past behaviour" }

1. Simulate project in SD model
2. Check SD project outcome against "Project past behaviour"
 - { For patterns available in "PERT/CPM past behaviour" invoke links:
 - DCOI-6.2.SD-PERT : Planned Project Resources Availability
 - DCO-1.2 : Planned Start / Finish Dates of SD-Tasks
 - DCO-2.2 : Planned Effort of SD-Tasks
 - DCO-4.2 : Planned Scope of SD-Tasks
 - DCO-3.2 : Planned Resource Allocation to SD-Tasks }
 - For non-PERT/CPM patterns, check "goodness-of-fit" conventionally:
 - SYMDB.Collected Metrics.Quality
 - SYMDB.Calculated Metrics.Performance Indices
 - SYMDB.Calculated Metrics.Process Metrics
 - Expert judgement patterns.Risk-related
 - Expert judgement patterns.Intangible issues.
 - Check relevant changes in the SYMDB.Uncovered Metrics for the past }
3. IF SD outcome is satisfactory THEN update uncovered metrics in SYMDB AND END
 - { If none of the consistency links or patterns in previous step is violated }
4. Diagnose inconsistencies
 - { Check which consistency links and patterns are violated }
5. Re-calibrate SD model manually
 - { Change the SD input parameters which are likely to eliminate the inconsistencies above. Consider the following actions:
 - i) If the modeller wants impose exogenous decisions to:
 - (a) reproduce past profiles of project resources availability, invoke link:
 - DEI-3.3.PERT-SD : Profiles of Project Resources Availability
 - <Simulate project in SD model>
 - (b) reproduce past profiles of resource allocation to SD-Tasks, invoke link:
 - DEOI-9.3.PERT-SD : Profiles of Resource Allocation to SD-Task
 - <Simulate project in SD model>
 - (c) reproduce present levels of resource allocation and availability, invoke links:
 - DEI-3.2.PERT-SD : Present Level of Project Resources Availability
 - DEOI-9.2.PERT-SD : Present Level of Resource Allocation to SD-Task
 - (d) reproduce present targets, invoke links:
 - DEOI-6.PERT-SD : Present Planned Start / Finish Dates of SD-Tasks
 - DEOI-7.PERT-SD : Present Planned CAC of SD-Tasks
 - DEOI-8.PERT-SD : Present Planned SCAC of SD-Tasks
 - ii) The modeller should consider the present value and shape of the "process metrics" in the SYMDB when calibrating the related SD input variables (e.g. nominal values, and learning curves)
 - If structural changes are made to the SD model, then update the analytical links as required }
6. Check consistency of new calibration with project plan in PERT/CPM model
 - { Invoke following links to check that parameters of the initial plan were not modified:
 - DCI-1 : Profiles of Project Resources Availability
 - DCI-2 : Project Start Date
 - DCOI-1.PERT-SD : Start and Finish Dates of SD-Tasks
 - DCOI-2.PERT-SD : Budget of SD-Tasks
 - DCOI-3.PERT-SD : Scope of SD-Tasks
 - DCOI-4.PERT-SD : Profiles of Resource Allocation to SD-Task
 - DCOI-5.PERT-SD : Budget Breakdown of SD-Tasks
7. IF calibration is not satisfactory THEN diagnose inconsistencies AND GOTO 5
 - { Check which consistency links are violated. Note that modifications to the parameters that describe the initial plan should not be allowed. }
8. GOTO 1

Once the “goodness-of-fit” is assessed, if this is satisfactory (unlikely to happen the first time), then the “uncovered metrics” stored in the SYMDB are updated (from the beginning of the project up to present), and the process ends. This update of the uncovered metrics is due to the fact that the new re-calibration of the SD model may, in some cases, change the past values of these metrics. This is based on the argument that, as more project past is available, the past calibration can be improved. For example, the new reality may indicate that last month there were less defects uncovered in the designs than estimated in the calibration at that time.

If the “goodness-of-fit” is not satisfactory, the patterns not being reproduced accurately are identified and diagnosed. At this stage it is necessary to re-calibrate the SD model so that the gaps are eliminated. Here two actions are suggested for consideration by the modeller:

- (i) call the analytical links to impose exogenous decisions in the SD model;
- (ii) compare the values of the relevant SD input variables with the process metrics in the SYMDB.

In action (i) links can be invoked either to adjust the past profiles of resource availability and allocation, or to adjust the current project targets. In the first case, the actual profiles of resources availability / allocation is imposed as an exogenous decision (i.e. the gap from the initial plan). The purpose is to supply the SD model with the actually required resources so that no endogenous corrections are perceived necessary within the model. It is important to note that this is a purely-exogenous solution for reproducing this pattern and hence is not desirable unless the real causes are not captured within the model. Note that while the links DEI-3.3 and DEOI-9.3 were specified in such a way that the whole past profiles are adjusted, the modeller may consider their implementation for only part of the project past. In case these links are invoked they will affect the present level of resources generated by the SD model and therefore the SD model should be simulated again.

The following two links that can be called will impose an exogenous in the model which will adjust only the present level of resource allocation / availability. These links can be used more regularly, in case the SD model is also re-calibrated regularly against the project past behaviour. In this scenario, the modeller will try to re-calibrate the model in a valid way such that the resource levels produced

endogenously are as close as possible to the desired levels. The validity of the calibration should not be violated for the sake of more accuracy. In reality, part of the causes can be exogenous to the SD model. To capture this exogenous component, the DEI-3.2 and DEOI-9.2 links can be invoked to generate the final exogenous adjustment (for the present data-point only).

In a similar way, the last three links that can be invoked adjust the expected project targets (schedule, cost and scope) at the present moment. These links can also be considered as final exogenous adjustments to the target-type of patterns for the present moment. However, it is important to note that the value of a target pattern for the present moment depends not only on project history but also on the current plan for the remaining future. Since this future plan is about to be readjusted in planning, the present value of the targets will also change. The value in the "PERT/CPM past behaviour" extracted in activity (M4) represents the expected targets if no re-planning of the future is implemented. On the other hand, the value produce by the SD model reflects some level of endogenous re-planning within the model. The main conclusion is that *the value of the present data-point for the target-patterns does not have to reproduced accurately by the SD model at this stage (unless the work is complete in the SD-Task or in the project)* – this will have to happen at the end of SYDPIM planning. Nevertheless, large deviations should be analysed as they may reflect an inconsistent calibration of the SD model. The links DEOI-6 to DEOI-8 can be used to "correct" the SD targets with an exogenous decision, but this should be done at the modeller's explicit preference.

It is also important to note that the adjustment of the present levels of resource allocation / availability implemented by the previous links DEI-3.2 and DEOI-9.2 will affect the targeted cumulative staff at completion – it is considered that this adjustment is done automatically within the model. More generally, it is assumed that exogenous adjustments to project targets introduced in the SD model will generate automatically within the model all the necessary adjustments so that an overall consistency of the project future plan is maintained. For example, if the scope at completion is increased through an exogenous input, then the necessary extra work units (i.e. elementary work tasks) must also be generated within the model. Any good SD project model which provides the user with exogenous inputs will also generate these required adjustments in other parts of the model.

In action (ii) it is suggested that the manual re-calibration of the SD model is carried out in a consistent manner with the values and shapes of the process metrics in the SYMDB. This helps to ensure that a “valid” calibration is achieved.

As mentioned, deciding what to change in the model is highly depend on the structure of the specific model. SYDPIM recommends that each organisation using a certain type of model, or style of modelling, should try to develop a set of guidelines about the calibration process. These will be derived from experience. While difficult to define model-independent generic rules, SYDPIM suggests the following principles:

- (1) calibration should be focus on those parameters which represent those conditions which have potential to change during implementation. These parameters are generally work-dependent. For example, the “average defect generation index” could have been different than expected in the plan. However, the “effect of the team size on productivity” overheads is unlikely to depend on the specific work plan. Instead, this effect is more likely to be a generic organisational characteristic independent from the specific work plan used to implement the project;
- (2) structural changes are expensive and can have a great impact on the outcome produced by the model. They can even change the representational meaning of certain variables, thereby affecting the validity of calibrations. These changes should only be implemented if they objectively represent actual major changes in the project. The representational meaning of the affected variables should be checked;
- (3) never sacrifice the validity of the parametric changes against the accuracy of the behaviour produced. A model is always a partial representation of the real system. Sometimes, a fully accurate reproduction of observed behaviour is not possible to achieve due to the simplifications in the model. The model should not be forced to produce a closer output on the basis of changing parameters to unrealistic values;
- (4) as proposed in the model validation framework of chapter 6, when a model is developed, ranges of validity for the values of the parameters should be specified. It is critical to maintain and improve this information based in practical use of the model. When calibrating the model, the values of the parameters

should not be change beyond these ranges. If that proves necessary, it should be subjected to careful analysis.

Finally, it is important to note that if the calibration of the SD model required structural changes then the analytical links should be updated as required – however, this is unlikely to happen.

In the next step consistency links are called to check whether changes to the initial plan were made. As discussed in the description of activity (P4a), these changes should not be allowed – while changes to the initial plan could prove effective in the achievement of the desired patterns, the reality is that the project was started with the expectations and work plan set at that time. In case the calibration is not satisfactory it should be changed, other wise the SD model is simulated again and the process repeats.

The algorithm proposed considers that both an updated PERT/CPM model and SYMDB are available. In case the SYMDB is not available, the process is the same except that:

- the patterns that the SD model will have to reproduce are only those stored in the “PERT/CPM past behaviour”. The non-PERT/CPM patterns are not available;
- no “process metrics” are available to guide the manual re-calibration of the SD model.

With less patterns to reproduce and with no process metrics available, the “level of validity” achieved in calibrating the SD model is likely to be lower and the re-calibration exercise eventually more difficult. Since the “quality” of this calibration has a great impact on the reliability of both diagnosis of the project past in monitoring and forecast of project future in planning, the implementation of the SYMDB is highly recommended. Nevertheless, the patterns available in the “PERT/CPM past behaviour” are meaningful and a proper implementation of SYDPIM is still achievable.

In case the PERT/CPM is not available, the main consequence is that the analytical links are not available to check the consistency of the produced behaviour, to

generate exogenous decisions and to check changes to the initial plan. The “goodness-if-fit” will have to be done conventionally for all patterns of past behaviour, and statistics / indices previously proposed are recommended. The process will be similar to the one described in the algorithm above.

In the worse scenario, neither the PERT/CPM model nor the SYMDB are available. The process would still be based on the algorithm above and would result in the following:

Algorithm: { calibration of SD model for project past behaviour }

1. Simulate project in SD model
2. Check SD project outcome against “Project past behaviour”
{ Check “goodness-of-fit” conventionally }
3. IF SD outcome is satisfactory END
{ If all patterns reproduced accurately }
4. Diagnose inconsistencies
{ Check which patterns are violated }
5. Re-calibrate SD model manually
{ Change the SD input parameters which are likely to eliminate the inconsistencies above }
6. GOTO 1

The algorithms for the other scenarios (i.e. SYMDB only and PERT/CPM model only) can be easily derived from the main algorithm above.

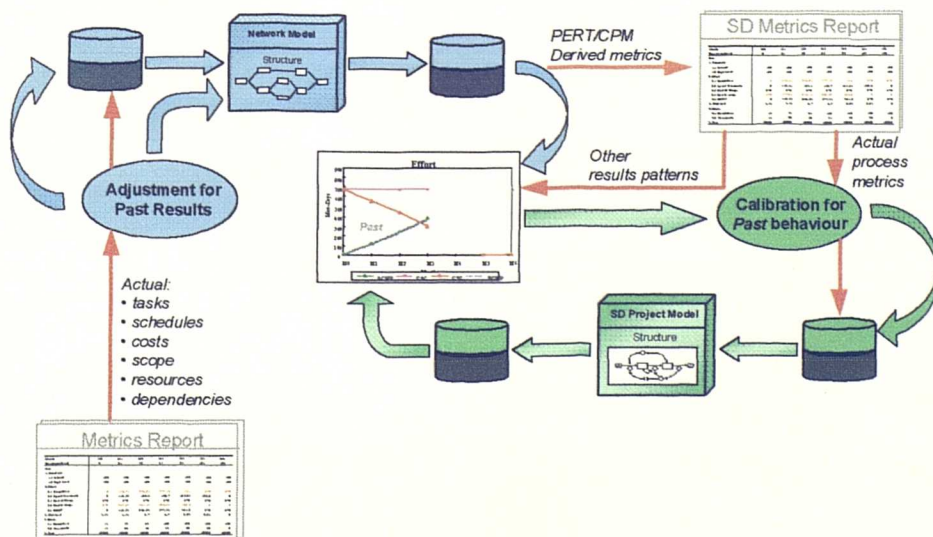


Figure 7.26 – Activity (M6): calibrate SD model to reproduce the project past behaviour

As shown in figure 7.26, at the end of this activity the SD model has been calibrated to reproduce the project past behaviour. The next step in monitoring is to diagnose this project past.

(M7) SD diagnosis of project past – SYDPIM feature

One of the most powerful features of a SD model is the ability to explain why things happen in a certain way. This is because a SD model generates the outcome endogenously from within its feedback structure, which holds the causes for behaviour. If the SD model is reproducing well the project past it can answer questions like: why have we spent more effort in reworking defects than planned? Why is the integration phase late? Why has productivity been lower than expected? This explanatory power is enhanced by the possibility of simulating retrospective “what-if” scenarios in the past and by the model’s ability to quantify unmeasured project status information.

In SYDPIM it is proposed that the SD diagnosis of the project past is based on the following actions:

- *extract performance and process-related metrics and compare them against the assumptions in the plan* – this identifies what happened differently than assumed (e.g. lower defect detection) and helps to understand why therefore the outcome was also different than expected (e.g. this is why we spent less effort than planned);
- *uncover intangible metrics about the present project status* – this enhances management perception of performance making it more realistic. For example, what is the likely amount of undetected defects currently in the system? Did staff fatigue affected the quality of their reviews? This extra information also helps to understand the causes of the current outcome. For example, due to fatigue the quality of the reviews was lower and more defects escaped. This is why less were detected and cost savings were achieved. The amount of undetected defects is worryingly high;
- *retrospective “what-if” analysis of other scenarios* – once calibrated the model simulates what really happened in the project so far. Changing the appropriate metrics, it is possible to simulate “what-would-have-happened” type of scenarios. These are not fictitious scenarios produced by the modeller’s

creativity but rather they should reflect relevant situations from which the project manager can derive important lessons to be applied in the remaining of the project. Useful scenarios generally answer questions like: could we have achieved better results if we had planned the project differently? What control actions could have mitigated effectively certain risks that occurred? This type of analysis can be used as the basis for continuous process improvement within the present project and across other projects;

- *causal analysis* – causal analysis is at the core of the SD approach. This consists in relating the behaviour produced by the model with its feedback structure. The project outcome is therefore explained in terms of feedback loop dominance and influence over the project. For example, the vicious circle of adding more staff leading to higher overheads, more errors and thereby to more rework was the dominant loop responsible for delays during a certain stage of the project. This analysis should also focus on identifying the feedback loops which are currently dominating the project outcome. By introducing control actions that affect the strength of these loops, the project manager can change the project outcome in the future. However, it should be noted that it is not easy to implement this type of analysis in an objective manner, specially if the SD model is complex and has many loops within its structure. Identifying the dominant loops and relating these to the outcome is an issue in SD which needs scientific developments (Richardson 1996). The SYDPIM model development method presented in chapter 7 proposes a generic feedback structure and a framework to keep track of the existing loops within the model. This framework should be used to support this type of analysis.

There are two types of output from this exercise: (i) generic lessons learned, and (ii) specific results, like quantitative estimates and selected decisions. In SYDPIM it is suggested that this type of output is recorded and is used to support re-planning within the current control cycle, as well as for comparison with future results. SYDPIM does not propose any specific type of results except for estimates of intangible metrics which will be stored in the SYMBD, in the following activity (M8). The following steps are proposed as guidelines to carry out the diagnosis in this activity:

- (1) *analysis of process metrics* – compare and quantify the deviations between the actual values of variables which are related to the process metrics, against those values assumed in the plan. These variables will typically include nominal

- values (e.g. nominal productivity) and endogenous effects over these nominal values (e.g. effect of progress on productivity; see Abdel-Hamid and Madnick 1991) – note that some of these effects are not affected by the work plan and its implementation, and so in most cases these should not differ (e.g. organisational, product type). Based on these deviations try to explain the actual project outcome against the plan (i.e. observed vs. planned behaviour). This type of analysis leads to explanations like: “Schedule slippage was due to lower productivity and defect generation higher than expected. Lower productivity was mainly due to a slower learning curve than assumed in the plan. Higher defect generation was due to the fact that the work was more complex than expected.” While the causes reported by this type of analysis have an impact on feedback loop dominance within the project, they are not themselves feedback loops. Feedback loops dominance is analysed on the following step;
- (2) *causal analysis* – based on the shapes of the behaviour patterns, identify the most relevant dynamic characteristics of the project behaviour, and try to relate these to feedback loop dominance. These dynamic characteristics typically refer to major changes (often persistent) to the project targets (e.g. the completion date suddenly slipping by). More generally, significant changes against the expected planned shapes typically result from the dominance of a certain loop (often undesired). In this diagnosis, it is also important to consider and analyse combinations of the dynamic changes. This type of analysis leads to explanations like: “Higher defect generation led to more rework discovery, which led to schedule pressure, poorer quality and even more defects being generated. This feedback loops caused a significant schedule slippage in coding.” In most cases, the dominant feedback loops identified in this step were strengthened by the deviations in the process metrics identified in the previous step. This relationship should be considered and explored in this causal analysis;
- (3) *what-if analysis to validate conclusions* – the analysis in the two previous steps lead to hypotheses of what were the causes for the observed past behaviour of the project. In some cases, there may be a clear evidence that these hypotheses are correct. However, in other cases that might not be so obvious. Furthermore, within the non-linear dynamic structure of the SD model, counter-intuitive effects abound. Therefore, the best approach to validate the explanatory conclusions taken in the previous steps is to run simulation tests.

For example, if a higher defect rate is thought to have been responsible for a schedule slippage, then a simulation can be run with lower defect rate to check whether the slippage is eliminated. Similarly, if a certain feedback loop is thought to have been the cause for a delay, then the strength of such feedback loop can be weakened and a simulation is run to test the new outcome. While the scenarios run in this type of analysis may reflect situations that could have occurred in the project, their primary aim is to test the explanatory hypotheses developed in the previous steps;

(4) *what-if analysis for process improvement* – another major advantage of running simulations in the past is to verify whether alternative planning and control decisions and policies would have performed better. Of course, this type of analysis can also be done in the future to improve the project plan (this is done in SYDPIM activities P1b and P2b). However, when done in the past this analysis benefits from a stronger validation because many of the assumptions in the plan are based on real data. For example, the conclusions from a future analysis may prove wrong if the assumed productivity is much higher than what will really be achieved, whereas in the past the productivity in the model must be consistent with the actual productivity in the project. The project past therefore constitutes a more valid arena to assess planning and control decisions and policies. In this step the project manager should focus on identifying scope for potential improvements which can be applied to future planning. The purpose is not to identify what or who to blame for poor performance.

Once these steps are implemented, the lessons learned and specific results should be recorded to support re-planning and future analyses. Since various scenarios were run in the model, the initial scenario entered in this activity, where the mode is reproducing the actual project past, should be re-set before the monitoring process proceeds to the following activity – this implies that the calibration for this scenario is recorded somewhere, possibly within the SD modelling tool.

The figure 7.27 shows the state of the SYDPIM monitoring process at the end of this activity.

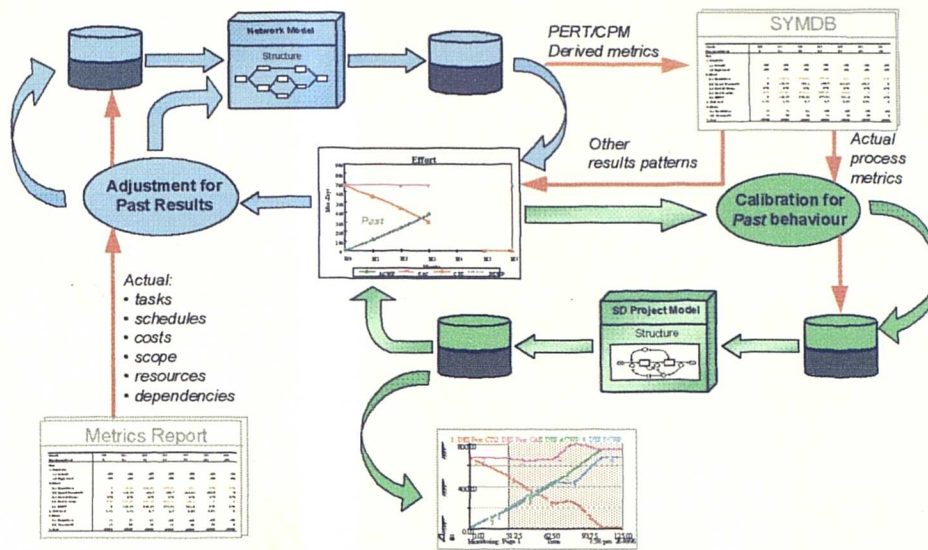


Figure 7.27 – Activity (M7): diagnosis of project past in SD model

(M8) update SYMDB with uncovered SD metrics – SYDPIM feature

There is a particular set of quantitative results which SYDPIM suggests is stored in the SYMDB (in case this database is being implemented): the intangible project metrics uncovered by the SD model. This is one of the most important and distinctive benefits of a SD project model: to provide enhanced project status information, by quantifying important metrics, which otherwise would be difficult or impossible to measure.

In SYDPIM, by “uncovered metrics” it is meant all those metrics about the project past behaviour and status, which are not collected directly from the project and which are not estimated by any other means. In practice, these metrics are either of intangible nature, and therefore impossible to collect, or they are difficult to estimate accurately using another mechanism, or yet the effort requires is too high. This type of metrics can hold a great value for understanding the project past and current status, and to anticipate future trends. For example, a project currently on-schedule but with a very high number of undetected defects might not be in as good health as it may look like under a conventional assessment: heavy rework and slippage in later stages is the likely future. Uncovering these metrics and storing them in the SYMDB provides the project manager with valuable information to re-plan the project before implementation proceeds in the wrong direction.

As SD project model contains many of these metrics within its structure, which are produced as an output. This is often regardless of whether the project manager needs and uses them – the SD model simply requires them to represent the project reality in a valid manner. Of course, the particular set of metrics that can be uncovered from the SD project model depends on the specific model being used.

Three sub-categories are proposed in SYDPIM based on project issues that a good SD project model should address and quantify: undetected quality issues (e.g. undetected defects), staff issues of subjective nature (e.g. staff fatigue) and feedback effects on process parameters (e.g. effect of defect density on cost to detect). A specific sub-set of metrics within each category was proposed in the formal specification of SYMDB. The object variables for each category are as follows:

- SYMDB.Uncovered metrics.Defect
- SYMDB.Uncovered metrics.Staff
- SYMDB.Uncovered metrics.Effects on process

Within each of these three categories the following specific metrics are proposed (the SYMDB object variables are straightforward to identify):

Uncovered SD metrics
<i>(1) Defects – for each SD-Task and for the whole project</i>
(1.1) Undetected
(1.2) Cumulative generated
(1.3) Generation rate
(1.4) Undetected density
(1.5) Cost to detect next
<i>(2) Staff – for each SD-Task and for the whole project</i>
(2.1) Fatigue
(2.2) Experience
<i>(3) Effects on process metrics – for each SD-Task and for the whole project</i>
(3.1) Productivity
(3.2) Defect generation
(3.3) Defect detection
(3.4) Defect rework

Table 7.32 – Project metrics to be uncovered using the SD model and stored in the SYMDB, in the SYDPIM activity (M8)

These specific metrics proposed in the SYMDB are aimed at providing a framework from which the modeller may develop a specific set adjusted to the particular SD

model. For example, some SD models widely used in practice do not consider a dual flow of work and defects, but simply closed loop of work tasks (e.g. Cooper 1980). This type of structure does not allow for defects to be tracked and quantified. In this case, the proposed metric “Undetected defects” should be replaced by “Undetected tasks needing rework”.

The process in this activity is simple and can be automated through the use of a software tool. Using the calibration for the past behaviour developed in activity (M6), it consists of storing the present values of the uncovered metrics in the SYMDB. For example, storing the present number of undetected defects in the system, or storing the present effect of staff experience on productivity. Like with all the other metrics, over-time the SYMDB will store the past behaviour patterns of these metrics.

The state of the SYDPIM monitoring process at the end of this activity is shown in figure 7.28 below.

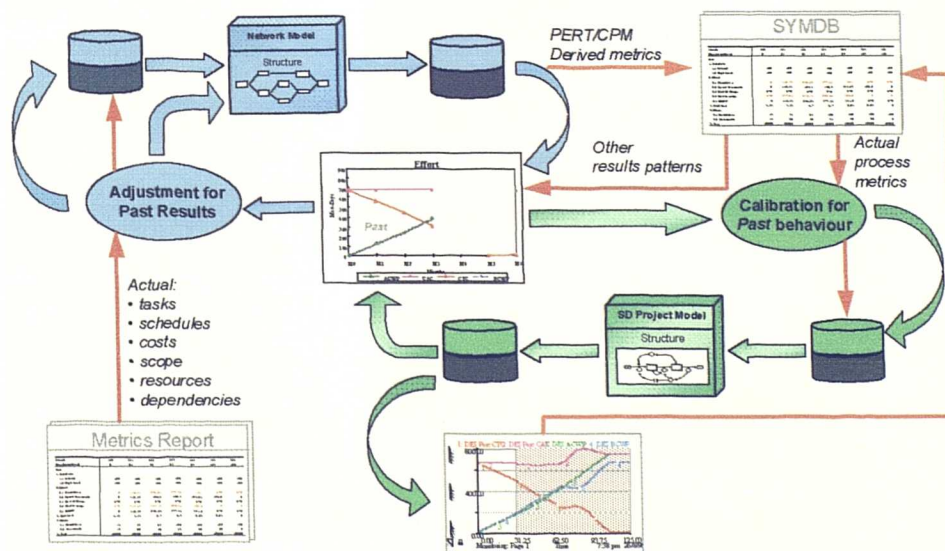


Figure 7.28 – Activity (M8): update SYMDB with uncovered SD metrics

With the SYDPIM monitoring process complete the project manager has developed an understanding of what is the real project status and performance, based on the diagnosis provided by the SD model. Throughout the monitoring process, the SD

model was calibrated to reproduce the project past and all the analysis was focused on this past segment of the project. However, as the project past unfolds a new vision of the future emerges. Given this past, and if the project plan is not changed, what will happen? Is the project still on target? The answer to this question is probably different from the one in the previous month... Does the future plan needs to be changed? Which control actions are more effective? Both the SD model and the PERT/CPM model provide answers to these questions, probably, and interestingly, different ones.

The SYDPIM process now moves onto planning where the project future is the concern to address. The metrics stored in the SYMDB, the information and the understanding developed during monitoring will provide valuable guidance in the (re)planning process. Figure 7.29 below shows the state of the SYDPIM process at the end of monitoring, just before planning is initiated.

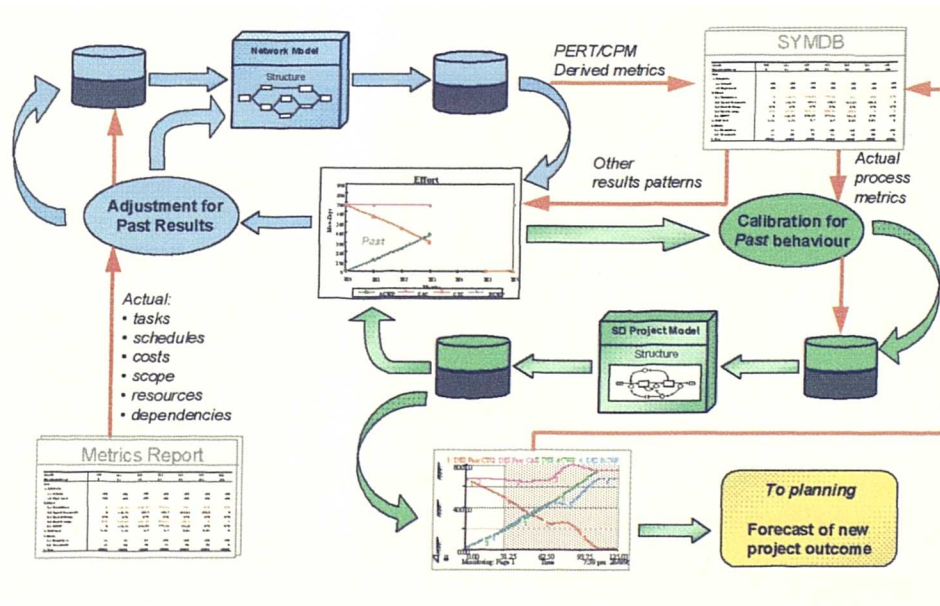


Figure 7.29 – From monitoring to (re)planning: given the updated past, a new forecast is produced by the SD model and the SYDPIM process proceeds to (re)planning

Summary

The SYDPIM activities are the elementary steps of the SYDPM process logic. Within each of these activities, project planning and monitoring actions are undertaken to control the project towards its targets. Some of the SYDPIM activities

are already part of the conventional PERT/CPM based framework, while others are new features due to the integrated use of a SD project model. For example, updating the PERT/CPM model with actual results is a conventional activity, while calibrating the SD model to reproduce the project behaviour, extracted from the PERT/CPM plan, is a new SYDPIM feature. Depending on the purpose of the activities, some can be described in a more formal and automated manner, while others involve more human judgement and are less structured. Nevertheless, for all cases the SYDPIM activities can be considered as a structured processes and were described in this way. Where possible, formal algorithms were proposed to support the implementation of these processes. The SYDPIM activities access the SYDPIM objects to retrieve information and also store data in these objects. Some SYDPIM activities "call" the analytical links (as if they were software routines), in order to implement the required actions.

The SYDPIM activities were described separately for the planning and monitoring functions of project control. The sequence in which the activities are to be implemented in the real world is imposed by the SYDPM process logic. However, this logic has loops and there are branches which provide alternative paths. Therefore, particular sequences were proposed to describe the SYDPIM activities, based on the conventional framework and likely planning and monitoring scenarios.

Since the SYDPIM process logic is flexible to accommodate various scenarios regarding model updated and data availability, the activities also need to be flexible to accommodate these same scenarios. Therefore, in some cases the processes within each activity may differ according to the scenario. The alternative scenarios considered refer to: (i) the frequency with which the PERT/CPM model is updated with actual results and hence to the availability of an updated plan, and (ii) the implementation of the SYDPIM metrics plan and hence to the availability of the SYMDB. In the ideal scenario, the SYMDB is available and the PERT/CPM model is updated in every control cycle. However, the practical implementation of SYDPIM also accommodates the worse scenario where, half-way through the project, none of these elements is available.

Overall, the planning activities are aimed at developing a project plan by adjusting and calibrating one model to the other. The SYDPIM planning process is based on

the principle that a new plan can be developed in either of the models, but it must then be tested in the other model. A project plan should only be implemented when represented in both models in a consistent manner, and when the analysis in each model is satisfactory. The adjustment and calibration of one model to the other is based on the principle that it must reproduce the same outcome and the that underlying assumptions are the same (i.e. for the right reasons). This process requires the exchange of data to ensure that the assumptions are consistent. It also requires data consistency mainly to check whether the outcome produced is the same. The transfer of a project plan from the PERT/CPM model to the SD model requires that the project dynamic behaviour is extracted from the PERT/CPM model. For both models, the process of model adjustment and calibration is partially automated through the use of the analytical links. It requires human judgement to be completed and validated. The calibration of the SD model can be supported by process metrics regarding the project past which are stored in the SYMDB. The process of readjusting one model to the other, in particular readjusting the PERT/CPM model, may imply the update of the analytical links. At the end of SYDPIM planning, both models are representing the same project plan and reproducing or portraying the same project outcome – the project behaviour produced by the SD model will be steady. The following step is to implement this plan.

The monitoring activities are aimed at updating both models with the actual results of the project past, diagnose this past in the SD model and update the SYMDB database with the relevant metrics. The process of updating these three objects in monitoring starts with the collection of data, followed by the update of the PERT/CPM model and of the SYMDB. Some of the metrics in the SYMDB will be derived from the PERT/CPM model, while others will be collected or calculated. In monitoring, the SD model is always re-calibrated to reproduce the project past, which is derived from the PERT/CPM model and from the SYMDB. If the PERT/CPM model is updated and available, the analytical links will be used to support this re-calibration process. In case structural changes are required to the SD model, then the analytical links may need to be updated. The diagnosis of the project past may uncover important metrics about the project status and past performance. These uncovered metrics will be stored in the SYMDB. In the end of monitoring, the PERT/CPM model is updated with actual results and the SD model

is calibrated to reproduce the updated project past. Both models are representing this same past reality and both models reproduce or portray the same project outcome, in the past. However, their view of the future will probably differ. The SYDPIM process logic follows to planning, where the project future is to be forecasted and analysed and a new adjusted plan is to be produced.

7.5 Final overview discussion

The SYDPIM Project Management Method described in this chapter provides a new formal framework for project control. This framework is based on the continuous use of a SD project model at the core of the control process, enhancing the planning and monitoring functions. In planning, the SD model works as a “what-if” test-bed, where a project plan can be assessed and improved prior to implementation in the real world. It provides a “virtual reality” simulation of the project, showing what is likely to happen if the current plan is implemented under various scenarios. In monitoring, the SD model provides a “diagnosis laboratory”, where the causes for past performance are identified. Retrospective “what-if” analysis can be carried out to support continuous process improvement. While used in this way, the SD project model is formally integrated with the conventional PERT/CPM model, through the establishment of analytical links of structure and data. Structural links are the basis to ensure that the project reality being represented in both models is the same. Data links support this representational consistency, and further allow data to be exchanged automatically between the models. The SYDPIM method also proposes the implementation of a specialised metrics plan aimed at improving the formal integration of the models, the managerial understanding of the project status, and thereby the performance of the overall decision-making process.

The ideal of this SYDPIM integration is an automated process to transfer the project representation from one model to the other. However, restrictions to this ideal scenario stem from differences in the scope of the models (i.e. they include different aspects of the project) and in the level of detail they consider. Elements in one model might not have correspondent elements in the other model. Most elements in the SD model need to be dis-aggregated so that they are transferred to the PERT/CPM model. Therefore, human judgement is required in this process. While this results from practical restrictions, human judgement is always a desirable

ingredient in the process – in general, attempts at fully automating processes which take place within complex social systems are likely to overlook important human aspects of subjective nature; hence the results they provide tend to become of limited practical use. The process of integrating the two models proposed in SYDPIM is therefore semi-automated and is based on the use of the analytical links.

While being a formal methodology, the aim of SYDPIM is not to provide a radical re-engineering of the project management process. On the other hand, the existing body of knowledge in this field incorporates an extensive set of concepts, methods, procedures and techniques, all of which are most valuable to project management. While a new theory, the aim of SYDPIM is to allow the project manager “to see farther by standing at the shoulders of giants”. In this way, SYDPIM builds upon the traditional project management framework, introducing some important novelties in the process. These novelties can be grouped and summarised as follows:

- (1) *changes to the project management process framework* – new steps have been added to the process logic of the planning and monitoring functions. This results in new activities having to be performed. These changes in the logic induce further changes in both technical and managerial aspects of the framework. Examples of technical changes are: a SD model will have to be available, and new data repositories need to be developed and updated. Possible managerial changes include: investing effort in collecting metrics, keeping the PERT/CPM model updated on a regular and timely manner, and interpreting the project results and assessing plans in a different way. Changes in the process logic may also induce changes to the management culture. SYDPIM motivates a proactive and realistic posture, where problems and risks are to be anticipated and identified earlier in the project, and where continuous process improvement is implemented. A critical requirement of the framework is the timely implementation of the SYDPIM activities. In particular, data and analyses from the models must be available whenever required, or otherwise the events in the project will overtake control. This requirement may impose changes to existing practices within the organisation.
- (2) *development, maintenance and use of specific objects* – a SD project needs to be developed, maintained and used throughout the project. The quality of this model is critical to the overall performance of SYDPIM. For this reason, SYDPIM provides a formal model development method (described in the

previous chapter). Other new objects required include the data repositories of project behaviour (e.g. “PERT/CPM future behaviour”), which are not used in the conventional framework. For some organisations, the SYMDB database may also be a new object that needs to be implemented. The implementation and maintenance of all these objects requires appropriate technological platforms, so that a timely update is achieved;

- (3) *development, maintenance and use of analytical inks between a PERT/CPM model and a SD project model* – the analytical links between the two models are perhaps the core technical novelty of SYDPIM. They can be used outside the scope of the SYDPIM framework – for example, for a one-off transfer of the history of a past project from the PERT/CPM model to a SD model, for diagnosis. But without the links, the SYDPIM framework could only be implemented at a very informal level, losing a great part of its discipline, validity and value. As with the objects, the analytical links require an appropriate technological platform for implementation.

These novelties need to be introduced in any organisation which intends to implement SYDPIM. What are the critical factors of implementation? The success of SYDPIM depends primarily on the successful achievement of the required changes to the conventional framework. The main critical factors are:

- (i) the level of structuring of the project management framework already in place;
- (ii) the management support and willingness to accept the required cultural changes;
- (iii) the technological platforms chosen to implement the objects, the analytical links and some of the processes within the activities;
- (iv) some key requirements regarding technical modelling;
- (v) the possibility of implementing SYDPIM gradually.

Regarding the first two factors, the more structured the existing framework the easier will it be to implement SYDPIM. The same applies to management support and cultural openness towards the required changes. The availability of a structured management environment typically depends upon the complexity and size of the project. Small-size projects are less likely to use highly structured management techniques, but for this type of project it is also unlikely that long term interactions

and feedback effects are relevant and hence problematic under a managerial perspective. On the other hand, these effects can be overwhelming in complex projects, and it has recently been recognised that highly structured management approaches must be in place (Cooper 1997, Conrad 1997, Williams 1997). Within a specific organisation, SYDPIM can be implemented more or less formally. This should depend on how structured the management process is, on the information made available by this process, and also on the experience gained and benefits perceived by the organisation while using SYDPIM.

The technological platforms are a major critical factor due to the timeliness required, which is imperative to achieve – a project standing by, waiting for data and control decisions, will most likely end-up moving ahead randomly and out of control. It is fundamental that the SYDPIM process does not “derail” due to slowness and consequent accomplishment of activities out of sequence. Technological platforms ensure that the required speed and reliable information will be achieved, preventing this to happen. In SYDPIM, technological platforms can be used to implement the following elements: objects, analytical links, and some sub-processes within the SYDPIM activities. It is not the purpose of SYDPIM to impose specific platforms, or analyse the possible candidates in detail. However, some general guidance is here proposed.

In the first place, it is important to consider that the appropriate platforms for each of these elements are not independent one another. For example, the platforms chosen for the different objects should be compatible, because there are various processes that compare or transfer data among them – e.g. comparing the project behaviour produced by SD model against the behaviour extracted from the PERT/CPM model. For this same reason, the platforms chosen affect the efficiency with which the SYDPIM activities can be implemented. Some of the processes within the SYDPIM activities are routines which can be fully automated, and may also involve processing large sets of data. This can be very laborious if implemented manually and hence these processes are good candidates to be implemented as software routines. These routines will work as software interfaces between the objects and analytical links. Therefore, the platforms chosen will affect the effectiveness of these software interfaces, and even whether they can be developed. Figure 7.30 shows the main sub-process within the SYDPIM activities

which can be implemented as software routines – numbered 1 to 6. There are other semi-automated processes which could also be supported by software routines (e.g. updating the analytical links from changes in the models) – for the sake of simplicity, these are not shown here. Figure 7.30 shows how these processes interact with the SYDPIM objects and with the analytical links.

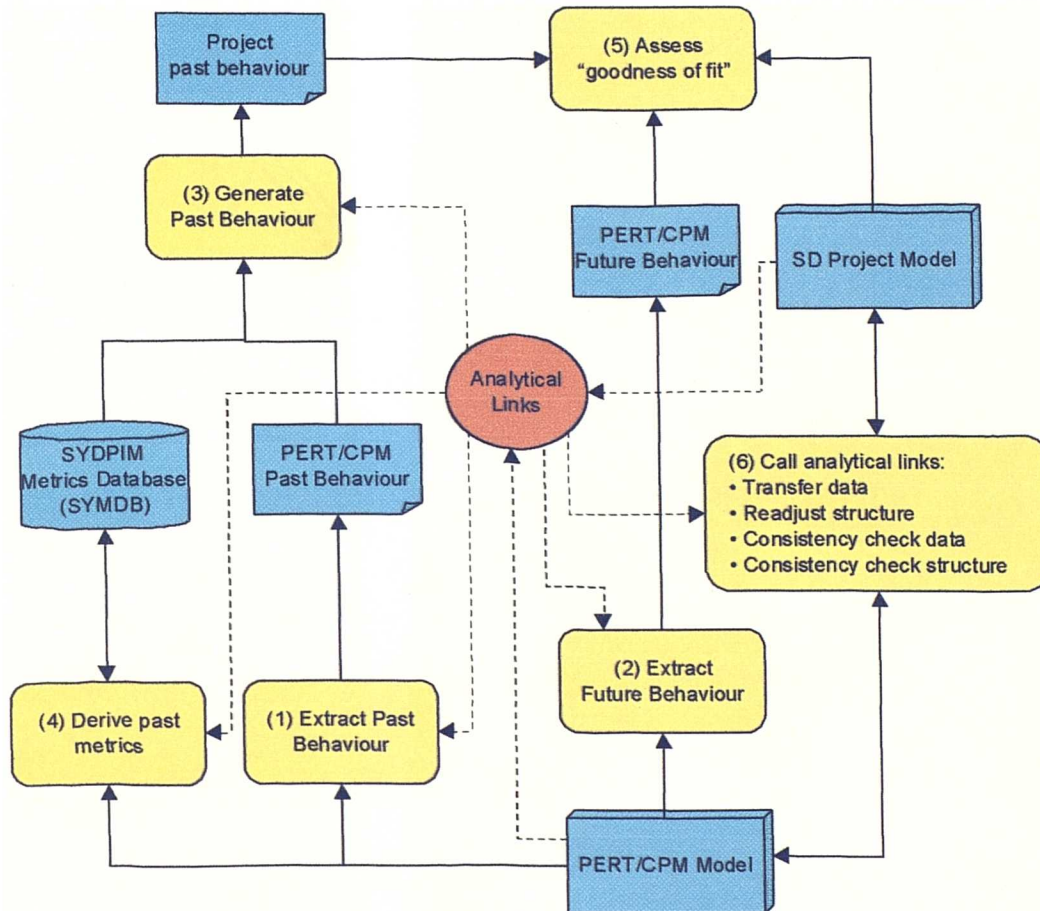


Figure 7.30 – The main processes within the SYDPIM activities which can be automated using software routines

Considering the relationships in this scenario, the SYDPIM guidelines regarding the choice of technological platforms are as follows:

- (a) *PERT/CPM model* – this model should be implemented using a conventional tool. The key requirement is to satisfy the assumptions of SYDPIM basic mode regarding this model. To the author’s knowledge, there are several tools available in the market, most of which support the assumed features. Since this model will be accessed by many of the automated processes (four are identified above), it is very important that its integration with other software applications is technically feasible and easy to implement – mainly, the reading and writing of

data to the project plan. It is also important that various network plans can be recorded in the tool, over the course of the project. It will also be useful if the tool supports the development and maintenance of a WBS and OBS for the project. Finally, features embedded in the tool which generate automatically various data-reports (in particular over-time data), will be very helpful (this supports the processes (1), (2) and (4) above);

- (b) *SD project model* – this model should be implemented in an existing SD modelling tool (see Coyle 1996 for a review; in particular see: Powersim, iThink/Stella and Vensim). The main requirement regarding the implementation of the SYDPIM framework is the possibility of data being read and written to the project model. This is required for the processes (5) and (6) above. This implies an easy integration with other software applications. Both, an automated implementation of the analytical links and an assessment of “goodness of fit” are highly desirable, as they accelerate the SD model calibration process. Therefore, the availability of automated algorithms for “goal seeking” and “goodness-of-fit” testing (to support re-calibration of the model) are an important feature. Other useful features are: the possibility of recording various simulation scenarios while maintaining active a base-scenario, easiness of extracting data from the model, effective configuration management of various versions of the SD model (note that the model can change throughout the project), and causal-tracing and feedback loop analysis features (to support the diagnoses of the produced behaviour). As discussed in the previous chapter (description of the SYDPIM Model Development Method), if a complex multi-phase project model is to be used, then a critical feature of the SD modelling tool is the possibility and easiness to replicate sub-structures and interconnect these into a whole model (i.e. “modular” development);
- (c) *repositories of project behaviour* – these refer to the three SYDPIM objects that describe the project behaviour: “PERT/CPM past behaviour”, “PERT/CPM future behaviour” and “Project past behaviour”. These objects consist of simple data repositories, which contain behaviour patterns in the form of data-points over-time. The more effective technological platforms depend on how these objects are being generated within the framework – processes (1), (2) and (3) in figure 7.30. If these processes are being implemented manually, then a spreadsheet tool can be used to store gradually the data-points of project behaviour. If they are being generated automatically by a software routine, which will read from the

PERT/CPM model and from the SYMDB, then these objects can still be implemented in a spreadsheet tool or in a special file produced by this application. The key requirement is that the behaviour patterns produced, and stored in the object, can be accessed by other software routines – e.g. processes (3) and (5) above;

- (d) *SYMDB database* – the purpose of this database is to store various metrics about the project past, over-time. To an extent, this is similar to the previous data repositories which store behaviour patterns. There are however two important differences. First, this database is aimed at providing analysis information, to support diagnosis of past performance and assessment of project status. It needs to be more than just a repository of data. As a primary requirement, it should support analysis features to provide the project manager with insightful information. Secondly, this database is also used as a source of data for the generation of the “Project past behaviour” (process (3) above) and provides the user with metrics to help calibrating the SD model. This database is written with data derived from the PERT/CPM model and (process (4) above). It is also written with data entered directly by the user, and with data derived from the SD model (uncovered metrics). Therefore, it must allow the user to enter data manually, and it should allow the reading and writing of data by other software applications. A spreadsheet application satisfies these requirements. A specialised database application can also be used for this purpose, as far as it satisfies these requirements;
- (e) *analytical links* – the analytical links are at the core of the processes within most SYDPIM activities. As shown in figure 7.30, they are accessed by many of these processes. The analytical links consists of formal specifications, which contain data, mathematical conditions and algorithms. For example, the structural links of work breakdown (see appendix D.1) contain a matrix mapping the tasks of the two models (a data structure), and operators which are implemented through algorithms. These specifications make reference to the variables in the models and so the analytical links must be able to access these variables. On the other hand, many processes within the SYDPIM activities must be able to read the data structures and the conditions within the links, as well as “call” the algorithms. Given these requirements, the ideal technological platform for the analytical links would be a specialised software application. Such an application would have to support the development and update of the

data-structures and conditions within the links, and it would support the implementation of the algorithms. To do this, it would have to be able to read from the models and from other SYDPIM objects. Since this ideal application does not exist in the market, the alternative is the use of a spreadsheet tool, supported by other electronic documents and manual processes. Depending on the number of links established and on the complexity of the models (in particular the SD model), this can be more or less laborious to implement, but still a feasible route;

- (f) *automated processes* – these processes can be implemented as software routines, allowing a much more reliable and timely accomplishment of the SYDPIM activities. There are two main options in terms of technological platform: embed these routines within the software applications that support the SYDPIM objects, or develop them in independent applications. For example, the processes (1) and (2) could be an “add-on” functionality to the PERT/CPM tool, or they could be an independent application that reads from the PERT/CPM files.

The overall ideal for the technological platform of the SYDPIM Project Management Method would be a fully integrated application, incorporating all the SYDPIM objects. This application would automate as many processes as possible, and it would provide the user with an interactive interface for those processes where human judgement is required. It would guide the project manager through the paths of the process logic of the SYDPIM framework, helping to ensure that the activities would be performed in the right sequence, in a timely and reliable manner. This software application is non-existent. The alternative to this is a mix of manual processes and existing software tools. While a less automated solution, this may well be a good approach for an organisation to gradually introduce SYDPIM in their projects. In contrast, a more “sudden” change and full commitment would be required if the integrated application was to be used.

There are two important processes that take place within the SYDPIM activities, which were not mentioned in the discussion above. They refer to the calibration of the SD model and to the readjustment of the PERT/CPM model. These processes take place within each model, and require human judgement. Regarding the calibration of the SD model, there are two main critical issues: validity and time

required. The validity of the calibrations determines the validity of all the analyses provided by the SD model. Achieving a valid calibration in complex models can be very time-consuming, eventually putting at risk the timeliness required. The ideal solution would be an automated calibration process, which would ensure both a valid and quick calibration. As discussed several times in this chapter, this “technology” is not available. The calibration process therefore requires human judgement. Nevertheless, semi-automated algorithms can be of help. In practice, these will consist of optimisation (or “goal-seeking”) algorithms, which refine and improve the accuracy of initial solutions. Regarding the readjustment of the PERT/CPM model to a new project plan developed in the SD model, this can also be a laborious process. The dis-aggregation required leads to various possible solutions, and therefore human judgement is once more required. Automating the process of changing the detailed logical network to represent the aggregate SD plan seems unfeasible, even though some approaches could be considered – e.g. dis-aggregation of data in proportion to a certain criteria; considering “stereotyped” networks for specific industries and project size, among others. To the author’s experience, these automated approaches tend to produce solutions which require extensive rework, not balanced by the benefits. An alternative would be to automate only some sub-steps of the process. However, most PERT/CPM tools available in the market already provide this type of support, so that changes can be easily and quickly implemented in the logical network.

The last requirement listed above for the successful implementation of SYDPIM is the possibility of implementing a gradual change, from the conventional framework (often unstructured) to the SYDPIM framework. For example, the organisation may find out that improvements to the existing framework are the first step required. The author’s opinion is that the vast majority of the potential market SYDPIM users (i.e. organisations carrying out projects and their project managers), is not prepared for a full and comprehensive implementation of SYDPIM. It is perhaps no surprise that most theoretical developments in PERT/CPM based planning, often involving complex mathematics, fail to pass the practical test in the real world and are therefore ignored by practitioners. This reality was the main reason why a *basic mode* for implementing SYDPIM was proposed in this work. This scenario imposes a standard use of the PERT/CPM model and does not require a complex SD project model (even a single-phase model can be used). The real world is that most

organisations that use PERT/CPM follow this standard approach and have never developed a SD project model before. In the basic mode, the implementation of the SYMDB database is recommended but this is not mandatory. The same flexibility applies to the regular update of the PERT/CPM model. The basic mode here proposed for SYDPIM is aimed at providing a bridge that supports the recommended gradual change from the conventional framework to the SYDPIM framework.

While the SYDPIM framework presented in this chapter is comprehensive, there are areas where further theoretical and practical developments would be useful. They would allow the SYDPIM methodology to be refined further, and thereby the development of more sophisticated modes of implementation. The key areas here identified and proposed by the author are the following:

- *development of software tools* – as mentioned above, software tools can play a vital role in improving the timeliness of the SYDPIM activities and of the overall framework. The processes identified in figure 7.30 are good candidates for the development of these tools. Ultimately, an integrated software tool would be the ideal. The automated calibration of the SD model and readjustment of the PERT/CPM model are also processes where software tools would be very useful; however, this also requires theoretical developments to validate the possible automation;
- *theoretical developments* – there are two types of developments which are particularly useful: (i) develop more sophisticated PERT/CPM models, to incorporate and quantify issues typically addressed explicitly in SD project models; and (ii) develop SD project models, specialised for integration with PERT/CPM models. These SD models would incorporate and quantify explicitly various aspects of the network logic of PERT/CPM. As examples, the author has developed some preliminary work in incorporating a measure of quality in the work tasks of a PERT/CPM network (Rodrigues 1994b), and will proceed with this research. On the other hand, a specialised SD model could consider indices that characterise the structure of the logical network, such as “the degree of parallelism” and number of “bottle-necks”. These more complex and specialised models would lead to further theoretical developments on the establishment of analytical links. Other useful areas for theoretical

developments are the automated calibration of the SD model and readjustment of the PERT/CPM network;

- *practical case-studies* – to the author's opinion it is very important to carry out further practical case-studies, from where valuable feedback can be obtained. What really helped / did not help the project? What were the difficulties of implementation? What needs more detail, and what can be simplified? What are the vital improvements? SYDPIM is a practical methodology, aimed at supporting the project management practice in the real world. The voice of this real world is therefore vital, as it will ultimately be the judge of whether this new approach is to be pursued further.

It is also important to stress that SYDPIM is intended neither to exhaust the full range of possible applications of System Dynamics in Project Management, nor to replace the use of any other different tool, technique or procedure, that can provide a useful contribution to the project management process. Other possible applications of System Dynamics, like interactive gaming for training managers, or developing simple models to create learning environments (Morecroft and Sterman 1994), may also provide a complementary and distinctive contribution, outside the scope of SYDPIM.

The discussion presented in this final section was aimed at providing an overview of the SYDPIM Project Management Method, and to discuss the practical and theoretical implications of this new approach. SYDPIM has laid a foundation upon which further developments and research questions need to be pursued.

This foundation consists of a simulation-based project management framework, where the reality of a project is simulated both in the past and in the future. This simulation feature provides the project manager with a laboratory to experiment with the future and diagnose the past. Continuous learning and improved actual decisions result from this process. System Dynamics provides the underlying theory that supports this simulation feature. SYDPIM is aimed at providing a formal mechanism so that maximum value can be achieved from this approach, by integrating it within the conventional framework. This formal mechanism was described in this work as a comprehensive step-by-step logical process. The formal elements of this process were specified in detail. Alternative paths were considered

within the process, providing the required flexibility for this formal mechanism to cope with constraints of reality.

The SYDPIM Project Management Method was developed from a conceptual framework presented in chapter 4. This framework was refined through a case-study undertaken within a real project, which was briefly introduced in chapter 5. During this case-study, SYDPIM was refined based on experiments carried out using real scenarios from the project. Some of these experiments did not interfere with the course of the project. But there were also some situations where SYDPIM was actively used to help managers solving real problems. The next chapter 8 provides a description of some of these practical applications.

8. Practical applications of SYDPIM

8.1 Introduction

The aim of this chapter is to illustrate the practical implementation of the SYDPIM methodology within a real project environment.

The two components of the SYDPIM methodology described in the previous chapters 6 and 7, the development method and the project management method, were developed throughout the KDCOM case-study project, briefly described in chapter 5. For the project management method, an initial conceptual framework was proposed prior to the beginning of the case-study, which was described in chapter 4. This framework was then refined in to a formal method throughout the case-study. Regarding the model development method, the need was only perceived in the beginning of the case-study. An initial outline was developed which was also refined throughout the case-study. The main underlying principle of this research approach was that feedback from the real world was essential in order to successfully refine these methods.

A consequence of this approach is that both SYDPIM methods were not ready to be implemented in full at beginning of the case-study project. Instead, they were implemented, then refined, and then implemented again. This feedback process was repeated several times, in an iterative fashion. As the methods were progressively refined, each time more formality was brought into the process.

It is not feasible to provide here a detailed description of all the practical applications of SYDPIM within this evolving process. The examples here provided were selected from this process. They focus on the following core aspects regarding the practical implementation of SYDPIM:

- *SYSPIM monitoring*: (1) extracting past behaviour from the PERT/CPM plan and from the metrics database, (2) calibrating the SD model to reproduce this behaviour, and (3) SD retrospective analysis to diagnose the project;
- *SYDPIM planning*: (1) extracting the project planned behaviour from the PERT/CPM plan, (2) calibrating the SD model to reproduce this behaviour, and (3) SD “what-if” analysis for risk assessment and improvement of the plan.

Another critical issues regarding the description of these practical applications is the amount of data and information involved as well as confidentiality issues. In particular, a detailed description of the PERT/CPM plans is not provided and some metrics are “disguised”. There were hundreds of these plans and reports being produced throughout the project. For illustrative purposes, summary descriptions are provided.

It is also important to note that the implementation of the analytical links, the transfer of data and the structural readjustment of the PERT/CPM plans were carried out manually. This results from the simple fact that SYDPIM had not been implemented before, it was being implemented in a tentative manner, and hence no support software tools were available. For most needs, a spreadsheet tool was used to collect and re-arrange the data as required (e.g. deriving behaviour patterns from the PERT/CPM plans). In some cases, specialised software routines were used to automatically extract metrics from the project electronic database system (e.g. defects).

In the next section 8.2, a brief description of how the SD project model was developed is provided. This focuses on illustrating the motivation for the underlying principles of the SYDPIM Model Development Method. The following section 8.3 provides a few illustrative examples of the SYDPIM Project Management Method. The first case refers to the monitoring process – according to the case-study strategy described in chapter 5, the first step was to carry out post mortem analysis of SWB1. The two following examples refer to project re-planning: assessing the effectiveness of formal design reviews and assessing the impacts of Client actions (scope changes). These two cases resulted from actual project management needs and were requested by management. The following example refers to the hand-over the SD model to the company at the end o the case-study. The SD model was calibrated to reproduce the behaviour of the second increment (SWB2). The final example was again requested by management some time after the end of the case-study, and refers to assessing the impacts of introducing in the project inexperienced staff from the Client, during the development of a second release of the system.

8.2 Application of the SYDPIM Model Development Method

8.2.1 Overview

At the beginning of the case-study, stage 1 (high-level design) of the C2 software development life-cycle was just starting, and stage 0 (requirements definition) was nearly complete (stage 0 refers to the whole CFCS system, and hence included the apportionment of some of the system requirements to the C2 SW sub-system). The software development stages ahead were therefore development type of work: detailed design and coding.

As mentioned in chapter 5, the overall strategy for model development was to develop a prototype single-phase model, as a generic structure for development tasks. This model would be calibrated and validated against the actual behaviour of the incoming development tasks. Since management had had no contact before with SD modelling this initial model would play an important role, illustrating in practice what a SD project model really was and how it could be used to improve project control. Being able to reproduce actual behaviour, diagnosing results (i.e. providing plausible explanations) and providing valuable insights was an essential achievement.

Prior to the quantification of a project model, the SYDPIM Model Development Method proposes an initial stage of causal analysis, followed by a specification of the model analysis requirements, and then by formal design. However, since the SYDPM model development method was being conceptualised, the development of the prototype model was not preceded by these stages in a formal manner. Some generic causal analysis had been presented to managers before in introductory meetings, but not specific analysis was conducted to identify the particular feedback effects to be captured in the prototype model. Therefore, informal meetings with managers and staff were used to identify the various factors and effects perceived to have a relevant impact on the project outcome. These meetings were also used to clarify the basic analysis requirements of the model. At this stage, these requirements were kept as simple as possible: for example, they did not consider staff turnover in the software development tasks (i.e. a stable team would conduct the work).

Also as part of the overall model development strategy, the aim was to improve this initial structure for development tasks, develop other generic structures (to model other type of non-development tasks, like testing), and eventually use these elementary structures as “building blocks” to develop a more complex multi-phase project model. This raised the need to specify an overall architecture for this multi-phase model. This took the form of a logical network linking various SD-Task (referred to as the SD-TNet, Rodrigues and Williams 1997). It was further necessary to identify the project work scope to be captured in each SD-Task. This was implemented by mapping the tasks in the project WBS to the SD-Tasks. This led to the conceptualisation of a critical analytical link between the SD model and the PERT/CPM model: the structural link for work correspondence (see SC-WBS in chapter 7).

As the initial prototype structure was being improved and other generic structures were being developed, new elements of the SYDPIM Model Development Method were further hypothesised, developed and tested. For example, the generic feedback classification framework presented in chapter 6 (figure 6.4) was developed in this way. Throughout this process, some of the main conclusions and concepts developed were reported in the literature (Rodrigues and Williams 1997, Rodrigues and Williams 1998).

The final SD project model is not described here in detail, for both commercial and confidential reasons. Furthermore, a detailed explanation of the model, equation by equation, would consume a considerable amount of the scope of this research. As mentioned in chapter 5, presenting a comprehensive project model is not the purpose of this research (unlike the work of Abdel-Hamid and Madnick 1991, and of Ford 1995). Nevertheless, an overview description of the model structure along with some of the formal design elements proposed in the SYDPIM Model Development Method are described in the following sub-sections.

8.2.2 Some elements of model design

The SYDPIM elements of formal design, which were described generically in chapter 6, were conceptualised throughout the case study and were also implemented in practice. Some examples are now described.

Feedback structure (stage 0, causal analysis)

The first stage in the SYDPIM Model Development Method is to specify the main aspects of the project feedback structure to be captured in the model. As proposed in chapter 6, this feedback structure may take the form of various influence diagrams, inter-related in a hierarchical way. The identification of the relevant feedback loops and exogenous factors should be guided by the generic classification framework proposed in chapter 6 (figure 6.4). All loops and factors identified should also be explicitly classified according to this framework.

When the initial prototype model was developed, managers were not familiar with describing a project in the form of feedback loops (although some presentations had been made to illustrate this concept). At this stage, it was perceived that the more effective way to identify the project feedback structure was not to go through an exhaustive identification of all the possible closed loops of cause-and-effect. Instead, management were very much aware of important “snow-ball” or “knock-on” general effects like: schedule pressure increases error generation which in turn causes delays and thereby even more schedule pressure. In reality, these “general effects” either correspond to feedback loops or lead to the generation of feedback loops. In other cases important “general risks” or problems were also perceived as relevant and these corresponded to exogenous factors in the SD model.

The strategy was therefore to develop a list of the relevant general effects and problems, and use this information to derive feedback loops and exogenous factors to be captured in the model structure. The SYDPIM classification framework also started to be conceptualised at this stage and the feedback loops and exogenous factors were classified accordingly. Table 8.1 illustrates this process, showing some of the more important feedback loops and exogenous factors captured in the initial prototype model. The first column identifies the general effect or problem reported

by management. The second column describes the underlying feedback loop, which is identified as a balancing loop or as vicious / virtuous circle in case of a reinforcing loop. The signs (+) / (-) are used to identify the type of loop and the type of effect.

General effect	Feedback loop	Classification
As the work progresses through the life-cycle, complexity increases and more errors are generated	<i>Balancing loop (-)</i> : development progress, (+) work complexity, (+) error generation, (+) errors detected, (+) rework needed, (-) development progress	E1
As errors are removed and error density decreases, the effort to detect more errors increases, becoming prohibitive	<i>Balancing loop (-)</i> : error removal, (-) error density, (-) effort required to detect an error, (-) error removal	E1
Due to learning, as the work proceeds less errors are generated by staff	<i>Virtuous circle (+)</i> : development progress, (+) learning, (-) error generation, (+) errors detected, (+) rework needed, (-) development progress	E2
Excessive schedule pressure increases error generation	<i>Vicious circle (+)</i> : schedule pressure, (+) error generation, (+) errors detected, (+) rework needed, (-) development progress, (-) schedule pressure	E2
Staff exhaustion / fatigue increases error generation	<i>Vicious circle (+)</i> : staff fatigue, (+) error generation, (+) errors detected, (+) rework needed, (-) development progress, (-) schedule pressure, (+) extra hours, (+) staff fatigue	E2
Schedule pressure leads to QA cuts and apparent catch-up with progress	<i>Balancing loop (-)</i> : Schedule pressure, (-) QA efficiency, (+) errors detected, (+) rework needed, (-) development progress, (-) schedule pressure	E2
QA cuts lead to more errors escaping to testing, more late rework and even more schedule pressure	<i>Vicious circle (+)</i> : schedule pressure, (-) QA efficiency, (+) errors detected, (-) errors escaped to testing, (+) late rework needed, (-) perceived progress, (-) schedule pressure.	E2
Staff distribute their daily effort between development and rework activities depending on the tasks remaining for each of these activities	<i>Balancing loop (-)</i> : tasks remaining in activity, (+) effort allocated to activity, (+) accomplished tasks, (-) tasks remaining in activity	E2
Staff are willing to work extra hours to catch-up but only up to a limit.	<i>Balancing loop (-)</i> : willingness to work extra hours, (+) actual extra hours of work, (+) staff exhaustion, (-) willingness to work extra hours	E3
In face of a schedule slippage management will consider schedule extensions	<i>Balancing loop (-)</i> : schedule delay, (+) schedule extension, (-) schedule delay	M1
The more the schedule extensions considered, the less management is willing to provide further extensions	<i>Balancing loop (-)</i> : management willingness to extend the schedule, (+) schedule extensions, (+) closeness to max tolerable completion date, (-) management willingness to extend the schedule	M2
Exogenous factors		
Introducing scope changes causes work out of sequence which, with a delay, increases error generation		Ex2
Introducing scope changes may lead to schedule extensions		ExM1

Table 8.1 – Formal design element of casual analysis (stage 0) for the prototype model: some of the feedback loops and exogenous factors identified

Model requirements (stage 1, requirements definition)

According to the SYDPIM Model Development Method, the next stage in the model design phase is to identify and specify the model requirements. These requirements are aimed at ensuring that the model will address the particular aspects and managerial needs of the project at hand. According to the SYDPIM method, these requirements should be grouped in the following categories:

- (1) *range of analysis supported* – this specifies the “what-if” analyses supported by the model and the level of detail. It can be divided into three sub-categories:
 - (i) management decisions;
 - (ii) risk factors (internal and external);
 - (iii) life-cycle phases covered.
- (2) *output produced* – this specifies the output that the model produces to describe and assess the project outcome;
- (3) *input required* – this specifies the input data required to perform the range of analyses specified in the first category above.

Based on preliminary informal meetings with management and staff, some simple requirements were specified for the “development task” prototype model. While simple, this was still a numerous set. Table 8.2 provides a list of the more important requirements regarding the range of analyses supported by the model. Tables 8.3 and 8.4 list the output produced and the input required respectively.

The identification of the range of analyses that the model would have to support was based on the identification of the management needs. Instead of trying to identify the full range of potential analyses that a SD project model could provide and check whether these were relevant, management was asked for a description of the most important question that they would like to ask a “virtual reality” model (like a crystal-ball). These descriptions were then used to identify how they should be translated in the model as “what-if” questions (based mainly on parameter changes). Of course, if any other important potential analysis known to be supported by a SD project model was not identified by management, they would be asked whether it should be included in the list. The descriptions produced by management are summarised in the second column of table 8.2.

Model requirement	Management/staff description
(1) Range of analyses supported	
<i>(1.1) Management decisions</i>	
• Initial planned completion date	Test compressing extending the duration
• Maximum tolerable completion date	Test alternative contingency duration
• Schedule adjustment policy	Test alternative ways of handling delays / pressure
• Initial planned budget	Test alternative effort allocation
• Planned budget fraction for QA	Test alternative QA effort intensity
• Planned budget fraction for rework	Test alternative contingency budgets for rework
• Review period	Test alternative duration for the QA review process
• Desired rework delay	Test alternative rework “waiting-time”
• Rework cut policy	Test the possibility of proceeding without fixing bugs
<i>(1.2) Risks</i>	
• Under-estimation of product size	What-if designs and code are large than estimated
• Variations on productivity	What-if staff productivity is lower due to...
• Variations on cost to rework bugs	What-if errors are harder to rework than normal...
• Variations on cost to detect bugs	What-if errors are harder to find than normal...
• Variations on error generation	What-if work quality is lower due to...

Table 8.2 – List of main range of analyses requirements for the initial prototype model

Regarding the output that the model had to produce in order to support the management needs, a similar approach was followed. Once questions were asked to the model, a project “outcome” would be produced. This time, management were asked about the specific results which needed to be produced by the model in order to describe this “outcome”. The most important results are shown in table 8.3. Perhaps not surprisingly, management answered within four distinctive perspectives which constitute a project’s generic objectives: schedule, cost, scope and quality. Regarding schedule and cost, the various indices shown in table 8.3 (ACWP, BCWP, BCWS, CAC, CTC, CPI, SPI) refer to the conventional project management framework (Nicholas 1990). There was also an additional perspective considered: process metrics. These metrics constitute the core of the “horse-power” of a project organisation in developing a product, and hence they are important to assess the overall organisation’s performance. Overall, this breakdown shows that management was primarily concerned with the project performance in each of the four dimensions of the objectives, as well as with the underlying causes at the product development process level. The breakdown of this category of model requirements (i.e. output produced) in this way, is not proposed explicitly in the SYDPIM Model Development Method (see chapter 6). However, in this particular project the feedback from management strongly suggested these five sub-

categories. This situation illustrates how SYDPIM should be used in practice: a structured basis to be adjusted in a flexible manner to the specific issues of a project.

Model requirement	Management/staff description
(2) Output produced	
<i>(2.1) Schedule related</i>	
• Actual / estimated completion	What was / is the actual / estimated completion?
• Schedule performance indicators: slippage, SPI	What was / is the overall schedule performance?
<i>(2.2) Cost related</i>	
• Cost expenditure indices: ACWP, BCWP, BCWS, CPI	How has cost expenditure evolved?
• Cost estimations: CTC, CAC	What was / is the actual / estimated cost at completion and cost remaining?
<i>(2.3) scope related</i>	
• Work accomplished	How much to the initial scope has been actually accomplished?
• Scope growth	Has scope growth occurred? How much and how?
<i>(2.4) Quality related</i>	
• Cumulative defects detected	How many defects have been detected?
• Current undiscovered defects	How many latent defects there are in the system?
<i>(2.5) Process related</i>	
• Actual productivity	What was the actual productivity achieved? (LOC/person-day)
• Actual defect generation	What was the actual defect introduction rate? (defect/KLOC)
• Actual cost to detect a defect	In average, how much did it cost to detect each defect? (person-hour/defect)
• Actual cost to fix a defect	In average, how much did it cost to fix each defect? (person-hour/defect)

Table 8.3 – List of main output requirements for the initial prototype model

The third and final category of model requirements has to do with the inputs required to calibrate the model for specific scenarios. This is a crucial set of requirements since it imposes conditions of information and data availability. Without this information and data the SD model cannot be used. At the same time, this is generally the area where management is less concerned with: the less questions the model asks in order to produce the desired results, the better. However, if the SD model is to be used in a valid and responsible manner, the person in charge of

model development and calibration (i.e. the modeller) must make a serious effort to identify these requirements.

It is important to note that not all the input data and information required by the SD model must be explicitly available from the real world. Assumptions and deductions can be made and used, as far as there is a valid conceptual foundation or empirical evidence to support them – this is an issue directly related with SD model calibration and validation, discussed in previous chapters. Nevertheless, it is important to identify up-front this required data and information, before model design proceeds into more detail. For those elements that cannot be achieved in a reliable manner, modelling alternatives must be devised.

The main input requirements identified for the prototype model are shown in table 8.4. Unlike with the previous categories, the process used to develop this list focused on the modeller identifying the likely input needs of the model, and then checking with management and staff whether these were feasible. If not, alternatives were discussed. The main concern to keep in mind during this process is to prevent situations in the future where required data is not available or is available too late. The timeliness of the analysis produced by the SD model, and hence of the whole SYDPIM usefulness, depends on how effectively these situations are prevented.

Once more, this category of model requirements was split into four categories. The “initial plan” contains the targets specified for the development task. Most of this data would be directly transferred from the PERT/CPM plan using data exchange links (DEOI-1, DEOI-2, DEOI-3 and DEOI-4; see table 7.5). The prototype model was going to be calibrated for individual development tasks implemented by small and stable teams. The initially planned staff profile was therefore considered to be constant (as in most project models published in the literature), and equals to the planned budget divided by the planned duration. The fraction of the budget initially planned for rework could be extracted from the PERT/CPM plan using the link DEOI-5, in case there were explicitly PERT/CPM tasks classified as such. Since this was not the case, this would have to be an estimate.

Model requirement
(3) Input required
<i>(3.1) Initial plan</i>
<ul style="list-style-type: none"> • Planned completion date • Planned Budget (person-day) • Rework budget (% of planned budget) • Staff profile (daily man-power) • Estimated size (LOC)
<i>(3.2) Development process policies</i>
<ul style="list-style-type: none"> • QA level (% of planned budget) • QA period (weeks) • Rework delay (weeks)
<i>(3.3) Control policies:</i>
<ul style="list-style-type: none"> • Schedule adjustment (in case of perceived delays or early completions how is the completion date adjusted) • Maximum tolerable completion date • Rework cut (in case of delays, defects detected can be passed onto the next phase without being fixed)
<i>(3.4) "nominal" process metrics and learning curves</i>
<ul style="list-style-type: none"> • Productivity (LOC/person-day) • Error generation (defect/KLOC) • Cost to detect (person-hour/defect) • Cost to fix (person-hour/defect) • Fraction of bad fixes (%)

Table 8.4 – List of main range of input requirements for the initial prototype model

The following two sub-categories of inputs constitute policies of product development and project control. For product development, different levels of QA effort could be considered (as a % fraction of the overall budget). The period specified to conduct inspection reviews (i.e. QA / QC) was about one week, but this could also be changed. There was also a desired time to rework detected defects (as also reported in Abdel-Hamid and Madnick 1990), which was of another week. This affects how quickly defects should be fixed once detected. These three policies are simple and intuitive to software development practitioners, and can have considerable impact on the outcome. Regarding control policies, two were considered: adjustment of the scheduled completion date (both extension and compression), and "rework cut". The use of a "maximum tolerable completion date", apart from the planned completion date constitutes part of the schedule adjustment policy. Once the completion date estimated by development staff moves beyond the currently planned date, management "willingness" to extend the schedule depends

on how close this is to the “maximum tolerable completion date (the modelling of this policy is described in some detail in Rodrigues and Williams 1998). “Rework cut” would consist in postponing the rework of detect errors to the next task. This would take place under conditions of considerable schedule pressure.

The final sub-categories of inputs captures the four main process metrics that build-up the “horse-power” of a development team: how fast to develop first time, how well is it developed first time, how hard is to find problems and how hard it is to fix them. A fraction of “bad fixes” is also considered, representing the percentage of errors which are not reworked properly. These metrics are not constant through the life-cycle of a development effort. They are affected by many factors within the project. Some can easily be captured endogenously in the model (e.g. impact of team growth communication overheads). Other factors can be more difficult to captured explicitly in the model and are known to co-relate strongly with progress (e.g. complexity of the work, and “human learning” / “habitual incremental improvement”, Turner 1990). Therefore, two components were considered for these metrics: a “nominal” value and the “learning curves”, which are multipliers that depend on work progress (a similar approach is reported in Abdel-Hamid and Madnick 1990). The “nominal” value is that expected value if these factors are not exerting any influence, and is also often referred to as the “normal” value (although it is arguable whether this is the “normal” situation). The actual final value of these metrics will depend on the nominal value, the learning curve, and on the resulting endogenous influences produced by these curves and other endogenous factors captured explicitly. For example, the final productivity reported at the end of the simulation could be say 40 LOC/person-day and the nominal value of 35 LOC/person-day. In this case, the endogenous influences, which probably varied over-time, would have played an overall positive impact on productivity. It was important to identify this type of required inputs, because reliable means to estimate or derive them were required. However, this is perhaps the most difficult sub-category of inputs to estimate, because metrics available from past developments would always refer to an aggregate which included the nominal values and endogenous curves (e.g. given the actual 35 LOC/person-day observed in past cases, how to conclude that the nominal was 40 LOC/person-day). This is once again, a SD calibration and validation exercise. In this case-study, empirical research available n the literature

was used. Informal meetings with managers were also the base to derive the learning curves and other endogenous effects.

The tables provided in this sub-section are an example of how the requirements stage of the SYDPIM Model Development Method can be implemented. The generic categories of requirements should be used to guide the process, in a flexible manner. If other categories or a further breakdown is appropriate it should be considered. Finally, these lists of requirements should not be regarded as 100% exhaustive, capturing every minor technical detail. Instead, they should be specified under a “management needs” and “model purpose” perspectives. Nevertheless, every important requirement, even if technical, should also be considered explicitly. Overlooking one of these requirements may compromise the model’s ability to respond to management expectations as well as its practical effectiveness.

Model architecture (stage 2, formal model design)

The next step in the SYDPIM Model Development Method is to develop a formal specification of the model design. This consists of a set of formal design elements:

- (1) *model architecture* – a logical network of interconnected SD-Tasks;
- (2) *task classification* – identification of the generic types of tasks used in the model architecture;
- (3) *tasks dependencies* – specifies the dependencies established between the tasks in great detail. Depending on the type of tasks involved this includes inter-task effects and managerial information;
- (4) *control decisions* – identifies the managerial control decisions considered in the model and maps these into the management tasks in the model architecture;
- (5) *mapping of feedback structure to architecture* – maps the feedback loops identified in stage 0 to the individual tasks and to the whole model architecture (loops take place within and across tasks).

Since the first model to be developed was a single-phase model, these elements would be considerably simpler than with a multi-phase model. Nevertheless, an initial model architecture was proposed to represent the whole first increment of the C2 software development project. The development of this architecture was based on two main elements: the product breakdown structure and the life-cycle

development process. These two elements were also the basis to specify the WBS of the C2 software project. The first critical issue was to identify the appropriate level of detail to be considered in the model architecture. SYDPIM advocates that this should be a more aggregate level than considered in the PERT/CPM plan (see chapters 6 and 7): each SD-Task will aggregate various PERT/CPM tasks. At this stage in the SYDPIM model development process, the modeller needs to identify the critical factors that must be taken into account to decide about this aggregation.

The SYDPIM Model Development Method does not impose any particular set of rules and conditions to develop the model architecture. The method suggests that this should be based on the use of three main types of tasks: engineering, management and human resource management (HRM). These tasks should be linked to capture the two main processes of a project: engineering and management. The approach followed in the case-study was based on this dual life-cycle view of the project. This emphasis on the interaction between engineering and management implies that the SD model should capture both characteristics of technical development (e.g. life-cycle phases, product components, rework, work dependencies, reviewing techniques), and the characteristics of the management process. This includes the procedures used to monitor progress and the managerial policies employed in re-planning the work schedules and the allocation of resources.

Regarding the management process, a single high-level project management task was considered. This management task would control the whole set of engineering tasks. However, it was also decided that in some cases, the engineering tasks could incorporate their own management sub-structure (a situation considered in the SYDPIM method; see chapter 6). This project management task would also control a single project-wide HRM task, which captured recruitment and training issues and staff turnover. In this way, each engineering task continuously reports progress to the high-level project management task. This information includes work progress, estimated completion date, estimated cost at completion, and man-power needed to meet the schedule. Based on this information, the project management task mimics the high-level decision-making in the project. This can include: (1) adjusting the completion schedules of the individual tasks, (2) re-schedule staff among the tasks, (3) increase the degree of overlapping between sequential tasks, and (4) hire more staff into the project. This modelling decision to represent the management process

results in not representing explicitly the management hierarchy in the model architecture. The main reasons for this had to do with the fact that, as mentioned in chapter 5, the project organisation was very much a *pure* project organisation – i.e. the OBS tailored for the WBS. Therefore the breakdown of the management hierarchy coincided with the work breakdown of the engineering process. The project teams working in each of the major product components and life-cycle phases were given a high level of authority to plan and control their work. On top of this work breakdown oriented management hierarchy there was the software management team. Human resource management was also performed by a single management team.

Regarding the engineering process, the breakdown into subtasks assumes that each of these tasks holds enough complexity so that planning and control might be problematic. According to SYDPIM, each task is simulated by an individual SD sub-model with a structure specialised on the type of work being performed. Three main generic types of engineering tasks were considered: (1) development, (2) integration, and (3) testing. A development task refers to the process of developing a particular sub-product, and includes the activities of developing, reviewing and reworking (design, coding fall in this category). An integration task refers to the process of transforming a set of input sub-products into a single final product (system integration falls in this category). A testing task refers to the process of running a set of pre-defined tests to check the functionality of a sub-product, diagnose faults, and rework the defects found (system testing fall in this category).

The breakdown of the engineering process into sub-tasks was based on three main factors: the life-cycle definition, the product structure, and the structure of the technical development teams. The lowest level of decomposition would consider a single task for each life-cycle phase of each software sub-component, being implemented by an individual software team. For most software projects such level of detail, very close to the network plan, might not be appropriate. However, this decomposition was used as a starting point. The next step was to aggregate these elementary sub-tasks into more complex tasks. There were two different types of aggregation:

- *horizontal aggregation* – ignoring the intermediate schedules of a set of sequential tasks and consider them as a single task with a single schedule and budget;
- *vertical aggregation* – ignoring the individual schedules of a set of parallel but inter-related tasks and consider them as a single task with a single schedule and budget.

The definition of the appropriate level of decomposition/aggregation of the engineering process was based on four main criteria: (1) type of work being performed, (2) major schedule milestones, (3) level of detail required for management purposes, and (4) minimum level of complexity required. The type of work being developed within a task should be of similar nature and strongly inter-related. As an example, the design and testing of a software component might not be appropriately captured by the same task, while the design of two tightly coupled software components may be aggregated into a single design task. The major schedule milestones of the development process should impose limits on the level of horizontal aggregation so that they are not overlooked in the model. In general, most managers would like a decision-support model to consider explicitly the individual schedules of the major life-cycle phases. Finally, each task must comprise a minimum level of complexity so that the use of a SD model is appropriate – System Dynamics models focuses on the interactions among a system's components; decomposing the software development process into highly detailed elements tends to eliminate the effects of these interactions, and the individual behaviour of such elements becomes characterised by discrete uncertain events, to which the higher-level continuous perspective of a SD model is not appropriate.

The final model architecture regarding the engineering process is shown in figure 8.1. In terms of vertical aggregation, the development of the nine software sub-components was aggregated into three major functional areas (Radar, SSGT, and Core). This was because the development of the components within each of these areas was tightly coupled. In terms of horizontal aggregation, the nine different stages of the C2 SW life-cycle were aggregated into five main development phases (design, coding, group testing, integration, and system testing). This resulted of the type of work being performed and management interest on intermediate schedules.

Overall this resulted in eleven SD-Tasks, each holding enough complexity to be modelled by an individual SD structure. Six of these tasks were modelled by a generic “development task” SD generic structure, one modelled by an integration structure and four by a testing structure.

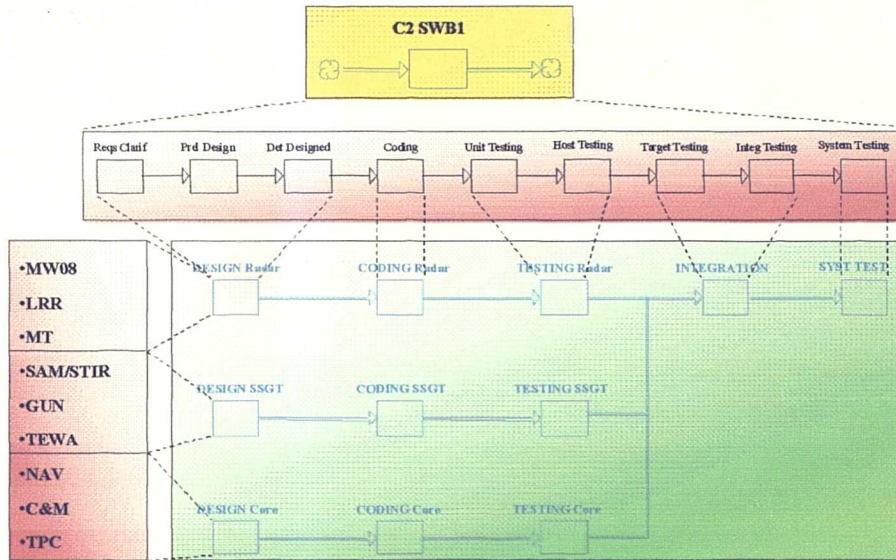


Figure 8.1 – Model architecture for the engineering process of the C2 SW project

The overall model architecture, linking the engineering process with the management process, is shown in figure 8.2. Two more tasks were added: high-level project management and project-wide human resource management.

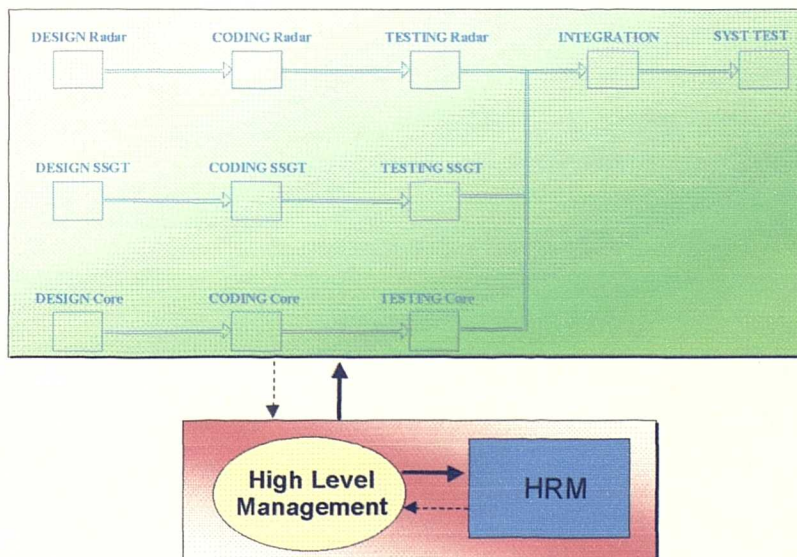


Figure 8.2 – Model architecture for the C2 SW project model

Other elements of formal design (stage 2)

Once the model architecture is developed, the other elements of formal design can be specified: task classification, tasks dependencies, control decisions and mapping of causal structure to the model architecture.

According to the SYDPIM development method, the aim of the task classification is to identify a set of generic tasks which will be modelled by a specific and also generic SD sub-structure. Each of these structures will be tailored to model one or more actual project tasks in the model architecture. The model architecture proposed for the C2 software development project (first increment), as shown in figure 8.2, considers five generic tasks which are proposed in the SYDPIM Model Development Method as a framework to develop the classification matrix (see table 6.7 in chapter 6):

- ETPD – engineering task of product development/transformation;
- ETPT – engineering task of product testing;
- ETPI – engineering task of product integration;
- MCMT – management control of multiple tasks;
- MCST – management control of single task;
- HRM – human resource management.

The first three are engineering types of tasks and the other three are management type of tasks (the last is HRM). Note that while the architecture proposed does not consider explicitly any management task that controls a single engineering task, the type MCST was considered because, as mentioned, all engineering task will contain their own internal management process. Table 8.5 shows the resultant task classification matrix developed. All design and coding tasks were classified as ETPD. Testing tasks of individual functional areas and of the whole system were classified as ETPT. All of these engineering tasks were also classified as MCST, because of containing an internal control process. The high level management task controls all engineering tasks and hence was classified as MCMT. The global HRM task was of course classified as HRM.

	ETPD	ETPT	ETPI	MCST	MCMT	HRM
Engineering Tasks						
Design Radar	✓			✓		
Design SSGT	✓			✓		
Design Core	✓			✓		
Coding Radar	✓			✓		
Coding SSGT	✓			✓		
Coding Core	✓			✓		
Testing Radar		✓		✓		
Testing SSGT		✓		✓		
Testing Core		✓		✓		
Integration			✓	✓		
System Testing		✓		✓		
Management Tasks						
High Level Management					✓	
HRM Tasks						
Project HRM						✓

Table 8.5 –Task classification matrix developed for the C2 SW project

The next step in formal design is to specify the effects and control information to be captured within the tasks' dependencies in the model architecture. The SYDPIM method proposes two types of links: control information-flow, and physical product-flow. The first type of dependencies can involve engineering and management tasks, while the second involves only engineering tasks. SYDPIM suggests that four main types of effects can be captured in a physical product flow dependency: technical requirements, work quality, QA effects and rework accomplishment (see table 6.8 in chapter 6). Regarding control information-flow between management tasks and between management and engineering tasks, a framework to identify the possible information contents is also suggested (see table 6.9 in chapter 6), which considers a fairly exhaustive list. The model architecture developed (figure 8.2) considers overall 22 dependencies:

- 10 product-flow dependencies between the engineering tasks;
- 11 information-flow dependencies between the high-level management and the engineering tasks;
- 1 information-flow dependency between the high-level management task and the HRM task.

The specification of the dependencies has a great impact on the model calibration. Furthermore, the validity of this formal design element requires a considerable amount of input from management and technical staff. Since a prototype single-phase model was being developed at this stage in the case-study, the detailed specification of these dependencies was postponed. It was felt that the feedback from applying the prototype model was needed, so that management and staff would develop a much thorough understanding of this design element. The final design element for the multi-phase model is not presented in this work (nor the model itself). On the one hand it is not the purpose of this research to focus on the detailed description of this extensive model. On the other hand, this would be a lengthy description which could jeopardise the illustrative purpose of this chapter. Confidentiality issues also prevent a detailed publication.

The following formal design element to be developed is the specification of the project control decisions generated within the management tasks. SYDPIM considers that this included both: (i) the identification of the specific control decisions captured in each management task in the model architecture, and (ii) the classification of how the decision is modelled: endogenous, exogenous or transmitted (see chapter 6). To support the development of this design element, SYDPIM proposes a base list of generic control decisions (see table 6.10). For the same reasons just mentioned above, this design element is not described here in detail for the multi-phase model. At this stage in the case-study the development of this design element was also postponed.

For illustrative purposes, this is described for the prototype single-phase model (see also example in table 6.11 of chapter 6). This model represents a generic "development task", and its internal management component would capture the decisions which are implied in the requirements specified in stage 1 (table 8.2). This generic structure would be calibrated and tailored to represent any of the 6 development tasks in the model architecture. The specific control decisions considered by management in each of these tasks was the same. These are shown and classified in table 8.6. Most of the decisions refer to the development of the initial plan for the task and are exogenous. Four endogenous decisions can take place to control the work while the task is on-going: (i) extending / compressing the

planned completion date, (ii) re-allocate the man-power available among the three development activities of development, QA and rework, (iii) reducing the QA level and (iv) cutting rework of defects. These three decisions are mainly aimed at handling delays. As previously mentioned, changing the staff level was not considered because the work in each of the development tasks (as in the model architecture of figure 8.2) was being carried out by stable teams.

Management decisions	Type of decision
<i>(1) Initial plan</i>	
• Initial planned completion date	EX
• Maximum tolerable completion date	EX
• Initial planned budget	EX
• Planned budget fraction for QA	EX
• Planned budget fraction for rework	EX
• Review period	EX
• Desired rework delay	EX
<i>(2) On-going re-planning</i>	
• Adjustment of planned completion date	ED
• Re-adjust man-power allocated to the activities	ED
• Adjustment of QA level	ED
• Rework cut policy	ED

Table 8.6 – Control decisions design element for the prototype model

The last formal design element to develop is the mapping of the causal feedback structure specified for the project in stage 0, into the model architecture. This consists in specifying which feedback loops and exogenous factors identified in stage 0 are to be captured in each of the tasks in the model architecture (i.e. intra-task effects), as well as across these tasks (inter-task effects). The SYDPIM method proposes that this design element is comprised of seven matrices to be developed as follows:

- (1) *intra-task engineering matrix* – feedback loops within each engineering task internal to the engineering process (i.e. E1, E2, and E3 types);
- (2) *inter-task engineering matrix* – feedback loops across several engineering tasks internal to the engineering process (i.e. E1, E2, and E3 types);
- (3) *intra-task management matrix* – feedback loops within each management task internal to the management process (i.e. M1, and M2 types);
- (4) *inter-task management matrix* – feedback loops across several management tasks internal to the management process (i.e. M1, and M2 types);

- (5) *engineering-management matrix* –feedback loops that cross through both engineering and management processes (i.e. EM1, EM2, and EM3 types);
- (6) *engineering exogenous factors* – exogenous factors within each engineering task internal to the engineering process (i.e. ExE1, and ExE2 types);
- (7) *management exogenous factors* –exogenous factors within each management task internal to the management process (i.e. ExM1, and ExM2 types).

Like with the previous design elements, this is not described here in detail for the multi-phase model. Regarding the prototype single-phase model, all the feedback loops and exogenous factors identified in stage 0 (some examples show in table 8.1) are to be captured in this SD generic structure – furthermore, management assumed that the same feedback effects and exogenous factors were relevant and would take place in all of the 6 development tasks in the model architecture. Because this is a single-phase model, matrices (2) and (4) are not specified. Matrix (5) will identify the engineering-management loops that take place between the engineering and internal management components of this SD structure. For a single-phase model, the development of the five matrices will consists in grouping accordingly, all feedback loops and exogenous factors identified in stage 0 (as in table 8.1) (see example in table 6.12 of chapter 6). For a multi-phase model, the seven matrices above are to be developed. Some of the feedback loops and exogenous factors identified in the design element of stage 0 may be mapped to more than one task.

In this sub-section, the implementation of the model design phase of the SYDPIM model development process was illustrated through the specification of some of the main design elements, regarding the SD project model developed in the case-study. Due to scope restrictions, some of these elements were specified only partially while others were simplified as referring to the prototype single-phase model. The purpose was to provide a practical exemplification, based on the case-study, of how the SYDPIM Model Development Method can be implemented. It should be stressed that throughout the case-study, the method was conceptualised and continuously refined and hence it was implemented in an iterative manner.

8.2.3 An overview of the implementation phase

Once the formal design stage is complete, the SYDPIM model development process proceeds to the second phase of implementation and validation. Here the SD project model is developed and quantified in the form of a “stock and flow” diagram, using an appropriate SD simulation language (see discussion in chapter 6). The SYDPIM method proposes a life-cycle of well ordered stages within this phase as follows (see figure 6.12 and algorithm 6.1) :

(4) component development:

- (a) development of generic super-structures;
- (b) tailoring of super-structures to project tasks;

(5) progressive integration of project tasks;

(6) final model validation.

In each stage of these stages, specific formal design elements (developed in the previous phase) provide the required input (see figure 6.13). Furthermore, a critical characteristic of the SYDPIM model development process is that, although there is a final model validation phase, a validation activity is carried continuously throughout this implementation life-cycle. For each stage, the SYDPIM method proposes specialised validation tests to be carried out (see figure 6.14).

The first stage is the development of the required generic super-structures and tailoring for the project tasks. In this case-study, the model design phase identified six of these structures as shown in table 8.5. This SYDPIM stage, various sub-steps are undertaken s formally described in algorithm 6.2:

- (i) for each super-structure produce a list of requirements;
- (ii) specify a reference scenario for each super-structure;
- (iii) implement and validate super-structure;
- (iv) replicate, tailor, calibrate and validate each super-structure for the various project tasks it will model.

The task classification matrix developed for the C2 software project model of this case-study shows that overall 6 generic structures will be tailored to model 13 project tasks. For example, the generic super-structure ETPD will be replicated and tailored to model 6 project asks (i.e. design and coding tasks).

The first step is to produce a list of requirements for each super-structure, based on the design information regarding the project tasks it will model. This includes: processes incorporated, links to other tasks, control decisions, and internal feedback loops and exogenous factors (see table 6.17). For example, the ETPD super-structure will have to capture all the characteristics of the 6 tasks it will model. These requirements can be represented as shown in table 6.17. For reasons of scope and confidentiality, this will not be described here in detail for the multi-phase model. Regarding the single-phase prototype model, this was a generic structure with two main (also generic) sub-components: ETPD (product development) and MCST (management control). The requirements for this generic structure are basically all that was identified in the design phase:

- processes incorporated: engineering and management;
- links to other tasks: not considered;
- control decisions: as in tale 8.6;
- internal feedback loops and exogenous factors: as identified in stage 0 (see table 8.1).

The next step in the SYDPIM development method is to specify a reference scenario of behaviour for the calibration and validation of each super-structure. According to SYDPIM this should have two main components: (i) the input characteristics, which refers to the various initial conditions in particular productivity related estimates and the planned targets, and (ii) the behaviour patterns to be produced, which can included various modes but must include a steady behaviour (i.e. targets achieved with minor variations if any). SYDPIM recommends that the mode of behaviour specified for the generic super-structures does not correspond to any of the specific tasks it will model. Instead, a fictitious or historical behaviour of similar tasks should be used. In the case-study, the company had not records of historical behaviours readily available for any of the generic super-structures. Therefore, a fictitious simple scenario was used, where a steady behaviour was to be reproduced (e.g. 100 tasks to be accomplished by 10 persons, within a 1000 person-day budget and a 100 days duration) – this is the recommended approach whenever historical scenarios are not available.

The following step in the SYDPIM model development method is to implement and validate each super-structure. The multi-phase model required six different super-structure to be developed. It is not the purpose of this research to describe here in detail the implementation of this complex model nor even of the prototype model (which contained over 300 equations). An illustrative example of how the SYDPIM implementation stage should be carried out has been presented in chapter 6. For illustrative purposes regarding the case-study, a brief overview of the core structure of the physical processes is here presented regarding the ETPD (development) and ETPT (testing) generic super-structures. The quantification of these structures is not presented.

As described in chapter 6, for the implementation of the super-structures the SYDPIM method suggests a sequence of steps based on the proposed generic project feedback structure (see figure 6.4). The steps to be followed will vary slightly for each of the three types of elementary tasks proposed in the SYDPIM method: engineering, management and HRM. The generic super-structures ETPD and ETPT are engineering type of tasks. SYDPIM suggests that this type of tasks will capture four main processes: (1) work accomplishment, (2) monitoring progress against exogenous targets, (3) reporting progress to external management, and (4) adjusting their behaviour in order to achieve the targets. It is proposed that these processes should be implemented progressively, according to the following sequence of steps:

- (1) implementation of physical processes;
- (2) implementation of E1 type of feedback loops and ExE2 exogenous factors;
- (3) implementation of E2 and E3 type of feedback loops and ExE1 exogenous factors.

Figure 8.3 shows an initial version of the core physical processes developed for the ETPD super-structure. The basic principle behind this structure is that the engineering process of a development task comprises the continuous life-cycle flows of two inter-related entities: product (e.g. designs), and defects. As the work is developed, reviewed and reworked throughout its life-cycle, in parallel, defects are also generated, detected, and reworked. However, some defects might not be detected by the reviewing activity and hence they escape, incorporating the sub-product delivered onto the next task. The stocks, rates and auxiliaries in blue,

represent the “feedback-absent” physical structure of work accomplishment – this is the output of the first step suggested to implement an engineering super-structure (also illustrated in figure 6.15 of chapter 6). The two flows of work and defects are inter-related, and some auxiliaries are use to implement the required relationships (e.g. “error density” type of variables). The variables in green which are linked to this physical structure through red arrows represent temporary exogenous inputs. In the final model, these variables are generated elsewhere and may originate endogenous feedback effects of types E1, E2, and possibly of types ME1, ME2 and ME3. This structure is well suited to represent the product development process of an individual task and it was therefore used as the basis for the single-phase model.

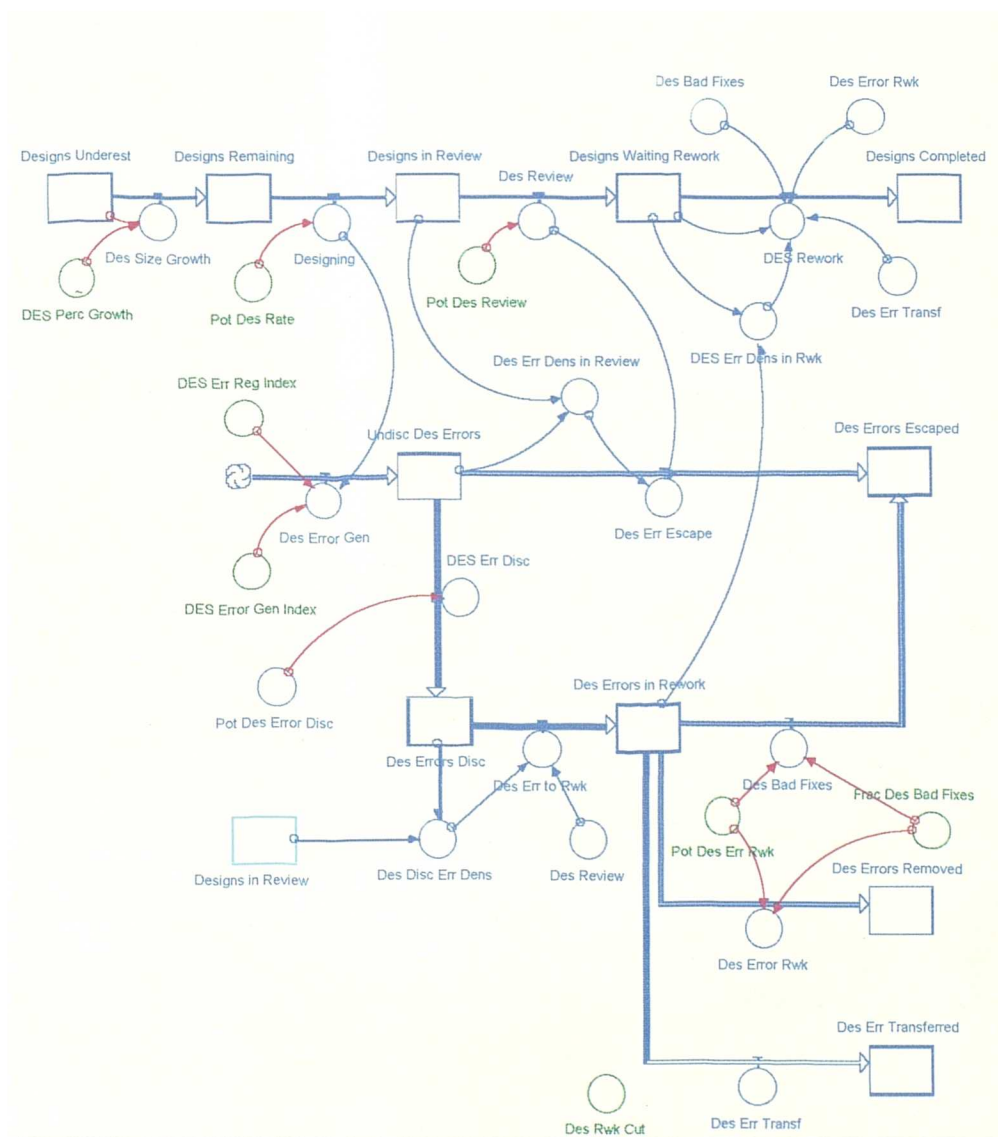


Figure 8.3 – Core physical process of the ETPD super-structure developed

However, for the multi-phase model this structure did not capture all that was necessary to represent a development task inter-acting with other tasks. This was particularly true for design type of tasks. For example, according to the structure above, designs are delivered to the successor tasks (say coding) with undiscovered defects. Once coding starts, these design defects will be detected and will need to be reworked. This may happen after all the designs have been delivered to coding, and hence when the design phase is considered as complete. However, these design defects are not the same as the coding defects, which are generated within the coding task where they will also be detected and reworked. Design defects are different than the coding defects and furthermore they are not necessarily reworked by the coding team. This way, there was the need to consider “post-delivery” defect detection and rework for the design tasks, possibly carried out by the design team. On the other hand, for a coding development task this “post-delivery” rework process was not needed because, unlike with the design defects, coding defects were “physically” detected and reworked in the following testing and integration tasks. This situation highlights an important issue regarding the structural requirements of the ETPD generic super-structure: when tailored to coding tasks, this “post-delivery” process would have to be “disabled” (possibly through an “on/off” switch”).

In order to capture this “post-delivery” rework process, the initial structure of figure 8.3 was refined improved. There can be various to solutions to this. The major problem that needs to be solved is the conflict between a continuous post-delivery rework and the establishment of a fixed completion date for the task. In practice, post-delivery rework takes place throughout the remaining product development life-cycle, possibly until the end of the project. On the other hand, life-cycle phases have intermediate end-dates. Therefore, how to consider a task completed and then accept that rework continuous until the end of the project? Should “post-delivery” rework force the task to “re-start again”? This is an existing problem in project modelling and to the author’s knowledge it has not been solved in a clear and explicit manner. In some cases, “indirect” solutions have bee used. These based in the idea that while the work in the task can proceed until the end of the project, there can be “indirect symptoms” that 100% of the product has already been delivered to the following phases (e.g. when x% of the planned budget is

consumed). There are various problems with these solutions. First the definition of the “indirect symptoms” often requires an arbitrary exogenous input, which raises validation difficulties. Secondly, this approach will generally not distinguish between pre-delivery “genuine” rework and post-delivery rework. Finally, the overall solution is less explicit and thereby less clear to the model user. Generic structures available in the literature are based on this approach. For example, the simple “rework-cycle” proposed by Cooper (1993), which contains only four stocks/levels. An explicit and robust solution to capture the “post-delivery” rework phenomenon requires considerations at the structural level. The starting point is a clear distinction between the concepts of “phases” and “activities”. Software development phases have precise start and completion time-points, whereas activities take place continuously throughout the life-cycle. For example, once the design phase is complete, the designing activity proceeds until the end of the project. In this way, the “rework cycle” generic structure (Cooper 1993), which considers that rework accomplishment takes place continuously throughout the project, represents activities much better than it may represent phases: once a “rework-cycle” starts, it only finishes by the end of the project. This prevents the explicit consideration of intermediate schedules. It should be noted that this limitation is not surprising given the simplicity of the “rework cycle” structure, which has only four levels/stocks.

Although it was not within the scope of this research to develop an innovative SD structure to model the software development process, the discipline of the SYDPIM model development, as well as its emphasis on validation and quality, has motivated a careful analysis to this outstanding modelling issue. Figure 8.4 shows the final SD structure for the physical process of an ETPD task – the auxiliary variables in black represent temporary exogenous inputs to the structure; the information flows in light grey refer to the control of non-negative stocks/levels, and are less relevant under a conceptualisation perspective. As suggested by the two sectors shown, the structure comprises two main components: initial development and product delivery, and post-delivery defect detection and rework. The first component is the same as the initial structure presented in figure 8.3, with some improvements which includes scope changes and an intra-task work progress dependency. These two aspects could be relevant in certain scenarios and were not captured in the initial version. The scope changes considered refer to pre-delivery of the designs (i.e. while they

are being developed); post-delivery scope changes is another interesting and complex modelling issue which is not here presented and discussed.

As a result of implementing the SYDPIM Model Development Method, an improvement to the existing structures used in past developments was achieved. It is argued in this research that this is a most valuable benefit of adopting a structured development process, such as the method proposed in this work .

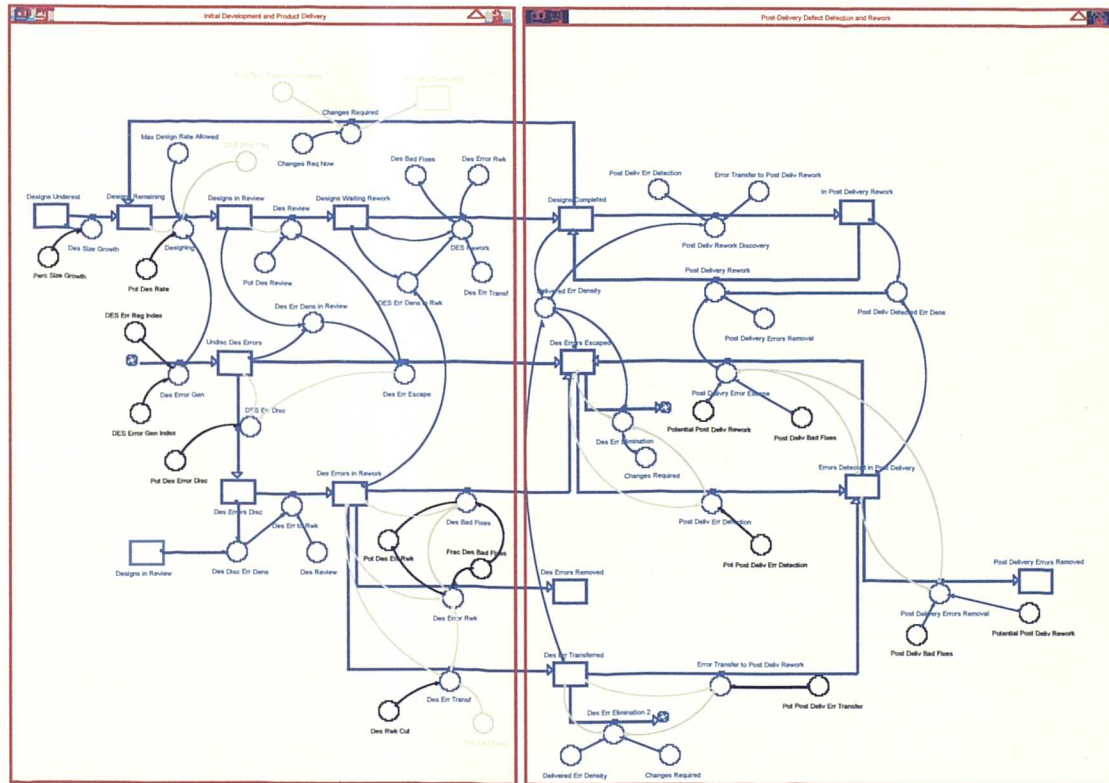


Figure 8.4 – Core physical process of the ETPD super-structure incorporating post-delivery rework

Figure 8.5 shows an initial version of the core physical processes developed for the ETPT super-structure (a testing task). The basic principle behind this structure is that the engineering process of a testing task comprises the continuous life-cycle flows of three inter-related entities: tests, product (i.e. code), and defects. As the test cases re prepared, run and failed or passed, the code of the software product becomes progressively tested. In parallel, defects are caught by the test cases that fail while others escape (note that tests do not ensure 100% of defect removal). As

failed tests are diagnosed, some defects are detected and removed, while others escape and will probably cause further testing to fail again. This testing task delivers a certain amount of code tested with a certain amount of undetected defects depending on the testing efficiency.

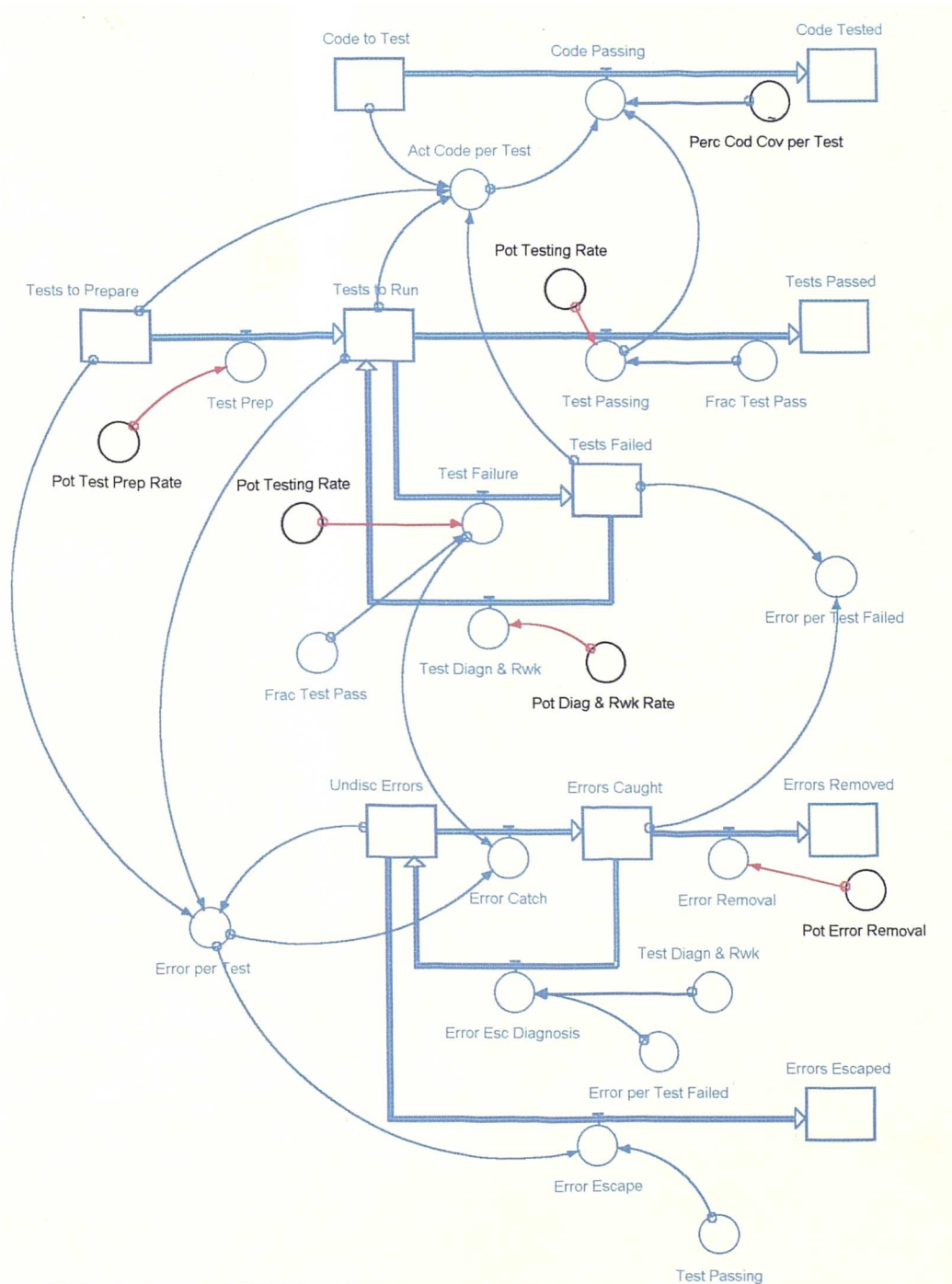


Figure 8.5 – Core physical process of the ETPT super-structure developed

The structures shown in figures 8.4 and 8.5 constitute the physical processes of two of the six generic super-structures to be developed for the multi-phase model of the case-study (i.e. ETPD and ETPT; see table 8.5). The next step in the implementation of these structures was to build progressively upon this structure the various types of feedback loops and exogenous factors, as suggested by the SYDPIM method (see chapter 6). The reference scenario of behaviour specified for each of these structures is used to validate this continuous implementation process. Once the required super-structures are developed the next step on the SYDPIM Model Development Method is to replicate and tailor them to the specific project tasks, as identified in the task classification matrix of figure 8.15. At the end of this component development stage there will be one SD structure per project task in the model architecture. The next stage in the SYDPIM method is to carry out a progressive integration process based on the tasks' reference scenario. The final step focuses on behaviour reproduction validation based on a specified project reference scenario, which may include past behaviour, planned behaviour and fictitious behaviour modes.

Regarding the prototype single-phase model, this consisted in coupling the ETPD structure with the MCST structure (not shown here). Although this was a simpler development, it followed the SYDPIM development life-cycle – e.g. this coupling corresponds to the integration stage 5. A snapshot of the implementation of this model using the iThink modelling tool is shown in figure 8.6. As already mentioned, the SYDPIM method was conceptualised based on feedback from the tentative development of the two models. The method was not available up-front to support the sequential development of these models. It was therefore implemented in an iterative manner, with its underlying principles being continuously tested and refined.

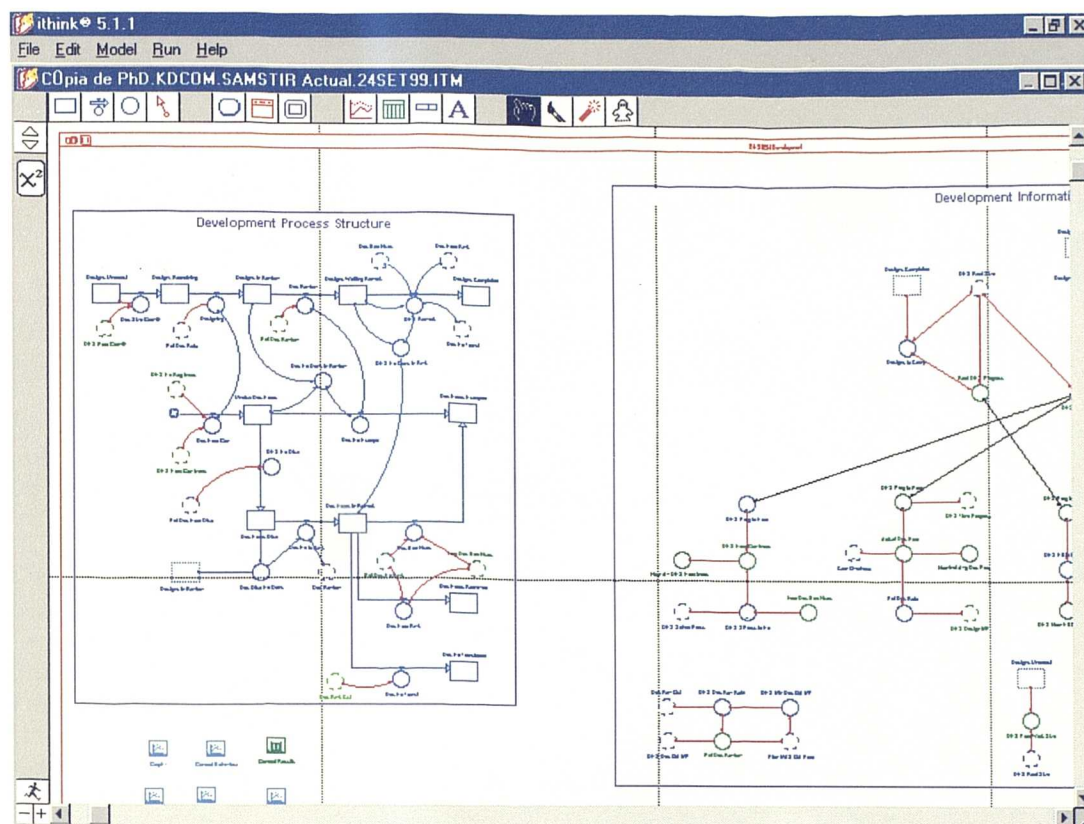


Figure 8.6 – A snapshot of the prototype single phase model implemented in iThink

The aim of this sub-section was to illustrate how the SYDPIM Model Development Method guided the development of the SD project models in the case-study, and in general how the method can be implemented within a real project context. The following section describes illustrative practical applications of the SYDPIM Project Management Method.

8.3 Applications of the SYDPIM Project Management Method

8.3.1 Overview

In this section some practical implementations of the SYDPIM Project Management Method are described. These were carried out in the course of the case-study: a SD project model was used to support the on-going planning and monitoring of the C2 software project. The aim of these examples is to illustrate how the generic SYDPIM Project Management Method, as described in chapter 7, can be implemented in practice within a real project.

Like with the model development method, the SYDPIM Project Management Method was progressively refined throughout the case-study. It was therefore implemented in an iterative manner, each time incorporating more detail. At the beginning of the case-study there was a conceptual high-level framework available, which was described in chapter 4. Further details were then conceptualised throughout the case-study, in particular the establishment of the analytical links between the SD and the PERT/CPM models (some preliminary work and conclusions were reported in Rodrigues and Williams 1997). This process of continuous refinement led to the formal SYDPIM Project Management Method presented in chapter 7.

In this way, the SD model was applied in several occasions throughout the case-study, prior to the final method being specified. However, for illustrative purposes, the examples here reported will describe the implementation of SYDPIM as close as possible to the final logical flow, as presented in chapter 7. The SYDPIM planning and monitoring process logic comprises a series of steps, wherein the SD and PERT/CPM models are used and sometimes exchange data. In the practical examples here presented not all of these steps will be described in detail. On the one hand, this would lead to a lengthy description, involving large amounts of data, and probably threatening the desired illustrative nature of the examples. On the other hand, most of the data and PERT/CPM plans cannot be here presented for reasons of confidentiality. This way, the description will focus on those aspects which are crucial to understand the practical issues of the method.

The examples here described refer to situations where management explicitly requested the use of the SD model. This is according to the strategy established for the case-study, as discussed in chapter 5. The first example presented refers to the diagnosis of the design phase of a sub-component of the C2 software system. This is also according to the strategy adopted for the case-study, where the initial applications of the model would focus on the reproduction and post mortem analysis of past stages of the project. The second example refers to future planning, although it is also based on the reproduction of the project past. Here, the SD model is used to investigate the potential impacts of replacing informal design and code reviews by a more formal reviewing technique called "Fagan inspections" (Fagan 1976, 1986). Management needed to know whether the actual benefits would outweigh the costs, what were the risks and what were the more favourable

re-planning scenarios (e.g. how much should design be extended to accommodate the extra effort required). This is a case where management pro-actively tries to improve the project plan. The next example is also about SYDPIM planning, but this time it refers to a risk analysis scenario. Management was worried with potential Client actions, in particular with potential scope changes. If requests for changes were made, management needed to know how to re-negotiate schedule extensions effectively. The next example refers to the calibration of the SD model to reproduce the project plan developed for the second increment of the C2 software project. This took place at the end of the case-study, when the model was passed onto the planning team. This is also a case of SYDPIM planning, when a new PERT/CPM plan is transferred to the SD model. The last example refers to the application of the SD model after the C2 SW project was complete, when the development of a second release of the system was to be initiated. This application refers to another case of risk analysis and re-planning. As part of the contractual agreement, the transfer of know-how to the Client implied involving some of their inexperienced staff in the project. This would require training and would divert the attention from the experienced company staff. Management needed to know how to best re-plan the work and re-allocate resources, so that the disruptive impacts were minimised.

These examples are now described separately in the following sub-sections.

8.3.2 SYDPIM Monitoring: diagnosing the design phase of a system component

In this example, the SD model is used to diagnose the design phase of a system component after this has been complete. This includes stages 1-3 of the C2 software development life-cycle described in chapter 5. The single-phase SD prototype model was used to model the design phase as a SD-Task in isolation from the rest of the project.

The SD analysis involves the following actions: calibrating the model to reproduce the past behaviour observed, use the model to identify causes for eventual deviations from the plan, and carry out retrospective “what-would-have-happened-if” analysis for process improvement. However, before any of these “SD analysis” type of actions can be carried out, according to SYDPIM, there are some previous

actions than need to be implemented. This is a situation where progress is being monitored and hence SYDPIM monitoring should be implemented.

As described in chapter 7, the process logic of SYDPIM monitoring can follow alternative paths depending on whether the PERT/CPM model is being updated regularly and whether the SYMDB is being implemented. In this case, both metrics were being collected and the PERT/CPM model was being updated regularly. In this situation the SYDPIM monitoring process logic comprises the following activities:

- (M1a) collect project result metrics
- (M2a) update PERT/CPM model and SYMDB with result metrics;
- (M3a) update analytical links if necessary;
- (M4) extract project past behaviour from PERT/CPM model;
- (M5) specify project past behaviour;
- (M6) re-calibrate SD model for past behaviour;
- (M7) SD diagnosis of project past;
- (M8) update SYMDB with uncovered SD metrics.

The implementation of these activities required that structural and data links were established between the PERT/CPM and the SD model. The appropriate PERT/CPM tasks were mapped to the SD-Task being simulated by the prototype model. Resources working in this design phase were also mapped to the SD-Task. The standard data-links proposed in chapter 7 were also established.

These monitoring activities are now described separately.

(M1a) collect project result metrics

In this activity, metrics regarding the actual results are collected from the project on a time-point basis. This is, while the design phase of this component was progressing these metrics were collected progressively in every control period. The control period adopted was the month. The specific metrics to be collected in this activity are shown in table 7.15 (see chapter 7). This considers metrics at three levels of detail: (1) project level, (2) PERT/CPM task level and (3) SD-Task level. In this example, there is only one SD-Task to consider which refers to the design

phase of the component being developed. In the PERT/CPM plan, various tasks were mapped to this SD-Task through the establishment of structural links (the actual PERT/CPM plan is not shown here, but it contained about 10 tasks). The first two categories of metrics were already being collected by the planning team as part of the established PERT/CPM procedures, and are not shown here. The metrics collected at the SD-Task level are shown in table 8.7 below.

Days (cumulative)	0	21	42	63	84	89	Unit
SYMDB metrics							
(3) SD-Tasks							
(3.1) defects detected	0	0	38	73	94	0	defect
(3.2) defects reworked	0	0	38	56	45	66	defect
(3.3) cumulative defects detected	0	0	38	111	205	205	defect
(3.4) cumulative defects reworked	0	0	38	94	139	205	defect
(3.5) defects awaiting rework	0	0	0	17	66	0	defect

Table 8.7 – Metrics collected at the SD-Task level in activity (M1a)

Note that these are defect metrics and hence have been “disguised” for reasons of confidentiality.

(M2a) update PERT/CPM model and SYMDB with result metrics

This activity is aimed at updating the PERT/CPM model and the SYMDB with the actual results. As described in chapter 7, this involves three steps:

- (1) update the PERT/CPM current plan and SYMDB with the result metrics collected in activity (M1a);
- (2) update the SYMDB with metrics derived from the PERT/CPM model;
- (3) update the SYMDB with metrics calculated from other metrics already stored in the database.

In the first step the PERT/CPM plan and the SYMDB are updated directly with the metrics collected in activity (M1a). The variables to be updated are identified in table 7.16 (see chapter 7). The update of the PERT/CPM model (i.e. categories (1) and (2) of table 7.16) was being carried out by the planning team as part of the established procedures. Regarding the update of the SYMDB, the metrics shown in table 8.7 above were updated in the database according to table 7.16 (category (3)). At the end of this step, if any structural changes occurred in the current PERT/CPM plan while being updated (e.g. adding or removal of tasks, resources and

dependencies from the network), then activity (M3a) must be called to update the analytical links. This was verified accordingly.

The second step was to update the SYDMDB with metrics derived from the PERT/CPM model. These are presented in table 7.17. This update involved some calculations which require the current and the initial PERT/CPM plans, as well as the structural links of work breakdown and organisational breakdown established between the two models. The calculation process and the specific requirements for each metric are presented in tables 7.18 through 7.22. These metrics were derived accordingly and were updated in the SYMDB. This shown in table 8.8 – note that this was a single-phase analysis using the SD prototype model, where the design phases was modelled and simulated in isolation from the rest of the project; therefore, no metrics were derived at the project level (i.e. category (1) in table 7.18). The SYMDB proposes a breakdown of the effort spent within the task, which depends on the type of SD-Task. In this example, the SD-Task was of engineering type and the breakdown considered was: development, QA and rework – the single-phase prototype model also considers a management component, but the effort spent with managerial control was considered as aggregated into these three engineering activities. The scope was measured in terms of % of functionality perceived to have been covered by the designs developed. The start date is here considered a day zero and the completion date corresponds to the duration of the task – in terms of actual dates, this phase started on late June 1995 and was planned to finish by the end of October 1995.

Days (cumulative)	0	21	42	63	84	89	Unit
Update of SYMDB <i>PERT/CPM derived metrics</i>							
(2) SD-Tasks							
(2.1) Schedule							
Start date[present] (*)	0	0	0	0	0	0	day
Finish date[present] (*)	84	84	84	84	89	89	day
(2.2) Effort							
ACWP[present]	0	50	120	174	215	227	person-day
ACWP Development	0	50	90	111	117	117	person-day
ACWP QA	0	0	15	33	45	45	person-day
ACWP Rework	0	0	15	30	53	65	person-day
BCWP[present]	0	73	133	194	230	242	person-day
BCWS[present]	0	61	121	182	242	242	person-day
CTC[present] (*)	242	192	105	36	12	0	person-day
CAC[present] (*)	242	242	225	210	227	227	person-day
(2.3) Scope							
SCAC[present] (*)	100%	100%	100%	100%	100%	100%	%
CSCC[present]	0	0	0	0	0	0	%
ASCWP[present]	0%	30%	55%	80%	95%	100%	%
SCTC[present] (*)	100%	70%	45%	20%	5%	0%	%
(2.4) Resources							
ASP[present]	2,88	2,88	2,88	2,88	2,88	2,88	person-month
PSP[present]	2,88	2,88	2,88	2,88	2,88	2,88	person-month
CASP[present]	0	2,88	5,76	8,64	11,52	12,21	person-month
CPSP[present]	0	2,88	5,76	8,64	11,52	12,21	person-month
CSPAC[present] (*)	11,52	11,52	11,52	11,52	12,21	12,21	person-month

Table 8.8 – SYMDB metrics derived from the PERT/CPM plan in activity (M2a)

The third step was to update the SYMDB with metrics calculated from other metrics already stored in the database. These metrics and the calculations required are shown in tables 7.23 through 7.24 (see appendix B). The resultant metrics for this example are shown in table 8.9 below.

Days (cumulative)	0	21	42	63	84	89	Unit
Update of SYMDB <i>Calculated metrics</i>							
(2) Performance indices							
(2.2) Task indices							
Earned value (EV)	0	50	137	206	230	242	person-day
CPI	1	1,45	1,11	1,11	1,07	1,07	–
SPI	1	1,20	1,10	1,07	0,95	1,00	–
Cost variance (CV)	0	-23	-13	-20	-15	-15	person-day
Schedule variance (SV)	0	-12	-12	-12	12	0	person-day
Accounting variance (AV)	0	-11	-1	-8	-27	-15	person-day
Time variance (TV)	0	-4,2	-4,13	-4,2	4,2	5	day
(3) Process metrics							
(3.1) Productivity							
(3.1.2) Task							
Gross	0	33	25	26	25	24	LOC/person-day
Net	0	33	34	40	45	47	LOC/person-day
(3.2) Defect							
(3.2.2) Task							
Detection index	0	0	12	25	39	37	defect/KLOC
Cost to detect	–	–	0,40	0,30	0,22	0,22	person-day/defect
Cost to rework	–	–	0,40	0,32	0,38	0,32	person-day/defect

Table 8.9 – SYMDB metrics derived from the PERT/CPM plan in activity (M2a)

Again, only metrics at the SD-Task level were calculated regarding the design phase being modelled. Note that for the productivity metrics, the scope was converted from %, as shown in table 8.8, to lines of code (LOC) based on the estimated size of the component (5 551 LOC). Also the defect index is measured in defect per KLOC (thousands of lines of code).

(M3a) update analytical links if necessary

As mentioned in chapter 7, this activity is entered when structural changes took place in the current PERT/CPM plan while being updated with actual results. Newly added or removed tasks, dependencies and resources may require some re-mapping to the SD model. The links that may need to be updated are the structural links. SYDPIM proposes two ways to identify the need for an update: the validity conditions imposed in the specification of these links are checked, and the structural consistency links are also checked. In this example, this activity was implemented accordingly.

(M4) extract project past behaviour from PERT/CPM model

The aim of this activity is to extract the project past behaviour from the PERT/CPM model – in the traditional framework, this past behaviour remains implicit in the PERT/CPM model. This process consist in deriving a set of patterns over-time from the past data stored in the PERT/CPM model. The specific patterns to be extracted, and the way in which they are calculated, is presented in tables 7.26 through 7.29. However, when the SYMDB is being implemented these patterns are already stored in this database (see description of activity (M4) in chapter 7). In any case, SYDPIM advocates that the object “PERT/CPM past behaviour” is always updated or generated for later use in activity (M6).

In this example, the SYMDB was updated in the previous activity. Either by implementing the calculations described in tables 7.26 through 7.29 or by simply using the SYMDB as the source of data, the resultant past behaviour patterns produced are the same. As an example, the schedule and effort patterns (tables 7.26 and 7.27), for the SD-Task that corresponds to the design phase of the system component being diagnosed, are shown in figure 8.7 – a spreadsheet tool was used

to plot the patterns. Because in this example the SYMDB is being implemented, these patterns correspond to the over-time data stored in this database, as shown in table 8.8 (data categories 2.1 and 2.2).

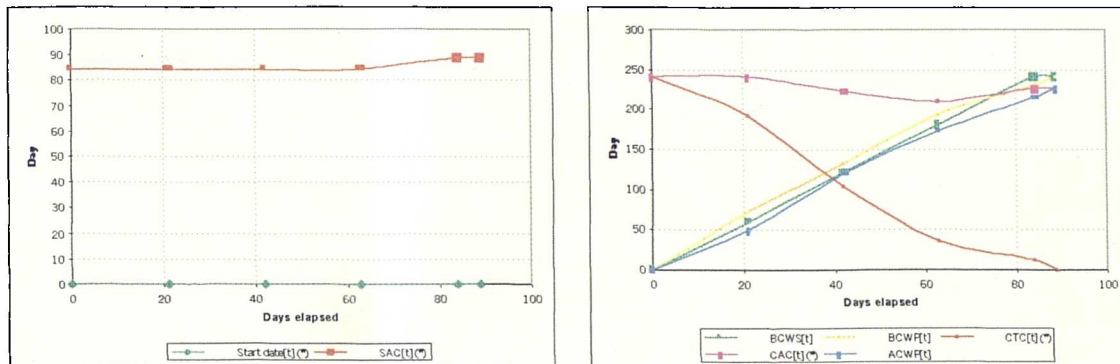


Figure 8.7 – Schedule and effort behaviour patterns extracted from the PERT/CPM model in SYDPIM activity (M4), according to the specification in tables 7.26 and 7.27.

The same patterns can be produced based on the SYDPIM calculation processes proposed in tables 7.26 and 7.27. For example, the planned completion date for the SD-Task is represented by the pattern SAC[t]. According to table 7.26 this is calculated as “...for each time-point, MAX planned finishing date of all tasks in the PERT/CPM plan of that time-point mapped to the SD-Task.” According to this, for each time-point the version of the PERT/CPM plan of that moment in time is used. Therefore, all past versions of the PERT/CPM plan stored within the PERT/CPM model are required. The structural link of structural correspondence (SC-WBS), which maps the PERT/CPM task to the SD-Task, is also needed. Since this link may also evolve over-time, all past versions for each moment in time are also required. Since the control period being used was the month (i.e. 21 days) and the SD-Task lasted for about 89 days, then six time-points were calculated as shown in table 8.10.

Time-point	Calculation	Value
Day 0	Latest planned completion date of task of the PERT/CPM initial plan mapped to the SD-Task	84
Day 21, Month 1	Latest planned completion date of task of the PERT/CPM plan of day 21, mapped to the SD-Task	84
Day 42, Month 2	Latest planned completion date of task of the PERT/CPM plan of day 42, mapped to the SD-Task	84
Day 63, Month 3	Latest planned completion date of task of the PERT/CPM plan of day 63, mapped to the SD-Task	84
Day 84, Month 4	Latest planned completion date of task of the PERT/CPM plan of day 84, mapped to the SD-Task	89
Day 89	Latest planned completion date of task of the PERT/CPM plan of day 89, mapped to the SD-Task	89

Table 8.10 – Calculation of the time-points for the behaviour pattern SAC[t] extracted from the PERT/CPM model

These calculations shows that in terms of final schedule, no re-planning actions were undertaken until day 84. This indicates that a schedule slippage was accepted only when the initially planned completion date was being reached. As the design work was not completed by day 84, as planned, an extra week of schedule extension was given to the design team and the PERT/CPM plan was updated accordingly. This new schedule was achieved successfully.

Overall, SYDPIM proposes that 16 basic patterns are extracted from the PERT/CPM model, as shown in tables 7.26 through 7.29 (2 for schedule, 5 for effort, 4 for scope, and 5 for resources) – if a breakdown of the ACWP is considered then a few more patterns are added to this. Figure 8.7 above shows 7 of these patterns. In this example, the patterns that refer to scope and resources are not particularly illustrative because scope changes did no occur and the number of resources working in the task was planned at a constant level and it also did not changed. The calculations proposed in tables 7-26 through 7.29 are straightforward to implement, as in the example shown in table 8.10 above. The implementation of these calculation processes can also be supported by the specific PERT/CPM tool in use (e.g. if it produces aggregate data for clusters of PERT/CPM tasks). Once these patterns were extracted they were stored in the “PERT/CPM past behaviour” object, which was implemented in a spreadsheet.

(M5) specify project past behaviour

The aim of this activity is to produce the final set of patterns which will describe the project past behaviour up to present. This past behaviour may include the patterns extracted from the PERT/CPM model in the previous activity, the patterns updated in the SYMDB in activity (M2a) or (M2b), and patterns extracted from expert judgement. This depends on whether the PERT/CPM model is available and updated and whether the SYMDB is in use. As described in chapter 7, three scenarios are considered in SYDPIM:

- (1) the SYMDB is available and updated with past results;
- (2) the SYMDB is not available and a PERT/CPM model is available and updated;
- (3) both the SYMDB and PERT/CPM model are not available and only expert judgement is available from activity (M1c).

In this example, the first scenario is considered and so SYDPIM proposes that the following action is carried out in this activity:

- extract from expert judgement the following type of patterns, as perceived relevant: (i) risk-related, and (ii) intangible issues (e.g. staff fatigue) – that the SD project model must contain variables within its structure which produce these patterns;
- Merge these patterns with the ones stored in the SYMDB and generate the “Project past behaviour” object.

The SYMDB contains all the patterns that can be quantified from the data collected so far and from the PERT/CPM model. Therefore, in this SYDPIM activity other patterns which are perceived as relevant are derived from expert judgement, and are merged to the ones stored in the SYMDB – for example, if the Client asked for changes then, the change requests over-time would be specified. In this example, SYDPIM was being attempted for the first time and there was no extra patterns that management perceived as relevant. Therefore, the final set of patterns used to describe the past behaviour of the specific design phase are the ones stored in the SYMDB. The formal specification of the SYMDB presented in chapter 7 (see also appendix C) considers four categories of metrics:

- (1) derived from PERT/CPM model;

- (2) collected from the project;
- (3) calculated from other metrics;
- (4) uncovered from SD model.

Ideally, all the patterns considered within each of these categories should be included in the specification of the project behaviour. In the next activity (M6) the SD model will have to reproduce all these patterns. The patterns that correspond to the metrics in category (1) were already extracted and specified from the PERT/CPM model in activity (M4) (as exemplified figure 8.7). The patterns in category (2) refer to metrics defects which were collected in activity (M1a). The patterns in category (3) are to an extent redundant because they are generated from other more elementary patterns. Therefore, if the SD model reproduces well the elementary patterns, then it will also reproduce well these patterns. However, depending on the mathematical transformation the accuracy of the behaviour reproduction can change considerably. For example, if the SD model reproduces well the patterns $ACWP[t]$ and $BCWP[t]$, then it will also reproduce well the pattern $CPI[t] := BCWP[t]/ACWP[t]$. However, when $ACWP[t]$ is small and is reproduced with a minor error, the resultant error in $CPI[t]$ can be of considerable magnitude. This should be taken into account when evaluating the “goodness-of-fit” in next activity (M6). It is not mandatory but it is recommended that these patterns are specified in the project past behaviour. Finally, the patterns in category (4) refer to metrics uncovered from the SD model. These patterns will only be available until the past moment in time where the SD model was calibrated to reproduce the project past. Therefore they are not available for the control period just elapsed. Assuming that the last calibration of the SD model is valid, it is desirable that the new (re)calibration of the SD model to be carried out in activity (M6), will reproduce the same patterns. However, it is also reasonable to assume that as more past is available from the project improvements to the past calibration can be made. This way, the past patterns derived from the previous calibration are included in the “project past behaviour” for verification in the next activity (M6). For all of the four categories, the SYMDB considers metrics at both SD-Task and whole project level. In this example, only patterns at the SD-Task level were considered, because the design phase of a system component was being modelled and diagnosed in isolation using the single-phase prototype model.

In the current example, there was no past calibration available (the SD model was being calibrated for the first time to reproduce the design phase of this component). Therefore, the patterns considered for the “project past behaviour” were the ones extracted from the PERT/CPM model (activity M4), plus the defect patterns based on metrics collected from the project and the patterns based on the calculated metrics. Figure 8.8 below shows examples of defect patterns (see table 8.7), and figure 8.9 shows the calculated patterns of the performance indices SPI and CPI, plotted over-time and as an XY graph (see table 8.9).

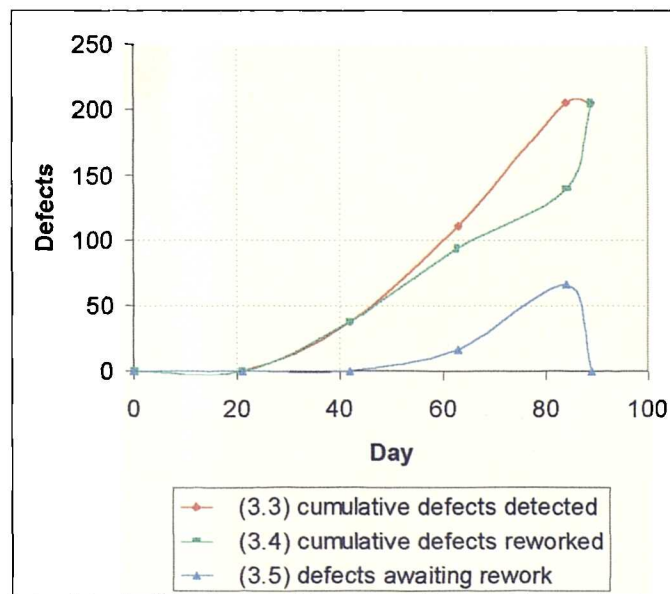


Figure 8.8 – Defect patterns incorporated in the project past behaviour, specified from metrics collected in the SYMDB (see table 8.7)

The defect patterns show a gap between the cumulative defects detected and reworked. The difference corresponds to the level of detected defects awaiting to be reworked. It was not until the end of the first month that the design reviews started detecting defects. In the beginning the completion dates was still distant and hence rework was carried out immediately. However, as the deadline approached greater priority was probably given to development and reworked as carried out at a slower pace. This probably explain the increase in the amount of defects awaiting rework. The last few days were probably spent only in reworking the resultant backlog of detected defects. The specification of the project past behaviour in this way immediately suggests hypothetical dynamic explanations for the observed outcome, prior to the use of the SD model. This type of exercise is not encouraged by the traditional project management framework.

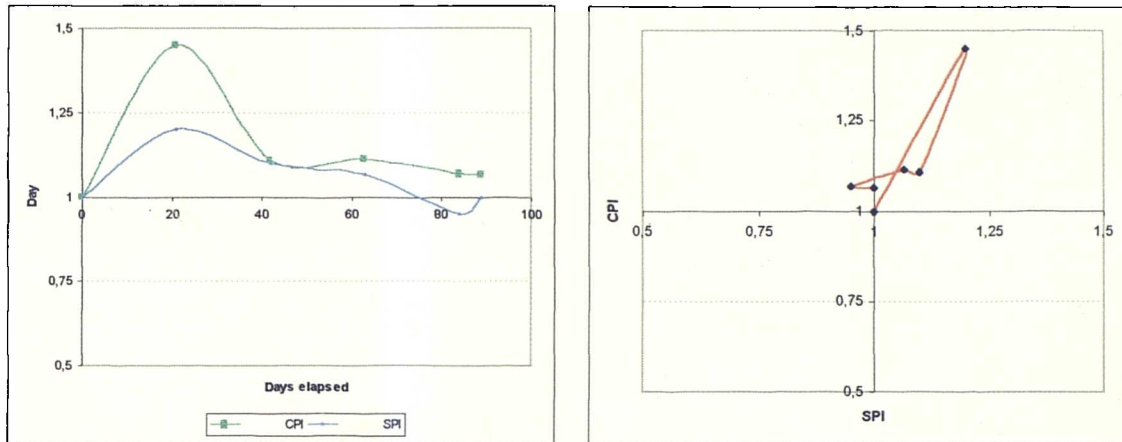


Figure 8.9 – Performance indices patterns incorporated in the project past behaviour, specified from metrics calculated within the SYMDB (see table 8.8)

The performance indices above (SPI for schedule and CPI for cost), suggest that in the early stages of design work ahead (i.e. BCWP greater than BCWS, hence $SPI > 1$) and costing less than planned (BCWP greater than ACWP, hence $CPI > 1$) – for most of the design phase, the indices stood in the top-right corner of the XY graph, hence indicating a good performance of both cost and schedule. However, as the work progressed into the stages of more detailed design, this optimistic indications started to fade away, in particular in terms of schedule. When the planned completion date was reached, the SPI index was less than 1, indicating that a schedule delay occurred (indices in the top-left region of the XY plot). When the work was finally completed, the CPI index was greater than 1 indicating that the design phase cost less than planned.

At the end of this SYDPIM activity, the past behaviour of the sign phase of the component being diagnosed was specified in this way. The final set of patterns was stored in the “project past behaviour” object, which was implemented in a spreadsheet. From the shape of these patterns, some interesting dynamic analysis and hypothesis were derived. The next step was to calibrate the initial prototype SD model to reproduce this behaviour.

(M6) re-calibrate SD model for past behaviour

The purpose of this activity is to calibrate the SD model to reproduce the project past behaviour just specified. When an updated PERT/CPM plan is available and

the SYMDB is being implemented, SYDPIM proposes a generic algorithm which was described in detail in chapter 7. This algorithm makes use of the analytical links in order to support the calibration process. These are used in three distinctive ways (see description of this activity in chapter 7):

- (1) to evaluate “goodness-of-fit” regarding PERT/CPM derived patterns – DCO links are used for this purpose, plus a DCOI link;
- (2) to calibrate the SD model through exogenous management decisions – DEI and DEOI links are used for this purpose;
- (3) to check whether the new calibration has changed the initial plan in the SD model – DCI and DCOI links are used for this purpose.

The first step in the algorithm proposed is to simulate the SD model with the previous calibration. This will be a calibration which will reproduce the previous project plan (i.e. a steady behaviour). This is because whenever monitoring is entered, the SD model retains the last calibration carried out in SYDPIM planning, which ensures this plan will be reproduced. In this example, the last time that the SD model had been calibrated was in the beginning of the design phase, and so the SD model was reproducing the initial plan. The second step in the algorithm is to compare the behaviour produced by the SD model against the project past behaviour specified in the previous activity. This is done mainly through the implementation of output-output data consistency links (DCO). These links compare the various PERT/CPM-derived behaviour patterns and produce statistics of “goodness-of-fit”. For the sake of clarity and purpose of this section, a detailed presentation of all these indices is not here presented. Figure 8.10 compares the schedule and effort patterns of the project past behaviour with the ones simulated by the SD model. The SYDPIM analytical link DCO-1.2 checks the consistency of the past schedule patterns while the analytical link DCO-2-2 checks the consistency of the past effort patterns. The behaviour produced by the SD model is steady and shows the design phase evolving and being complete on schedule and budget. On the other hand, the project past behaviour shows the schedule slipping by and the design phase being complete under-budget. Clearly, the model is not reproducing well what happened in this design phase and hence needs to be calibrated.

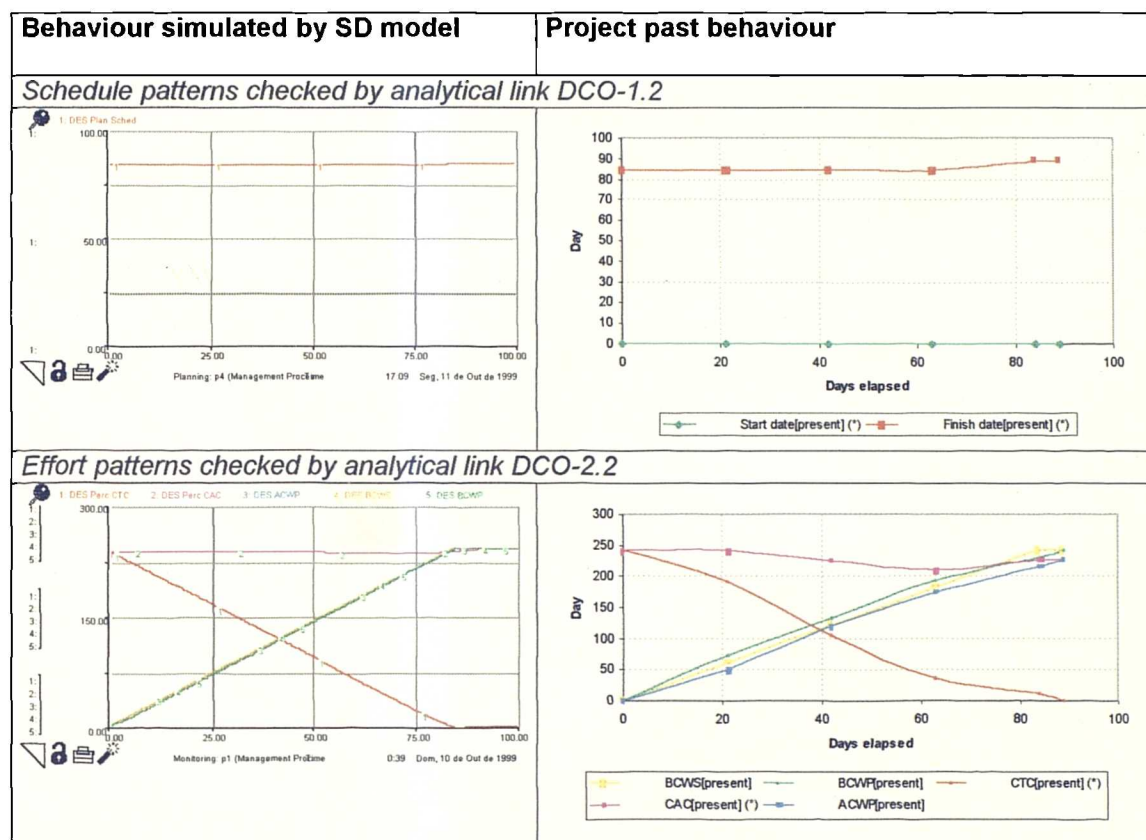


Figure 8.10 – Schedule and effort patterns: project past behaviour versus behaviour produced by the SD model

The next step in the algorithm is to diagnose the differences between the two sets of patterns. In this case, there are differences in the schedule and in the effort patterns. There were no relevant differences regarding the resources patterns (this was constant throughout the design phase) nor regarding scope. However, there were also differences regarding the defect patterns: the planned behaviour produced by the SD model indicated a higher number of defect detection and lower defect generation. The diagnosis of the differences between the patterns should focus on both shape and final values (e.g. how the schedule changed and the final completion date). Table 8.11 below shows some comparative final data.

Metrics / Patterns	SD simulation (initial plan)	Project past behaviour
ACWP (person-day)	242	227
ACWP-Development (person-day)	150 (62%)	117 (52%)
ACWP-Rework (person-day)	46 (19%)	65 (29%)
ACWP-QA (person-day)	46 (19%)	45 (19%)
Completion date (SAC) (day)	84	89
Defects detected (defect)	220	205

Table 8.11 – Comparing the final values of behaviour patterns between the SD simulation of the initial plan and the project past behaviour

A detailed description of the calibration process is not presented here. However, table 8.12 below shows the final parameter changes and the correspondent underlying rationale – the identification of a plausible rationale is essential to support the validation of the calibration process.

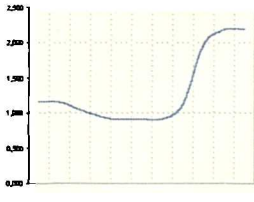
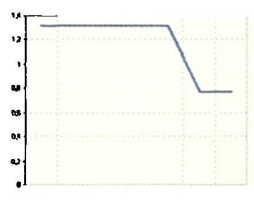
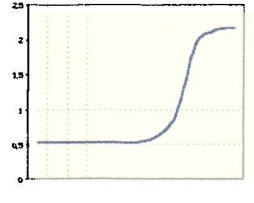
SD parameter changed	Initial	Current	Rationale
Nominal productivity (LOC/person-day)	37	47	Metrics collected (see table 8.9). Due to more rework, development productivity was higher.
Nominal cost to detect a defect (person-day/defect)	0.1254	0.1254	No
Nominal cost to rework a defect (person-day/defect)	0.21	0.315	Staff reported that overall defects were harder to fix than expected. Metrics collected supported this (see tables 8.8 and 8.9).
Productivity learning curve (multiplier)	Constant (=1)		Initially, work complexity (detailed design) decreased productivity but learning increased in the later stages. Metrics collected support this (see tables 8.8 and 8.9).
Cost to detect learning curve (multiplier)	Constant (=1)		Defects were harder to detect in the high-level architectural design, as easier as design is more detailed and formal. Metrics collected support this (see tables 8.8 and 8.9).
Cost to rework learning curve (multiplier)	Constant (=1)		Staff reported that the last defects of detailed formal design were harder to fix and ultimately responsible for the schedule slippage. Metrics collected support this (see tables 8.8 and 8.9).

Table 8.12 – Parameter changes and rationale to calibrate the SD model for the actual behaviour

The re-calibration of the SD model was primarily based on changes to three of the four process metrics which form the “horse-power” of a development team. Note

that the three instantaneous curves adjusted, which are a function of progress, can be derived from the metrics collected presented in tables 8.8 and 8.9 (i.e. similar shapes are achieved). According to the algorithm proposed in chapter 7 for this activity, the values of the process metrics should be used to support this re-calibration and this was done in this example. However, exogenous decision were not imposed via the analytical links. This is because no relevant management action was implemented which affected the outcome of the design phase. The only control action was the schedule readjustment and this was generated successfully within the model. The next step in the algorithm is to check whether “invalid” changes were made to the initial plan – as shown by table 8.12, this was not the case. Once this re-calibration is satisfactory, the SD model is simulated again and the process repeats. This time, an acceptable fit was achieved. Figure 8.11 shows a comparison between the behaviour patterns achieved regarding schedule and cost.

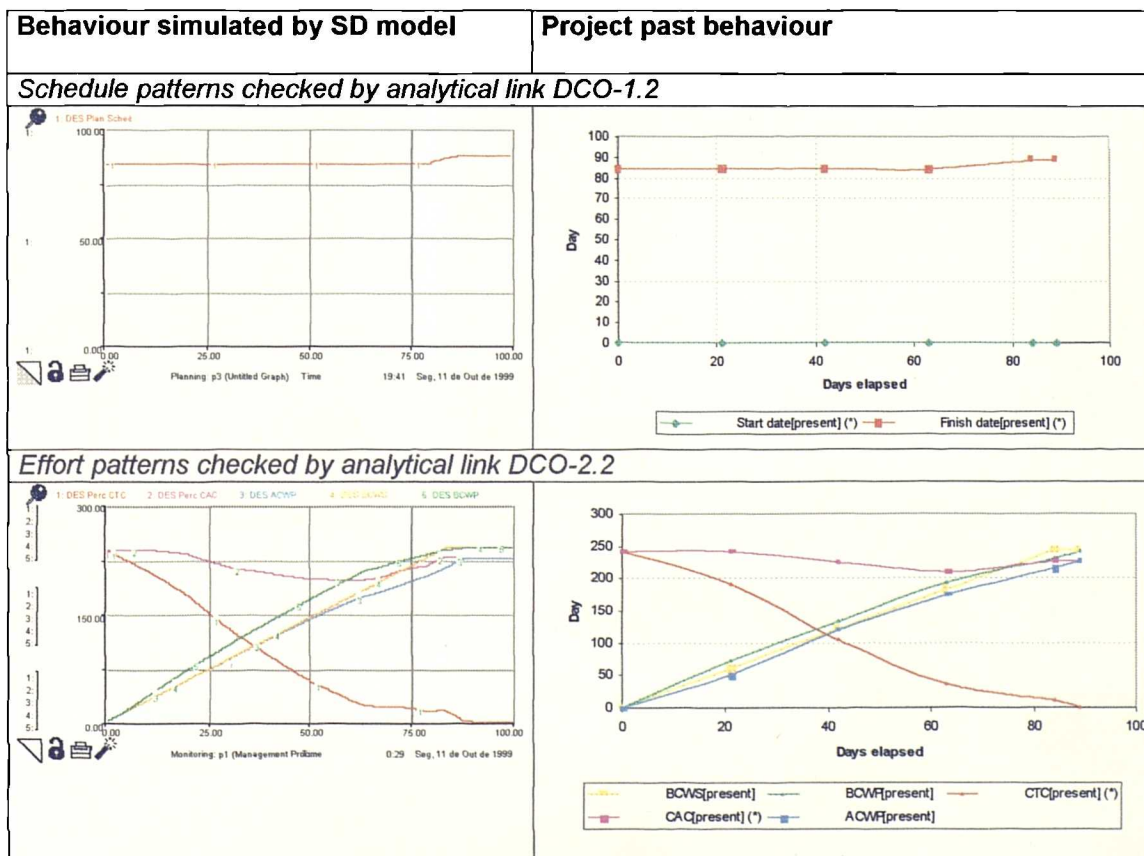


Figure 8.11 – Schedule and effort patterns: project past behaviour versus behaviour produced by the SD model

Not only the PERT/CPM derived patterns were reasonable reproduced by the SD model. For example, figure 8.12 below shows some of the defect patterns – since these are non-PERT/CPM patterns, the analytical links cannot be used to check the “goodness-of-fit”. Overall, the patterns produced by the SD model are smoother and less discrete than the ones in the project past behaviour. This is due to the continuous nature of the SD simulation. For example, in the SD model defects start being detected early in the simulation whereas in the actual project past behaviour, no defects were detected in the first month. In general, in “instantaneous” type of patterns or in accumulations which are depleted, this difference between discrete events in the real world and the continuous nature of the SD model are more evident. However, overall an acceptable fit was achieved (statistics of “goodness-of-fit” were produced for evaluation, as proposed by SYDPIM). This is reinforced by the final values of the key behaviour patterns, as shown in table 8.13 below.

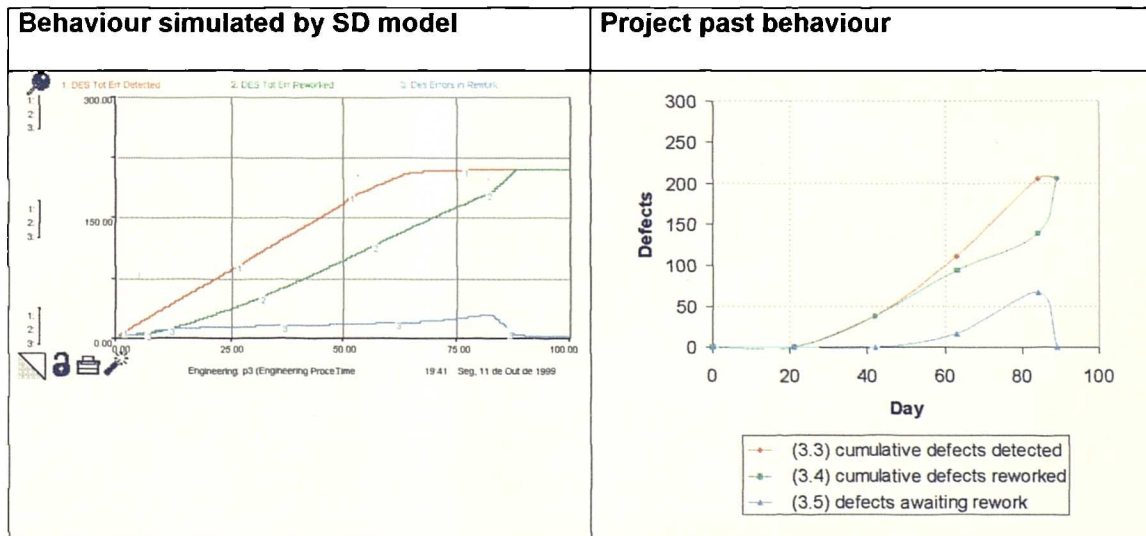


Figure 8.12 – Defect patterns: project past behaviour versus behaviour produced by the SD model

The final results are extremely close to the ones actually occurred. The few variations are minor and could be eliminated with further refinements of the calibration. However, the extra effort was not worth and it is also important not to sacrifice the validity of the calibration for the sake of further accuracy in the final results.

Metrics / Patterns	SD simulation	Project past behaviour
ACWP (person-day)	227.15	227
ACWP-Development (person-day)	117.12	117
ACWP-Rework (person-day)	64.50	65
ACWP-QA (person-day)	45.43	45
Completion date (SAC) (day)	88.00	89
Defects detected (defect)	207.23	205

Table 8.13 – Comparing the final values of behaviour patterns between the SD simulation after re-calibration and the project past behaviour

Figure 8.13 below compares the XY plot of the performance indices SPI and CPI. Since these result from mathematical divisions of other patterns, the variations tend to propagate. Nevertheless, the overall shape of the curve is very similar, suggesting that a satisfactory calibration was achieved.

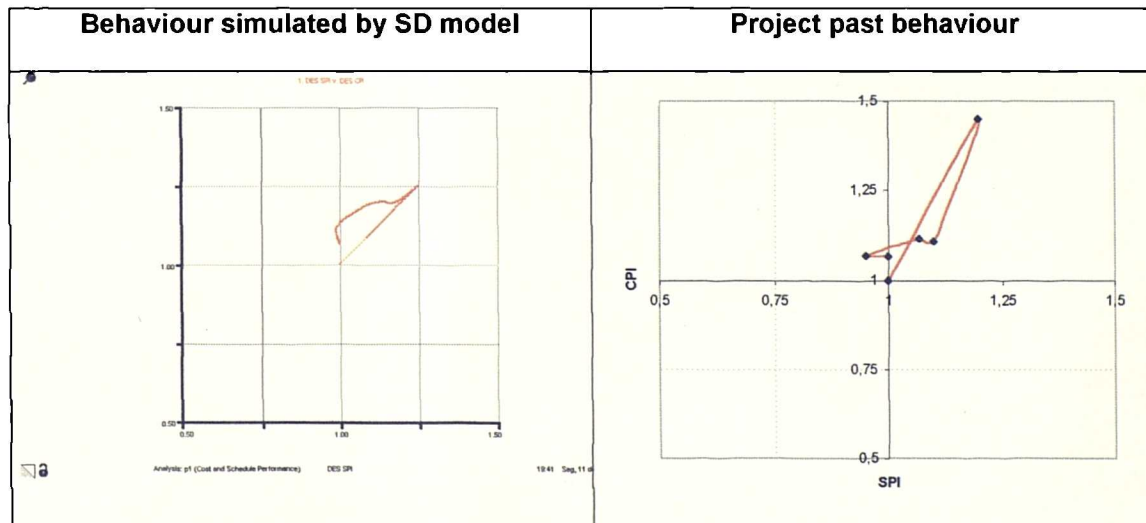


Figure 8.13 – Comparing the performance indices SPI and CPI between the SD simulation after re-calibration, and the project past behaviour

Figure 8.14 shows a snapshot in iThink of the SD model reproducing the actual behaviour of the design phase. The graph at the centre-bottom shows the “progress ramp” (Cooper 1993), which plots the perceived progress (X-axis) against the real progress (Y-axis). The vertical shape near the end shows the occurrence of the classic “90% syndrome” where late rework causes the perceived progress to get stuck at the 90% level, while real progress is achieved as rework is accomplished. As it will be discussed in the SYDPIM activity, this phenomenon took place in this design (although at a small scale).

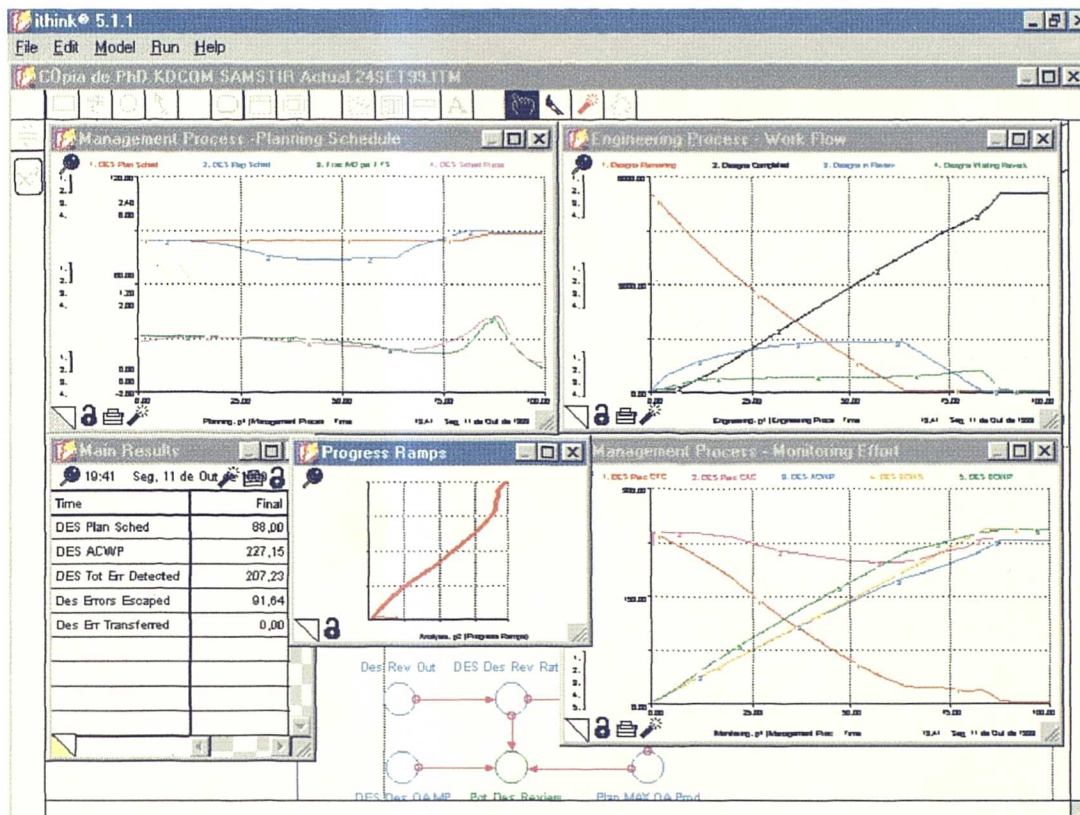


Figure 8.14 – A snapshot of the SD prototype model in iThink calibrated to reproduce the past behaviour of the design phase

Once the model was reproducing well project the past behaviour of the design phase, the next step was to diagnose what had happened using the SD model.

(M7) SD diagnosis of project past

One of the most powerful features of a SD project model is its ability to explain why a project has evolved, or will evolve, in a certain way. This is because a SD model embodies explicitly an explanatory theory about the system being modelled (Barlas 1996).

The purpose of this SYDPIM activity is to focus on the relevant aspects of the project past behaviour (e.g. when did the schedule start slipping by), and use the model to uncover the underlying causes (e.g. it was because of late hard defects). As described in chapter 7, the SYDPM project management method proposes that the diagnosis of the project past is based on four possible distinctive ways of using the SD model:

- *extract performance and process-related metrics*, and compare them against the assumptions in the plan (e.g. were defects harder to fix than assumed in the plan?);
- *uncover intangible metrics about the present project status* (e.g. there are currently about 50 defects/KLOC uncovered in the code);
- *retrospective “what-if” analysis of other scenarios* (e.g. could have the delay been prevented if we had started with more resources);
- *causal analysis* (e.g. at a certain stage the vicious circle between schedule pressure and delays due to defect discovery became dominant, and schedule slippage could not be prevented).

These types of diagnosis can be carried out independently from one another. However, it is suggested by SYDPIM that a more insightful understanding can be achieved if they are combined in a specific sequence of steps, which was proposed in chapter 7 as follows:

- (1) *analysis of process metrics* – compare and quantify the deviations between the actual values of those variables and input parameters relate to process metrics, against those values assumed in the plan. This is aimed at identifying fundamental variations from the assumptions in the plan (e.g. productivity was lower than expected, or the learning curve was slower than expected);
- (2) *causal analysis* – based on the shapes of the behaviour patterns, identify the most relevant dynamic characteristics of the project behaviour, and try to relate these to feedback loop dominance. This is aimed at identifying the feedback processes responsible for the disruptive changes in the project outcome (e.g. because productivity was lower than expected, slow progress led to schedule pressure which disrupted quality, caused late rework and even more pressure; this vicious circle led to continuous slippage);
- (3) *what-if analysis to validate conclusions* – the analysis in the two previous steps probably led to hypotheses to explain the causes for the observed past behaviour. To validate these explanatory hypothesis, retrospective what-if scenarios are tested (e.g. what-if productivity had been as planned; what-if the schedule had been extended by more days as soon as first signs of delays emerged: would have the vicious circle of schedule pressure been prevented?);
- (4) *what-if analysis for process improvement* – another major advantage of running simulations in the past is to verify whether alternative planning and control

policies and decisions would have led to better results (e.g. would have a greater QA budget led to better results?).

This set of SD analyses and the proposed sequence of steps can be taken as a base framework to conduct the diagnosis.

In this current example, the first step was to compare the values of the main variables related to process metrics. This comparison is shown in table 8.14 below:

Variable	Planned	Actual	Deviation
<i>Productivity</i>			
Gross productivity (task/person-day)	22,92	24,43	7%
Net productivity (task/person-day)	37,00	47,39	28%
<i>Effort expenditure</i>			
Development effort (person-day)	150,00	117,12	-22%
QA effort (person-day)	46,00	45,43	-1%
Rework effort (person-day)	46,00	64,50	40%
<i>Defect detection and rework</i>			
Cost to rework (person-day/defect)	0,21	0,31	48%
Cost to detect (person-day/defect)	0,21	0,21	0%
<i>Defect indices</i>			
Defects generated (defect/KLOC)	45,36	51,02	12%
Defects detected (defect/KLOC)	39,71	37,33	-6%
Defects escaped /defect/KLOC)	8,61	16,51	92%

Table 8.14 – Comparison of variables related to process metrics for diagnosis of project past (stage 1)

The particular set of variables to be compared depend on the type of industry, management perspectives and ultimately on the SD model (variables need to become available). The variables selected for this example reflect the software nature of the project. They also reflect the SYDPIM perspective of development performance, which is based on four main dimensions of overall productivity: development productivity (how fast is work developed first time), error generation (the quality of the development work), error detection (how effectively are defects detected), and error rework (how effectively are defects fixed). The deviations presented in the last column of table 8.14r provide some initial insights about the outcome of the design phase of this component. This phase was completed one week later but with effort savings (the model assumed that in part effort saved would spent by the team members in other tasks). The gross productivity refers to the overall productivity achieved. This metric indicates a 7% increase which explains the overall effort savings. However, the development productivity metric

increased much more (28%), suggesting that much more savings could have been achieved (the designs were developed first time much faster than expected). Why were the savings only of 7%, and why was there a delay? The effort metrics indicate that much more effort was spent on rework than expected (a 40 % increase). There are two main possible explanations for this: either more defects were detected than expected, or they were harder to fix. The metric “cost to fix” indicates that the cost to fix was in fact 48% higher than assumed in the plan. However, the number of defects detected decreased just slightly (6%). The defect detection metrics (QA expenditure and cost to detect) showed no variation. Overall, this analysis suggests that the designs were easier to develop first time (perhaps because the team was familiarised with the system functionality, as they had worked on similar developments before), but the defects found were harder to fix than expected. Since most of the defects are detected in the final reviews of the detailed designs, this was perhaps the reason for the schedule slippage. The calibrated learning curve “cost to rework”, shown in table 8.12, supports this argument. On the other hand, the effort savings resulted from the early development of the designs at a higher productivity than planned. An analysis of a simple set of relevant metrics in this way starts revealing the underlying causes of the outcome. However, the SD mode provides more metrics than the ones that can be conventionally collected, and which can lead to further valuable insights. For example, the conventional metrics in table 8.14 indicate effort savings and a schedule slippage but nothing relevant appeared to happen with the quality of the designs: the planned QA effort was spent and the expected number of defects was detected. The metrics “defect generated” and “defects escaped” cannot be collected by conventional procedures at this stage in the life-cycle. The SD model provides interesting estimates: more defects were generated (+5 per KLOC) than as if the design phase had been implemented “normally” (i.e. steady behaviour) and much more defects escaped to coding (about twice). A higher defect generation was probably due to schedule pressure in the later stages causing a considerable increase in the number of “bad fixes” (i.e. defects improperly reworked). Since the increase of “bad fixes” is not perceived by staff, the QA level was kept as planned. The higher number of defects generated and the increase in “bad fixes” led to the higher number of defect escaping. The conclusion is that while a week delay is not too serious and is probably outweighed by the savings in effort, the quality impacts are serious. By estimating these

intangible defect metrics, the SD model adds the quality dimension to the conventional cost-time trade-off.

It is important to relate this type of analysis to the calibration of the SD model. First, some of the causes for the observed outcome start to show up during the calibration process. For example, the input parameter “nominal productivity” was increased from 37 task/person-day to 47 task/person-day (see table 8.12), the same variation observed in the “gross productivity” output metric from the SD model. Furthermore, the shape of the input learning curves in table 8.12 also suggest underlying causes for the outcome. Therefore, the diagnosis of the process-metrics related variables should be made at the light of the calibration. However, it is equally important to note that while the output variables analysed in this diagnosis step are directly influenced by the calibration changes, they are also affected *indirectly* by the feedback structure. Often these indirect effects are of greater magnitude than the direct effects. For example, there was no change to the input parameter “nominal error generation index”, but the resultant defect generation and defect escape increased considerably. The variables analysed in this diagnosis step aggregate the overall direct and indirect feedback effects. While the calibration changes are important and should be considered in this analysis, the output variables hold a richer diagnosis information. Finally, there may be important issues that should be considered in this analysis depending on the model features. For example, the effort and productivity metrics shown above do not account for non-reported extra-hours worked by the staff when under pressure. The SD model used provides both the “reported” and the “real” productivity and effort (it also estimated the number of non-reported extra hours worked). Under conditions of schedule pressure, the real productivity is lower than the reported productivity.

The second step proposed in this diagnosis is causal analysis. This consists in identifying the specific feedback loops responsible for the key aspects of the project behaviour. It is suggested that the generic feedback structure proposed in the SYDPIM development method and the list of feedback loops specified in the design phase of the model development process (see figure 6.4 and stage 0 of model design chapter 6) are used for this purpose – an extra value of the SYDPIM Model Development Method. This prevents the modeller from having to identify the key feedback loops within the model structure. For the sake of scope, this step is not

illustrated here in detail. Typically, S-shaped variations indicate an alternate dominance between positive and negative feedback loops. Some generic project feedback loops, which are easily classified according to the framework in figure 6.4, will be responsible for these shapes – e.g. reinforcing loop between productivity learning and work progress, versus the control loop which allocates less effort to development as rework emerges and there is less development to do. An effective way of identifying dominant loops is to verify the actual impacts occurred on the process metrics. For example, if error generation was seriously hurt by a certain effect, then it is likely that a reinforcing loop took place through this cause-effect relationship. Figure 8.15 below shows the effect on error generation of schedule pressure. Only in the later stages, pressure increased and probably led to the generation of more errors.

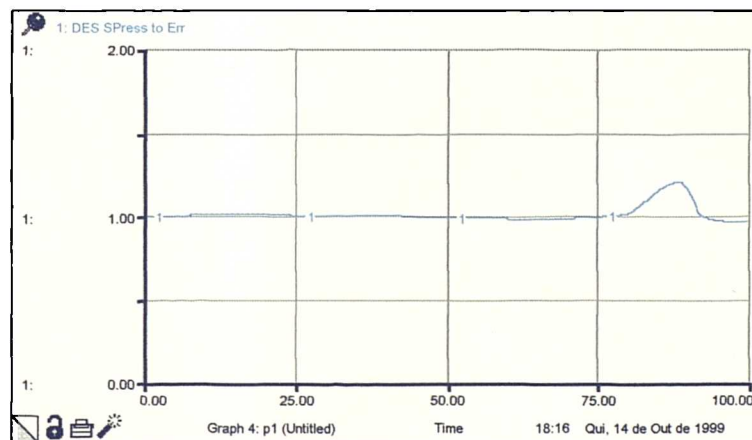


Figure 8.15 – The effect of schedule pressure on error generation

The following steps (3) and (4) of the diagnosis process refer to retrospective “what-if” analysis. Step (3) is aimed at confirming possible explanations while step(4) focuses on process improvement. In both cases, an exhaustive analysis can be carried out with numerous “what-if” scenarios being formulated and tested. As an example, two simple examples are here presented. Regarding the testing of possible explanations, the analysis of step (1) suggested that late expensive rework was responsible for the schedule slippage. One way to test whether this is a plausible explanation, is to eliminate the effects of this learning curve, replacing it by a flat straight line, while keeping all the other calibration. The outcome of this retrospective scenario is shown in figure 8.16, compared with the base calibration (i.e. actual past).

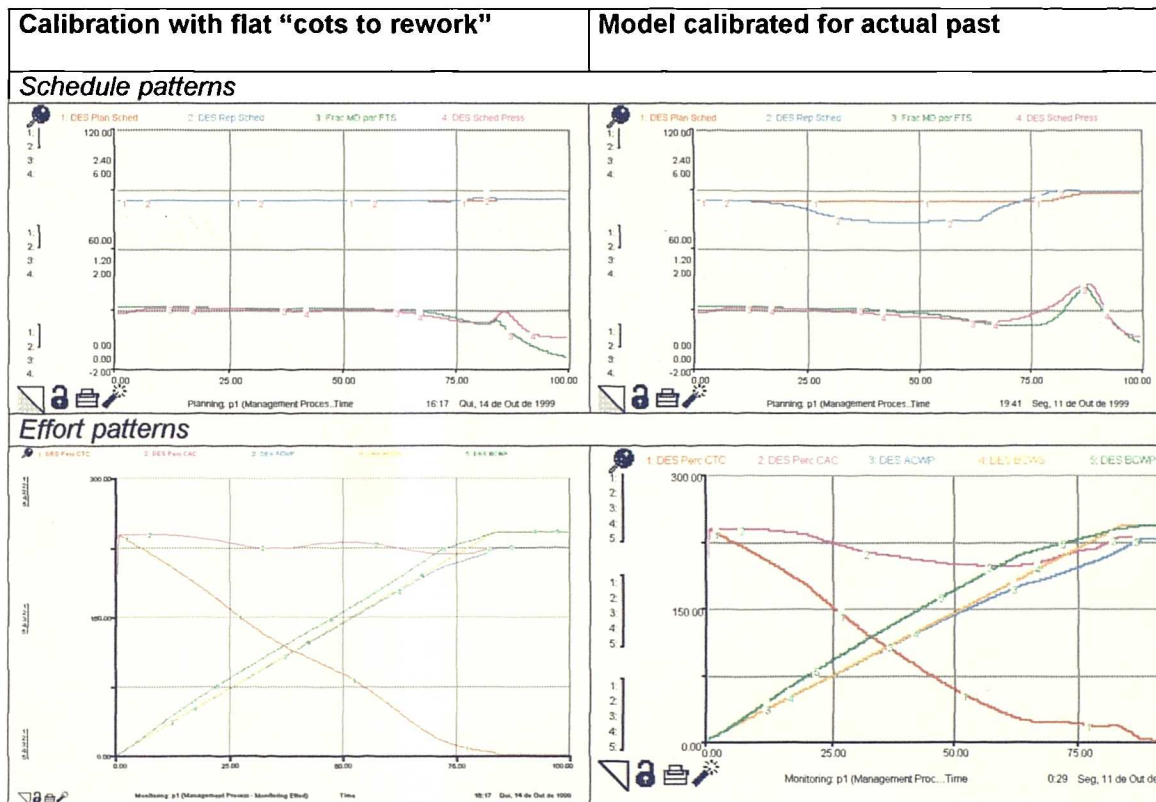


Figure 8.16 – Schedule and effort patterns: actual past versus retrospective scenario with a flat “cost to rework” learning curve

The schedule patterns include four variables: the planned completion date at the management level (red), the estimated completion date reported by designers (blue), designers schedule pressure (pink), and the level of “slack usage” (green) (this refers to non-reported extra hours; the normal level is 0.6, as reported in Abdel-Hamid and Madnick 1991). This analysis suggests that with a flat learning curve for “cost to rework”, the same effort savings are achieved but the schedule slippage does not occur, because there are no “rework surprises” at the end. Also interesting is that the flat learning curve also eliminates the designers early perception that the work may be completed earlier than expected (as per the blue curve in actual past). This is because the flat learning curve also increases the cost to rework in the early stages.

As the conclusions from this retrospective analysis support (or not) the hypothesis, further scenarios can be tested to gain a more in depth understanding of the real causes. In practice, many scenarios can be analysed. Only a few of these can be recorded for future analyses and to support management reports. In this exercise it

is important to focus on the appropriate model variables, as the analyst can easily become overwhelmed by the numerous patterns and metrics that a SD model can produce.

Regarding process improvement, this analysis focus on changing those factors under managerial control. These factors can refer to both the project management process and the product development process. In the management process two types of factors can be changed: initial planning decisions (e.g. planned completion date, planned profile of resources), and control policies (e.g. schedule adjustment in face of perceived delays, resource adjustment).

In the previous steps, the diagnosis of the design phase suggests that an excessive budget was planned for development work, because the net development productivity was considerably higher than expected. Therefore, if more budget had been initially planned to QA and rework activities, probably better results could have been achieved. If so, the following question is by how much. An excessive increase will certainly have undesired impacts. Table 8.15 compares the base case (20% for QA and 20% for rework), with other two scenarios. In the first there is a 5% and 10% increase for QA and rework respectively. In the second scenario there is 20% and 25%. The results suggest a sharp decrease in the number of defects escaping, hence a considerably quality gain. However, while in the first scenario the work is finished earlier than in the base case, in the second scenario the schedule slippage worsens considerably from the base case, but the quality gains against the first scenario are minor. In both scenarios, the effort spent is the same ,and less than in the base case (this is because less defects are generated and reworked). This analysis suggests that moderate increase in the QA and rework planned budgets would have been beneficial. When taking into account these conclusions for future planning it is important to be aware that these refer to he specific conditions in which this design phase was implemented.

Scenario	QA: 20% Rework: 20% (base)	QA: 25% Rework: 30%	QA: 40% Rework: 45%
Schedule (day)	89	87	94
ACWP (person-day)	227	223	223
Defects (defect)	205	47	44

Table 8.15 – Results from retrospective “what-if” scenarios for process improvement: increasing the QA and rework levels.

Many other type of scenarios can be tested to support process improvement type of analysis. Typically, the factors selected to be changed are varied within a certain range, thereby generating a series of scenarios. The main final results from these scenarios are plotted in a graph for visual analysis. This type of analysis is prone to generate feelings of “guiltiness” about the past or alternatively negative reactions against the model (e.g. “We have done the best of the possible so far”). In order to prevent this, in some occasions it is appropriate to restrict the access to the conclusions only to the appropriate persons. The generalisation and extrapolation of the conclusions to the future of the project should be made with care. For example, exceptional events out of management control could have occurred in the past and may not take place in the future.

(M8) update SYMDB with uncovered SD metrics

The final step in SYDPIM monitoring is to uncover important metrics about the project status using the SD model. This is one of the most useful and distinctive features of a SD project model, because these metrics provide crucial information about the project. Without the SD model they would be difficult or even impossible to measure. Once uncovered, these metrics are then stored in the SYMDB.

SYDPIM proposes that three categories of metrics can be uncovered by the SD project model:

- defect related;
- staff related;
- effects on process metrics.

For each category, SYDPIM proposes a specific set of metrics, as described in table 7.32 (see chapter 7). These metrics are stored in the SYDMDB as series over-time, where each time-point corresponds to a control cycle (e.g. every month). In this example, metrics were uncovered only at the end of the design phase using the prototype model. These are shown in the table 8.16, which presents the actual calibration and the calibration “as planned” (i.e. results if the design phase had been implemented as specified and assumed in the initial plan).

Uncovered SD metrics	Actual	"Planned"	Variation
(1) Defects			
(1.1) Total generated (defect)	283	252	12,3%
(1.2) Generation index (defect/KLOC)	51	45	13,3%
(1.3) Defects escaped (defect)	92	48	91,6%
(1.4) Density of defects escaped (defect/KLOC)	17	9	88,8%
(2) Staff			
(2.1) Fatigue (ordinal scale)	1	0	--
(2.2) Non-reported hours worked (hour)	3	0	--

Table 8.16 – Uncovered metrics about the project status at the end of the design phase

The effects on process metrics are not shown here. These are curves over-time, similar to the one presented in figure 8.15. These metrics suggest that about 18% of the defects generated escaped in the design phase as implemented. This is roughly the same ratio as if work had been implemented as planned. However, in the actual scenario 13% more defects were generated and the bad news are that the number of defects escaped almost doubled. This means that, while not a serious problem (the density of the defects escaped is still at a “normal” level), in the different conditions in which the design phase was implemented overall the quality of the design work and the effectiveness of QA decreased. Careful diagnosis revealed that this was due in great part to some “optimistic” assumptions in the plan.

This type of information is useful to assess the project status and to re-plan the future. For example, having an estimate of the defect density passed onto the next phase helps to establish an appropriate level of QA effort in the future. These metrics also help to diagnoses better what happened in the past.

This is the last activity of SYDPIM monitoring. The next step is to re-plan the future, about which the SD model, now updated with the recent past, provides a new forecast.

8.3.3 SYDPIM planning: assessing the impacts of Fagan inspections

In this example, SYDPIM was used to support a planning decision. This referred to the possible use of the Fagan inspections technique in the software design of the first increment of the C2 software project (SWB1). This took place half-way through the design phase.

The system integration was expected to be complex, incorporating components with new code, others with slightly re-used code and other with heavily re-used components. The final delivery of the system to the Client was subjected to rigid milestones and high penalties for delays. If critical architectural problems emerged in the integration phase, it could be too late to solve the problem. It was therefore critical to ensure a good quality of the designs. Management needed to know what were the possible solutions, and how to best readjust the plan in order to implement them.

A possible solution raised by management was to implement more thorough design reviews. The informal design reviews in use would be replaced by the Fagan inspections technique. The idea was attractive but there were some potential complications: it was known that the Fagan technique could be highly effort-consuming. With tight time-scales, would the benefits outweigh the costs? The Fagan technique would require more time in the design phase. But how much more was really needed? And what other adjustments should be made to the plan so that the technique would be effective?

While traditional PERT/CPM tools could be used to try devising alternative plans, there were difficulties with this option. First, it was too time-consuming. Secondly, most of direct implications of using Fagan inspections could not be represented explicitly in the PERT/CPM plan, quickly, and in a way easy to understand. On the other hand, the indirect implications were of systemic nature and could not be represented in the PERT/CPM plan. Furthermore, the estimates required for the individual detailed PERT/CPM tasks were not available, and so “best guesses” would have to be used. Overall, the outcomes would be more imposed by these guesses than by the systemic impacts of implementing Fagan. Finally, quick “what-if” analysis would be difficult to implement with the gigantic PERT/CPM networks in use.

SYDPIM was used to carry out this analysis. According to the process logic of SYDPIM planning described in chapter 7, some alternative paths can be followed to implement this process. The one followed in chapter 7 to describe the SYDPIM activities is based on the use of the PERT/CPM model as the primary tool to re-plan

the project future. This is also according to the strategy adopted for the case-study, where the SD model was being used as a tentative complement to the PERT/CPM model. The sequence of the SYDPIM planning activities is as follows:

- (P1a) PERT/CPM forecast and analysis of project future
- (P2a) Re-plan project future in PERT/CPM model
- (P5) Update analytical links if necessary
- (P3a) Extract future behaviour from PERT/CPM model
- (P4a) Re-calibrate SD model to PERT/CPM plan
- (P1b) SD forecast and analysis of project future
- (P2b) Re-plan project future in SD model
- (P3b) Readjust PERT/CPM model to SD plan
- (P6) Update target metrics

Like in the previous example the single-phase prototype model was used to simulate the design phase of the whole SWB1 system. Therefore, all the PERT/CPM tasks which refer to the design of all individual components were mapped to this model, in the specification of the SC-WBS link.

The implementation of each SYDPIM activity is now described separately.

Activities (P1a) and (P2a)

These activities refer to the forecast and re-plan of the project future in the PERT/CPM model. When the use of SYDPIM was requested, the PERT/CPM model was already updated with actual results and an updated plan for the future had been developed and was being implemented (which did not include Fagan inspections). Therefore, these two activities, which are a conventional feature of the conventional framework, were already implemented.

(P5) Update analytical links if necessary

The purpose of this activity is to create or update the analytical links of structural correspondence established between the two models, based on the new plan developed for the future. As described in chapter 7, SYDPIM proposes that this is done in the following sequence:

- (1) SC-WBS – PERT/CPM tasks are mapped or re-mapped to the SD-Tasks;
- (2) SC-OBS – PERT/CPM resources are mapped or re-mapped to the SD-Resources;
- (3) SC-WD – PERT/CPM dependencies are mapped or re-mapped to SD-Dependencies (both intra- and inter-task).

In this application, these links were created. All the PERT/CPM tasks that represented design work of all components of SWB1 were mapped to the SD model. Likewise, the resources allocated were also mapped. Inter-task dependencies were not mapped because the design phase was being simulated in isolation.

(P3a) Extract future behaviour from PERT/CPM model

The purpose of this activity is to extract and specify the planned future behaviour from the PERT/CPM model, and store the derived patterns in the object “PERT/CPM future behaviour”. The specific patterns to be derived, the information requirements (data, analytical links, and versions of the PERT/CPM plan), and the calculation processes are presented in tables 7.8 through 7.11 of appendix B, and were discussed in chapter 7. In this application, this SYDPIM process was followed to extract the planned behaviour from the current “non-Fagan” PERT/CPM plan (only one SD-Task was considered).

When this intervention was requested, the design phase was half-way. In an ideal SYDPIM scenario, the SD model would be already calibrated to reproduce the elapsed past of this phase. However, the SD model had not been used before at this level of aggregation to monitor the design phase. In this situation, SYDPIM imposes that prior to using the SD model in planning, it *must* be first calibrated to reproduce the project past. This is an imperative requirement because the SD model stores important information about the “unperceived” project current status (in particular the amount of undetected defects in the system). If the SD model is used to simulate the future without this calibration it will miss this important information, which is likely to have a major impact on the results. The validity of the future projections is not defensible. Therefore the SD model was first calibrated to reproduce the initial plan for the design phase and then slightly readjusted to reproduce some observed minor deviations – this process corresponds to

implementing SYDPIM initial planning followed by the monitoring of current past; as described in chapter 7, this is the recommended approach whenever the SD model is to be used for the first time half-way through the project. Figure 8.17 below illustrates the process followed to specify the planned future of the design phase, where Fagan inspections are not implemented.

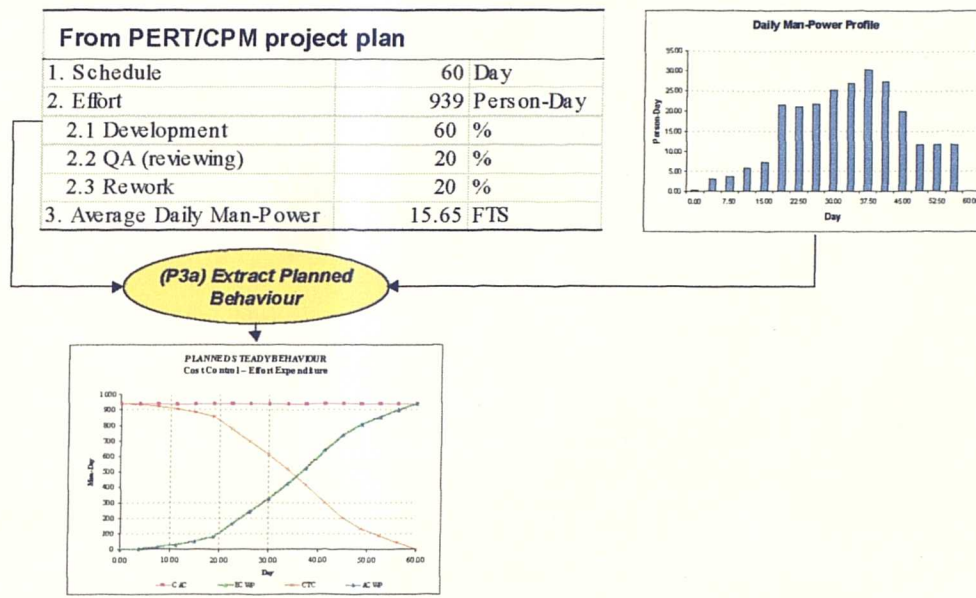


Figure 8.17 – Specification of “non-Fagan” planned future behaviour from PERT/CPM model

(P4a) Re-calibrate SD model to PERT/CPM plan

The purpose of this activity is to calibrate the SD model to reproduce the future behaviour extracted from the PERT/CPM model. This is basically a SD calibration exercise, where the modeller tries to achieve a good fit for plausible reasons.

A detailed algorithm was proposed in the detailed description of this activity in chapter 7. SYDPIM proposes a manual process where human intervention and judgement is essential. In order to support this process, various data links are used to exchange data from the PERT/CPM model to the SD model and to check the consistency of the behaviour produced (various statistics of “goodness-of-fit” are suggested).

In this practical application, this process was implemented and the SD model was calibrated to reproduce the planned behaviour without Fagan inspections.

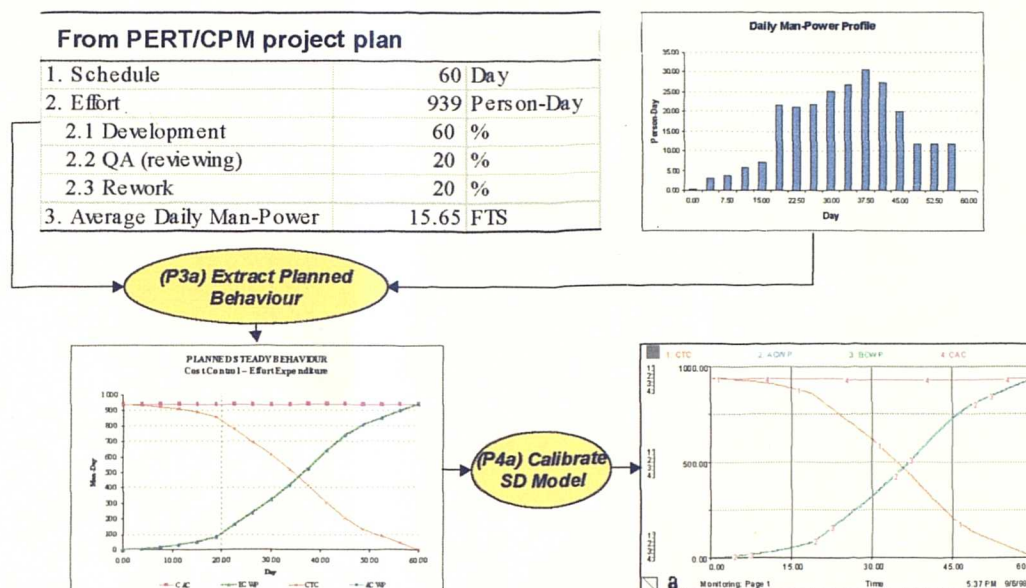


Figure 8.18 – Calibration of SD model to reproduce the planned behaviour of the design phase without Fagan inspections

(P1b) SD forecast and analysis of project future

This SYDPIM activity can be entered either from monitoring or from the calibration of the SD model to reproduce the current PERT/CPM project plan (see SYDPIM planning process logic in figure 7.7). In this practical application, the latter was the scenario at hand. The SD model had just been calibrated to reproduce the “successful” implementation of the non-Fagan PERT/CPM future plan (i.e. exhibiting a steady behaviour).

In this scenario, the primary role of the SD model is to uncover the assumptions underlying the plan and to assess the plan’s sensitivity to risks: is the plan realistic? Is the plan robust against uncontrollable disruptive events? In this practical application management was mainly concerned with the effectiveness of re-planning decision: implementing Fagan inspection. So there was no special focus on uncovering assumptions and on risk assessment. For illustrative purposes, two simple examples of how these analyses could have been carried out is presented.

Regarding the uncovering of assumptions, this is based on using the SD model to extract process related metrics. From the calibration of the non-Fagan plan, some of the metrics extracted are presented in table 8.17 below. The specific process-related metrics presented are illustrative. Other metrics could be derived from these metrics for a more detailed analysis of the plan (e.g. % of defects escaped against the total generated). The last two columns suggest the comparison with benchmark metrics collected from past similar projects. This helps management to identify the likely unrealistic assumptions in the plan (e.g. a too low defect generation index).

Process-metric	Assumed in the plan	Benchmark	Variation
Productivity			
Net (LOC/person-day)	53,87
Gross (LOC/person-day)	32,21
Effort distribution			
Development	59,7%
QA	20,5%
Rework	19,8%
Defects			
Generation index (defect/KLOC)	21,32
Detection index (defect/KLOC)	10,99
Escape index (defect/KLOC)	11,18
Cost to detect (person-day/defect)	0,575
Cost to rework (person-day/defect)	0,559

Table 8.17 – Some of the process-metrics extracted from the non-Fagan PERT/CPM plan

Regarding risk assessment, the analysis requires the following steps: (i) identification of the likely risks, (ii) translation of these risks into parameter changes (the model design elements of the SYDPM development method can support this task), and (iii) simulation to quantify the impacts. As part of generic risk analysis, the risks can be combined to assess their cross-impacts. Tale 8.18 illustrates this process. Two risks are considered, and an extra scenario is produced to combine these risks. The impacts assessed refer to the final cost, schedule, defects escaped (an indication of quality), and staff fatigue (due to working non-reported hours and over-time; the ordinal scale ranges from 0% to 100%). This analysis leads to an interesting conclusion: if the quality of the work is lower, there are no meaningful impacts on cost and schedule and considerable increase n the number of defects escaped. This suggest that the current QA activity does not have the capacity to detect extra defects generated (an argument in favour of implementing the Fagan technique).

Risk	Parameter Changes	Impacts (% variation)			
		ACWP	Schedule	Defects Escaped	Staff Fatigue
1. Work more complex	Productivity lower 40%	12%	10%	18%	15%
2. Lower quality	Defect generation higher 20%	1%	0%	46%	1%
3. Work more complex and lower Quality (1+2)	Both above	12%	10%	55%	16%
...

Table 8.18 – Assessing the impacts of risks using the SD model

There is generally scope for this type of risk assessment activity to be exhaustive. It is therefore important that management focuses on the relevant risks.

(P2b) Re-plan project future in SD model

The purpose of this activity is to improve the project plan in the SD model. In this case the re-planning decision being considered was the implementation of the Fagan inspection technique. This implementation required that the project plan was appropriately readjusted. This included the representation of the Fagan inspection process in the model as well as other changes to the plan in order to better accommodate the requirements of this technique – e.g. management were aware that the schedule of the design needed to be extended.

The SYDPIM Model Development Method encourages that the SD model is developed incorporating all possible re-planning type of decisions (see table 6.10 in chapter 6). However, there can be situations where management wants to test decisions which were not anticipated in the design phase. This was the case with the Fagan inspections technique. When this happens, prior to testing a decision it is necessary to readjust the model structure to represent the decision.

The first step in accomplishing this task was to carry out some literature research, data collection and interviews with managers about the practical implementation of Fagan inspections: what would really change in the designing process? This allowed the author to identify the critical direct impacts of using Fagan inspections.

These direct impacts were then quantified and various factors were incorporated in the model. For example, defect detection was set to be 80% more efficient due to the reduced overlap of defects detected by different reviewers, and due to explicit links to parent and related documents (i.e. every document would explicitly reference its parents and other related ones). Other direct impacts captured included a faster learning curve, a less damaging effect of schedule pressure on quality, and the allocation of QA man-power in direct proportion to the size of the document being reviewed. Verification and validation tests were run to check the model's sensitivity to the new changes, before actually using it.

Then, inside the SD model various Fagan and "non-Fagan" scenarios were considered and tested in a "safe-environment". Would Fagan help to improve the quality of the designs? By how much? And how much would that cost? What are the trade-offs? What would be the best way of accommodating Fagan inspections?

To answer these questions, a multidimensional analysis was carried out considering the following combinations:

(A) Which reviewing technique?

- (1) Conventional
- (2) Fagan inspections

(B) How should the design phase be re-scheduled?

- (1) 60 days with no slippage allowed (as planned)
- (2) 60 days with slippage allowed
- (3) 70 days with slippage allowed
- (4) 80 days with no slippage allowed

(C) How to readjust the QA planned budget?

- (1) Low (15%)
- (2) Medium (20%)
- (3) High (30%)
- (4) Very high (40%)

A total of 32 scenarios were tested through simulation.

The first main conclusion from the analysis was that Fagan inspections would provide significant quality benefits, and more efficiently than increasing the QA level

with the conventional reviews. The numerical results were compiled. Five main scenarios are presented here:

- (1) keep the plan as is and implement Fagan;
- (2) as an alternative to Fagan, increase the QA level from 20% to 40% using the conventional reviewing techniques;
- (3) try Fagan with the current plan but allowing schedule slippage upon staff request (i.e. low schedule pressure);
- (4) try Fagan and extend the design schedule 20 days more. No slippage allowed (i.e. full schedule pressure);
- (5) try 40% of QA level with the conventional techniques and extend the design schedule 20 days more. No slippage allowed (i.e. full schedule pressure).

The results from these five scenarios are shown in figure 8.19 – note that Fagan scenarios are in blue with the strips to the right. Non-Fagan are in red and with the strips to the left. The Y-axis represents a ratio against the base-case (i.e. non-Fagan current plan). The results show that Fagan offers significant quality benefits and, under the same circumstances, performs better than increasing the QA level with conventional techniques.

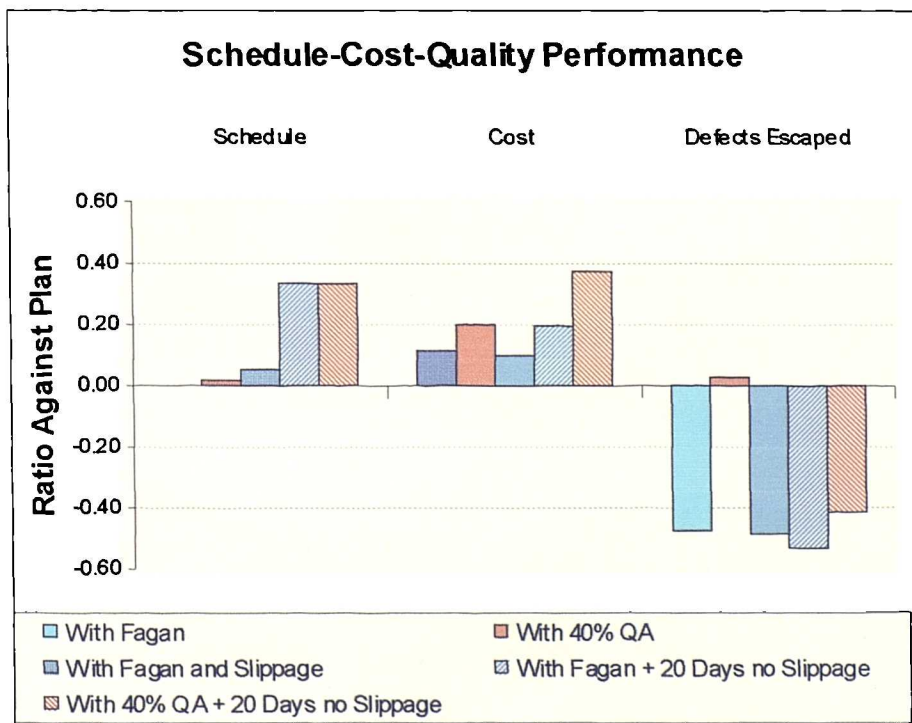


Figure 8.19 – Comparing the SD results of Fagan against non-Fagan scenarios

The second main conclusion was that with Fagan inspections there was more scope to improve quality by giving design more time and further increase the QA level – but this would also be at a higher cost.

Figure 8.20 shows the cost-schedule-quality trade-off of implementing Fagan with an increasing amount of time and effort allocated to QA activities.

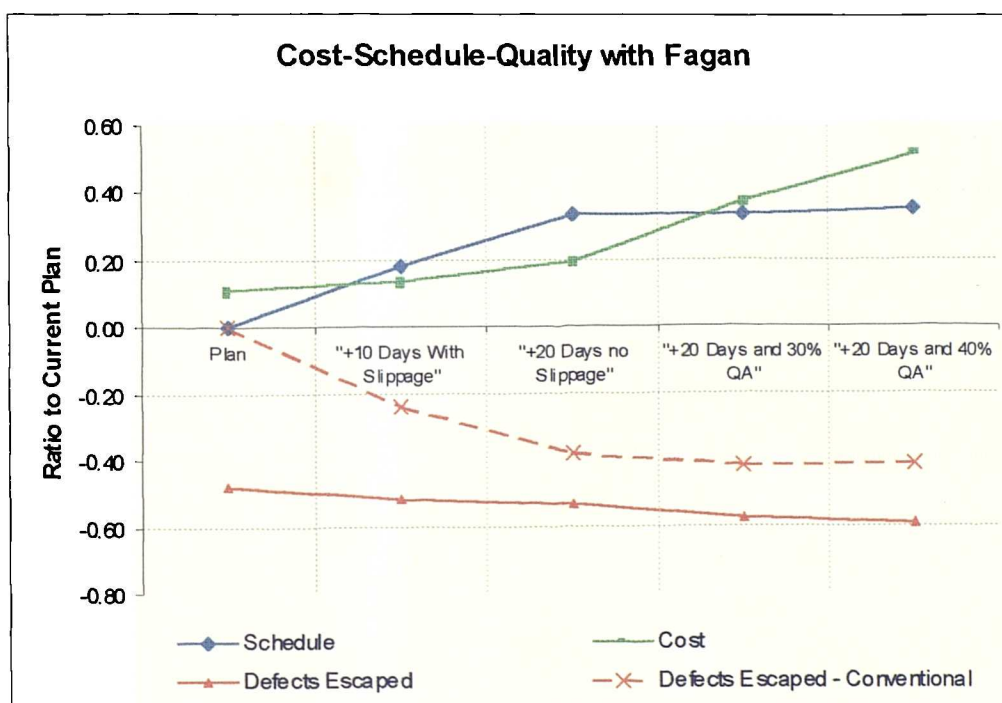


Figure 8.20 – How to best implement Fagan: assessing various scenarios

The dashed red line (defects escaped) refers to the use of conventional techniques under similar conditions. By comparison with the thick red line of Fagan (also defects escaped), it can be seen that the performance of the conventional techniques reaches a “plateau” after which increasing more the QA effort brings little extra quality benefits. The same is not true with Fagan inspections, where quality continues to improve.

(P3b) Readjust PERT/CPM model to SD plan

Once the “optimal” Fagan scenario was identified (a subjective management judgement process combining the schedule, cost and quality impacts, as forecasted

by the SD model), the next step was to translate the new plan into the PERT/CPM model, according to the analytical links established between the two models. For reasons of confidentiality the actual decision is not identified here.

Figure 8.21 considers an hypothetical scenario (30% QA with 20 days slippage), to illustrate how this SYDPIM activity is carried out. The SYDPIM Project Management Method proposes that this process of readjusting the PERT/CPM model to a new SD plan requires human intervention and judgement. However, it is also proposed that this can be semi-automated through the use of SYDPIM analytical links. A detailed algorithm was proposed for this purpose in the detailed description of this activity in chapter 7.

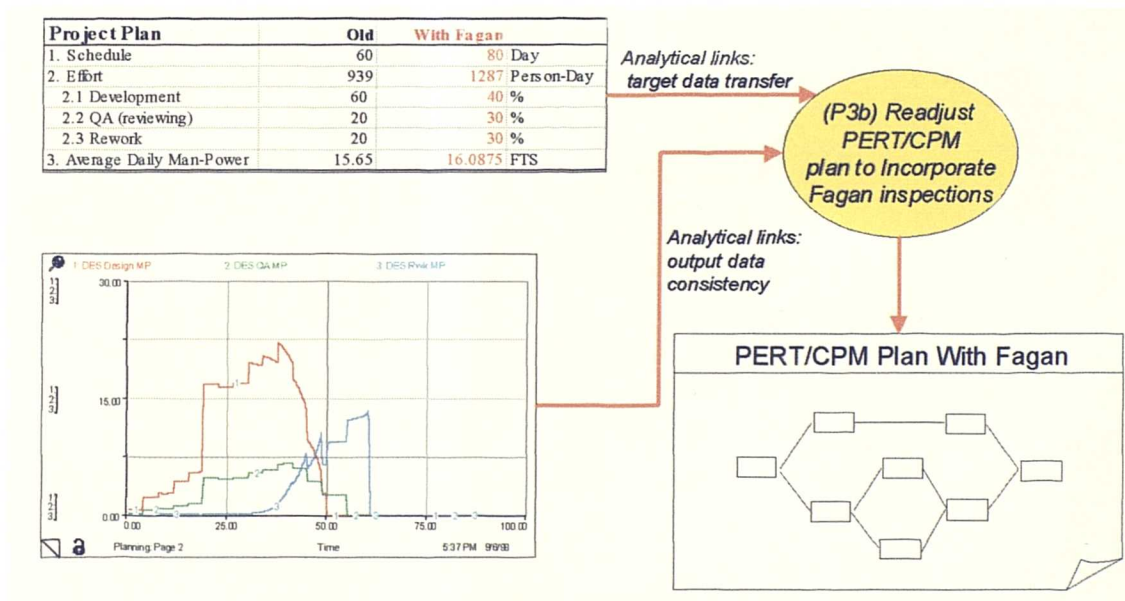


Figure 8.21 – Transferring the selected Fagan plan from the SD model to the PERT/CPM model

(P6) Update target metrics

As discussed in chapter 7, the purpose of this activity is to update the target metrics in the SYDPIM objects SYMDB, “PERT/CPM past behaviour” and “Project past behaviour”. This is required because changes in the future PERT/CPM plan carried out in the previous activities, may affect the project targets.

In this particular example it was assumed that the management re-planning decision was to increase the QA fraction up to 30% and extend the schedule by 20 days.

The schedule extensions implied an increase in the planned budget as well as in the planned staff profile for the remaining of the design phase, as shown in figure 8.20. Therefore, the present value of these three targets metrics were updated in the objects SYMDB, "PERT/CPM past behaviour" and "Project past behaviour". The new values were derived from the updated PERT/CPM plan using the calculation process described in tables 7.8 through 7.11, and according to the condition in table 7.14 (in this case a single SD-Task not completed). As an example, the new values for the metrics CAC and SAC are shown in figure 8.20 above.

8.3.4 SYDPIM planning: assessing the impacts of Client behaviour

In this example, SYDPIM was used to support risk analysis within planning. The risks addressed had to do with Client behaviour. Management was concerned with requirements changes being requested by the Client, towards the end of the design phase. At that stage, the detailed and formal system design would be nearly complete. Changes would therefore become expensive and would delay a critical milestone: the completion of the design phase. Management felt that such changes would inevitably lead to major knock-on disruptions towards the end of the project. A broader discussion of this practical application of SYDPIM can be found in Rodrigues and Williams (1998).

In general, Client behaviour was perceived to be of crucial importance within the management of the KDCOM project. Of particular relevance was the subjectivity involved in interpreting the several contractual agreements. This problem could be exacerbated by cultural differences, which tend to create major obstacles to effective communication. Consequently, the threat of changes being introduced in the system requirements, particularly during the middle and later stages of the life-cycle, posed a major risk. Estimating and quantifying the final impacts of these changes on the project (major schedules, cost, and product quality), were recognised by management to be a major problem as were the inadequacies of the traditional PERT/CPM tools in capturing their secondary downstream effects. In this practical application, SYDPIM was implemented to illustrate how the methodology can be used to address this managerial need.

When this analysis was carried out, there were no actual changes requested by the Client. The aim was to develop a pro-active risk analysis exercise, which would help management to assess the impacts of likely changes, as well as to identify the more effective mitigating actions to negotiate with the Client. In this context of negotiation, the SD analyses could be shown to the Client as a means to demonstrate the negative impacts of keeping the original plan if change requests occurred. Due to various “political” sensitivities involved, the particular analysis carried out was fictitious, based on the SD calibration for the past outcome of the design phase of the SAM/STIR component (already described above in this chapter) – therefore, a retrospective analysis was carried out. One of the advantages of using this scenario was that the calibration of the process metrics (e.g. productivity, learning curves) was closer to reality (hence “more valid”). While a fictitious scenario, this illustrated how a same type of analysis could be applied to other areas of the project. Furthermore, some lessons learned and conclusions could be considered as generic and hence applicable to similar situations in the future. Overall, the focus of this analysis was on the importance of managing changes in the early stages of design, when a detailed plan is not available for the later stages. Some of the metrics presented in this example have been “disguised”, but are based on real data collected from the project.

This SYDPIM based risk analysis took place within planning. According to the SYDPIM process logic, a possible the sequence of the SYDPIM activities is as follows:

- (P1a) PERT/CPM forecast and analysis of project future
- (P2a) Re-plan project future in PERT/CPM model
- (P5) Update analytical links if necessary
- (P3a) Extract future behaviour from PERT/CPM model
- (P4a) Re-calibrate SD model to PERT/CPM plan
- (P1b) SD forecast and analysis of project future
- (P2b) Re-plan project future in SD model
- (P3b) Readjust PERT/CPM model to SD plan
- (P6) Update target metrics

This sequence puts emphasis on using the PERT/CPM model as the primary re-planning tool, a strategy adopted in the case-study. The implementation of these

activities was similar to the process described in the previous example and a detailed description would be extremely lengthy. The implementation of the SYDPIM activities is now briefly described separately for each activity, focusing on the analysis carried out in the SD model.

Activities (P1a) and (P2a)

The initial PERT/CPM plan for the design phase of the SAM/STIR component was considered.

(P5) Update analytical links if necessary

The purpose of this activity is to create or update the analytical links of structural correspondence established between the two models, based on the new plan developed for the future. Since the scenario used was the calibration for the design phase of the SAM/STIR component, the analytical links had already been updated (as described in the first example).

Activities (P3a) and (P4a)

There was already a calibration available for the actual past of the design phase of the SAM/STIR component. However, this calibration did not portray a steady behaviour with constant targets (as shown in figure 8.11). There was also a calibration available for the initial plan (as shown in figure 8.10), but this was based on “less valid” learning curves regarding the key process metrics. The strategy was therefore to re-calibrate the nominal values of the process metrics so that a steady behaviour was achieved (i.e. the outcome as planned), and keep the learning curves of the post mortem calibration. A calibration closer to reality was therefore achieved in this way, because the variations imposed by the learning curves were closer to reality (as opposed to the initial calibration where they were constant). In this calibration no Client actions were considered.

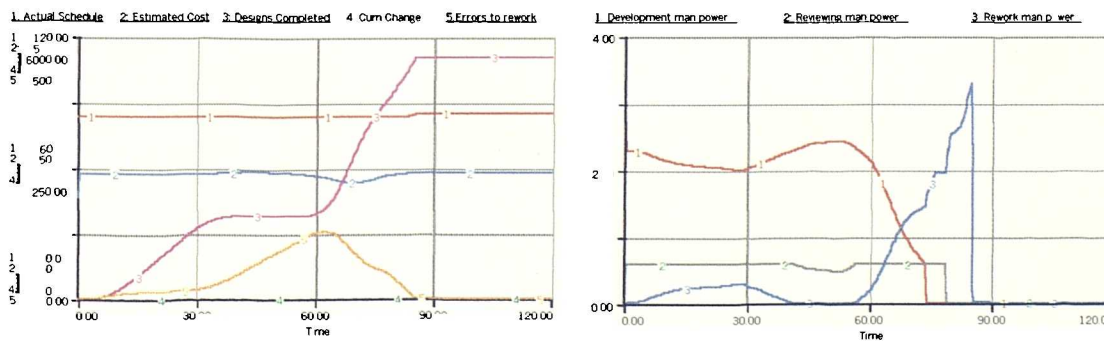
The design phase of the SAM/STIR component was planned according to the figures shown in table 8.19. The estimated size of this functional area, measured in DSI (delivered source instructions, or lines of code), represented about 12% of

SWB1, which incorporated another eight functional areas. The breakdown of this budget was based on a project-wide policy of allocating 19% to each of QA (design reviews in this case) and rework activities. The average daily man-power was 2.88 full-time-staff (FTS), and given the short time-period management were not considering changing the staff level after the work had been initiated.

Metric	Planned Estimate	Unit
1. Estimated size	5 551	DSI
2. Schedule	84	day
3. Budget	242	man-day
(excluding management)	62% (150)	
3.1 Development	19% (46)	
3.2 QA (reviewing)	19% (46)	
3.3 Rework		
4. Productivity		DSI/man-day
4.1 Gross (1/3)	23	
4.2 Development (1/3.1)	37	
5. Daily man-power (3/2)	2.88	FTS

Table 8.19 – Planned targets for the design phase of SAM/STIR

The general dynamic behaviour produced by the SD model when calibrated for this plan is shown in figure 8.22(a) below, according to which both targets of cost and schedule remain roughly constant (curves 1 and 2); curve 3 shows the work progress and curve 5 the amount of remaining rework.



(a) General project behaviour (design phase)

(b) Man-power allocation

Figure 8.22 – The base case: steady behaviour produced with the model calibrated according to the plans

Figure 22(b) shows how the man-power is scheduled among the three engineering activities. According to the calibration, productivity would slows down in stage 2 (requirements clarification) due to an increased complexity in the design work. The

consequence is an increase of man-power allocated to development in this stage and a slow down in the rework activity. As staff gets more familiarised with the details and productivity increases, man-power is re-directed to rework towards the end of stage 3; finally there is a week of revision and a week of rework. The model suggests a light "QA skipping" which is possibly caused by the difficulties in stage 2.

(P1b) SD forecast and analysis of project future

In this sequence of SDYPIM activities, the SD model was just calibrated to reproduce the steady behaviour extracted from the PERT/CPM plan. The SD analysis therefore starts with assessing whether the assumptions uncovered by the calibration are realistic (e.g. is the defect generation rate too low?). The next step is to assess the plan's sensitivity to risks. In this case, the risk to be introduced was the Client requesting requirements changes. The aim of this analysis was to identify the likely impacts and whether the project plan needed to be readjusted.

Calibrating the SD model to analyse the impacts of changes required the definition and quantification of some parameters:

- (1) a measure for the magnitude of the changes, defined as the percentage of work developed that needed to be re-done due to the changes;
- (2) a description of how these changes are introduced over time, defined by an "S-shaped" curve identifying the cumulative fraction of (1) being introduced over time: the steeper the shape the more sudden the introduction of changes would be;
- (3) identification and quantification of the *direct* effects of changes on several process metrics (e.g. error generation, development productivity, effort to rework). Based on managers' opinion some direct effects were selected and their impacts were tested in the model. Where the impact is irrelevant, inclusion of the parameter could not be justified. After experimentation, it was found that the most relevant effect was the impact on error generation. Besides, the later the change is introduced the stronger the effects of "work out of sequence", and hence we also considered that these would grow exponentially. These assumptions were tested separately with simple scenarios.

When changes are introduced there is an immediate need to reassess whether the final schedule should be readjusted. While the contractor reaches an agreement with the Client regarding the milestone, management has then to decide how to pass this onto staff. Typically, management agrees a shorter deadline with development staff as both a contingency measure and to ensure a minimum level of schedule pressure (the scheduled duration shown in table 8.19 refers to this date while the agreement with the client was of 100 days, hence 16 days of contingency). This involves deciding the following policies:

- (i) how management reacts to slippage reported by the development staff in order to avoid over-running the schedule milestone. This was captured in the model according to the curve shown in figure 8.23: the x-axis represents an ordinal measure of how close the completion date reported by the development staff is to the schedule agreed with the Client (a value of 0 represents that the staff is reporting within the initially agreed schedule, and 1 represents reporting a delay which will extend the completion date to the schedule milestone agreed with the Client); the y-axis represents the fraction of the consequent schedule extension requested by the staff that managers are willing to accept. As an example, if the staff reports a completion date of 92 days (half-way between 84 and 100, hence 0.5 in the x-axis), a delay of 8 days, then managers are only willing to accept 40% (y-axis) of that extension (i.e. $40\% \times 8 = 3.2$ days), and hence the schedule agreed with the staff would be extended to day 87. The overall policy, based on current practices, shows that while early reported delays are not accepted and pressure is put on the staff, as slippage becomes closer to the schedule milestone agreed with the Client, management starts accepting a fraction of this slippage in order to avoid the negative effects of pressure. However, once these delays have been allowed, management again starts refusing further reported slippage threatening the schedule milestone.
- (ii) given a certain amount of introduced changes how many extra days are given to the development staff in order to minimise slippage within an acceptable level of quality. This is captured in the model by a policy parameter representing the proportion of schedule adjustment given to the development staff in relation to the proportion of the changes introduced, ranging from 0% to 100%. As an example, a 50% level means that given the duration agreed with the Client of 100 days, a 30% introduction of changes would be matched by giving to the staff a

schedule adjustment of 15 days (i.e. $100 \times [30\% \times 50\%]$) – we herein refer to this parameter as the "adjustment fraction."

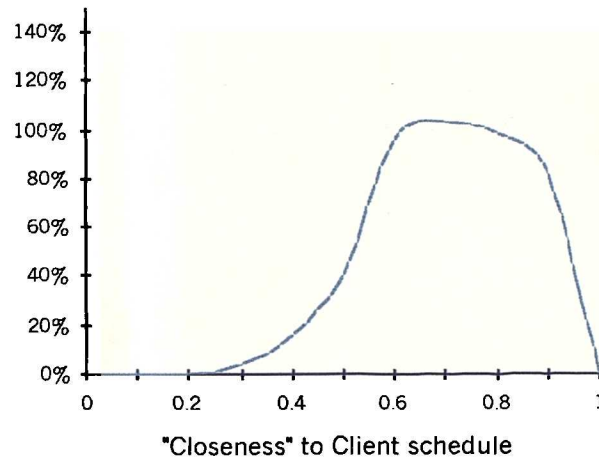


Figure 8.23 – Management policy: controlling schedule contingency

In this example a 30% magnitude of total demanded changes was considered, being introduced smoothly, starting at 75% of the planned progress and ramping at 90%. The model allowed assessing the impacts of these changes at the end of the design phase and to test several schedule adjustment policies. The main purpose of this experimental investigation was to provide a qualitative *and* quantitative analysis of the triangular trade-off between cost, quality, and time. While the long term cost savings of early error detection are considerable, it is difficult to estimate this during the design phase, when the product has not yet materialised. Therefore, the problem of managing changes within this phase is not so much about controlling the cost and schedule but assessing how these are traded against quality. This experimental investigation focused on this problem.

The assessment of different scenarios was based on three output metrics: schedule achieved, cost, and defects escaped. Providing explicit estimation of this last intangible as a measure of quality is of crucial importance. In practice, using the numbers of detected errors leads to the self-fulfilling tendency of skipping QA after "enough" errors have been detected. In analysing schedule adjustment policies the curve of figure 8.23 was not changed since it represents a project-wide generic

policy of managing schedule contingencies. Instead, the parameter "adjustment fraction" was changed in each simulation in order to investigate how the schedule agreed with the development staff should be readjusted in the face of introduced changes.

Figure 8.24 shows the behaviour produced when changes are introduced and no schedule adjustment is made. Here "schedule pressure" is put at its maximum as an attempt to cope with the changes and still achieve the original schedule. The dashed line (curve 4) in figure 8.24(a) shows the "S-shaped" introduction of changes starting roughly at day 65. The impacts on the general behaviour are an immediate steep decrease in the amount of work completed (curve 3) while later schedule slippage cannot be avoided (curve 1), and cost exceeds the original budget. Figure 8.24(b) shows how all man-power is suddenly directed into development in an attempt to implement the changes quickly. Consequently, rework is delayed and there is a cut in the man-power allocated to the reviews (curve 2). This "QA cut" does not affect the review progress: under high schedule pressure staff spend less time reviewing documents but still report a normal reviewing rate. As a result of this poorer QA activity more defects are likely to escape. After changes have been implemented and development is completed, man-power is directed back to rework and reviewing.

Although this restrictive scheduling policy does not avoid a delay, the results still appear attractive: a 14% over-run and a 13% over-spend (less than half of the proportion of changes). Furthermore, the increase in the amount of defects detected was in the same proportion of the changes (about 30%), reflecting how staff implicitly assess QA progress when under schedule pressure: "if I have to re-design 30% more then I should find 30% more defects". However, the (intangible) number of defects escaped suffered a steep increase of 72%. This is a consequence of two major secondary effects: an increase in error generation, and a decrease in QA effectiveness. Although the long term consequences of a high defect escape rate are not visible at the end of the design phase, they will emerge later in the project when corrective actions are more difficult and very expensive.

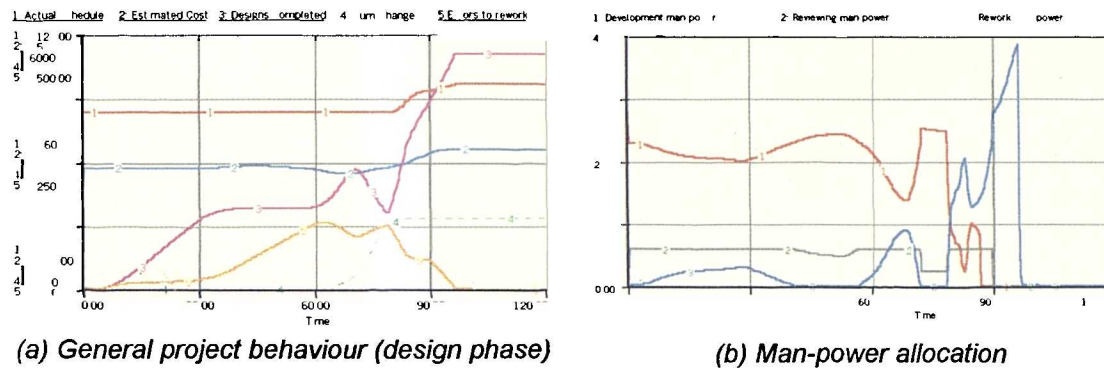


Figure 8.24 – Unsteady behaviour produced when the model is calibrated with changes and no schedule adjustment

The conclusion from the analysis carried out in this SYDPIM activity was that if the Client was to request requirements changes, the project plans would need to be readjusted in order to mitigate the impacts of poor design quality. The next step is therefore to identify the most effective re-planning actions. This takes place in the following SYDPIM activity.

(P2b) Re-plan project future in SD model

In order to mitigate the negative impacts of requirements changes, management considered extending the completion date of the design phase as the most viable solution. The model was therefore used to explore the potential benefits of extending the schedule, as a way to reduce the negative effects of schedule pressure. Figure 8.25 shows the deviations from the base case (when no changes are introduced) for various levels of schedule extension. Alleviating schedule pressure takes effect beyond the 20% level of the parameter "adjustment fraction", where increasing the schedule adjustment to balance the changes results in later completion and more over-expenditure. However, the gains in the number of defects escaped are of much greater magnitude. Beyond the 60% level the quality of the designs does not improve significantly while cost and schedule keep increasing. This suggests that an appropriate level of schedule adjustment would be around 60%. For example, given the duration agreed with the Client of 100 days, if 30% of changes are introduced, then 18 days (i.e. $100 \times [30\% \times 60\%]$) of delay should be passed on to staff. In our example the completion time would be extended from 84 days to 102 days, as changes are continuously introduced.

Testing this policy in the model showed that the actual completion date would be day 104, a 4 day over-run in the eyes of the Client.

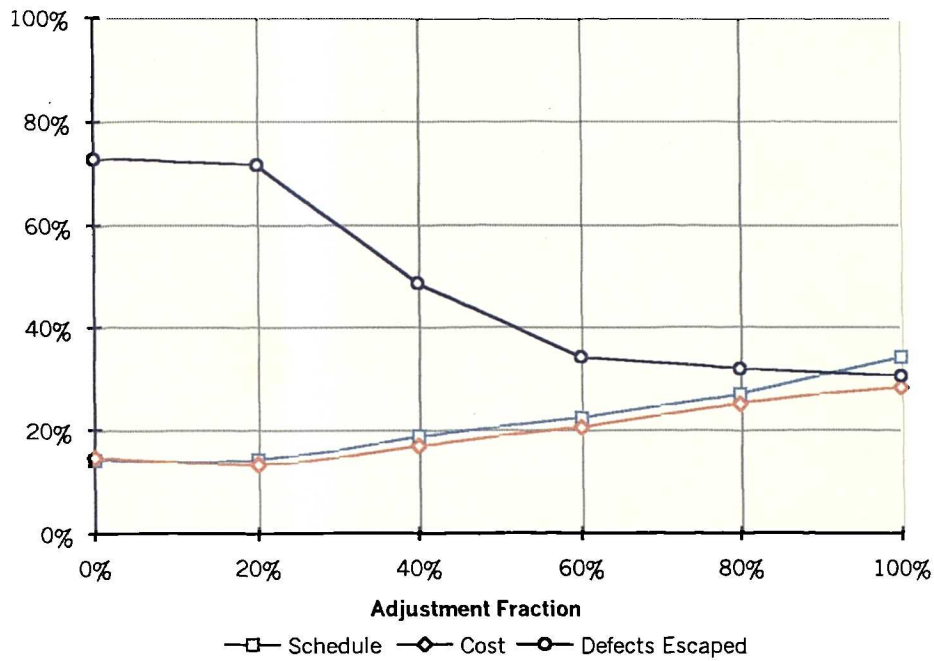
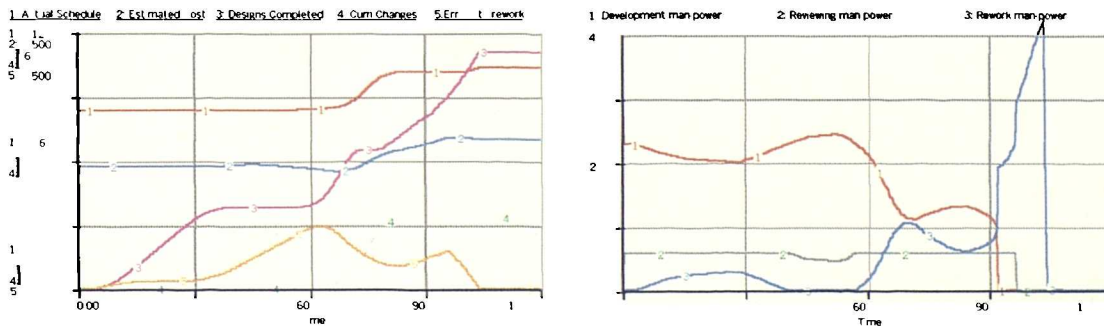


Figure 8.25 – Impact on results of different levels of schedule adjustment

The behaviour produced is shown in figure 8.26, where the patterns assume smooth shapes similar to the base case (figure 8.21).



(a) General project behaviour (design phase)

(b) Man-power allocation

Figure 8.26 – Behaviour produced when the model was calibrated with changes and with a 60% schedule "adjustment fraction"

In the case where the schedule milestone of 100 days could not be extended the model would suggest an adjustment fraction around 40% (12 days). For a more detailed analysis, numerical results are shown in table 8.20.

Results	Base Case	Adjustment Fraction						Unit
		0%	20%	40%	60%	80%	100%	
Schedule	85	97	97	101	104	108	114	Day
Cost	241	276	273	282	291	301	309	man-day
Productivity	23.0	20.1	20.3	19.7	19.1	18.4	18.0	DSI/man-day
Defects Detected	1.00	1.30	1.28	1.40	1.45	1.45	1.45	error/KDSI (*)
Defects Escaped	1.00	1.72	1.71	1.48	1.34	1.32	1.31	error/KDSI (*)

Table 8.20 – Numerical results: percentage deviations from base case

(*) defect metrics have been normalised

The main conclusion from this SYDPIM investigation was that when changes are introduced later in the design phase, management must not try to keep a tight schedule. This is for two reasons: (1) implementing design changes under pressure exacerbates error generation and disrupts QA effectiveness, so a high number of defects are likely to escape; and (2) while the quality of the designs is not tangible at this stage in the development life-cycle, the long term impacts of low design quality are very important in later phases.

The main final output from this SYDPIM activity is an improved future plan, which results from imposing changes in the existing plan. In the following SYDPIM activity this plan is converted back to the PERT/CPM model. However, in the present application the analysis carried out in this activity was a pro-active risk analysis where mitigating re-planning actions are identified for potential risks. In situations like this, management may not want to implement these re-planning actions at the present moment. Instead, management records this actions in the risk register and implements them only if the specific risk emerges. In this case, management decided that if the Client would ask in the future for requirements changes, the schedule would readjusted as recommended by the analysis above. This is another important output from this SYDPIM activity: risk mitigating policies (instead of specific decisions). Another possible output would be a list of specific risk scenarios and correspondent mitigating actions.

(P3b) Readjust PERT/CPM model to SD plan

Because management did not decide to readjust the future plan at this stage, the current plan was not changed and therefore there was no need to readjust the PERT/CPM model.

(P6) Update target metrics

Once more because the current PERT/CPM plan was not changed from the SD analysis, these metrics did not need to be updated (i.e. the project targets did not change).

8.3.5 Calibrating for the plans of SWB2

The plan for the case-study was to test and refine the SYDPIM methodology during the first increment of the software project (SWB1), where the SD model would be used for post mortem type of analysis. As shown by the examples of application just described, not only post mortem analyses were carried out, as pro-active applications were also performed. Once SWB1 was complete, the strategy for the case-study was thereafter to leave the SD models and the methodology available to be used pro-actively in the second increment (SWB2).

For this purpose the, SD model was calibrated to reproduce the initial PERT/CPM plan of SWB2. The development of this plan covers activities (P1a) and (P2a) of the SYDPIM process logic. The following activity (P5) consists of creating the analytical links of structural correspondence between the SD model and the PERT/CPM model. This consists of mapping tasks, resources and work dependencies. This mapping is not described here in detail. Overall, the development process was vertically aggregated in the SD model into three main “chunks” of tightly coupled software components. The next step was to extract the planned future behaviour for the SWB2 project, and to calibrate the SD model to reproduce this behaviour. This corresponds to activities (P3a) and (P4a). As an example, the implementation of these activities is here briefly illustrated for one of the “chunks” of SWB2. This was planned for an overall duration of 556 days, and with a budget of 2 585 person-day.

Figure 8.27 shows some of the behaviour patterns extracted from the initial PERT/CPM plan, according to the calculation processes proposed in tables 7.8 through 7.11 (see chapter 7). The dots in blue represent the patterns as extracted from the PERT/CPM model: SAC[t], ACWP[t], CAC[t], CTC[t], PSP[t]. The lines in pink represent the behaviour reproduced by the SD model. The process followed to calibrate the SD model was based on the algorithm proposed in the formal description of the SYDPIM activity (P4a) (see chapter 7). This visual representation suggest that the SD model is reproducing well the planned steady behaviour.

Figures 8.28, 8.29 and 8.30 illustrate how the “goodness of fit” was assessed quantitatively throughout the case-study, and as proposed by SYDPIM in activity (P4a) (see chapter 7). Figure 8.28 shows a summary of the Theil’s statistics and the correlation coefficient. Both the correlation coefficient and the RMSPE indicate a very close fit. For most of the patterns, even the minor deviations observed are due to the “point prediction” component of the “error”, as desired (Sterman 1984). Figure 8.29 shows the detailed mathematical calculation of these indices.

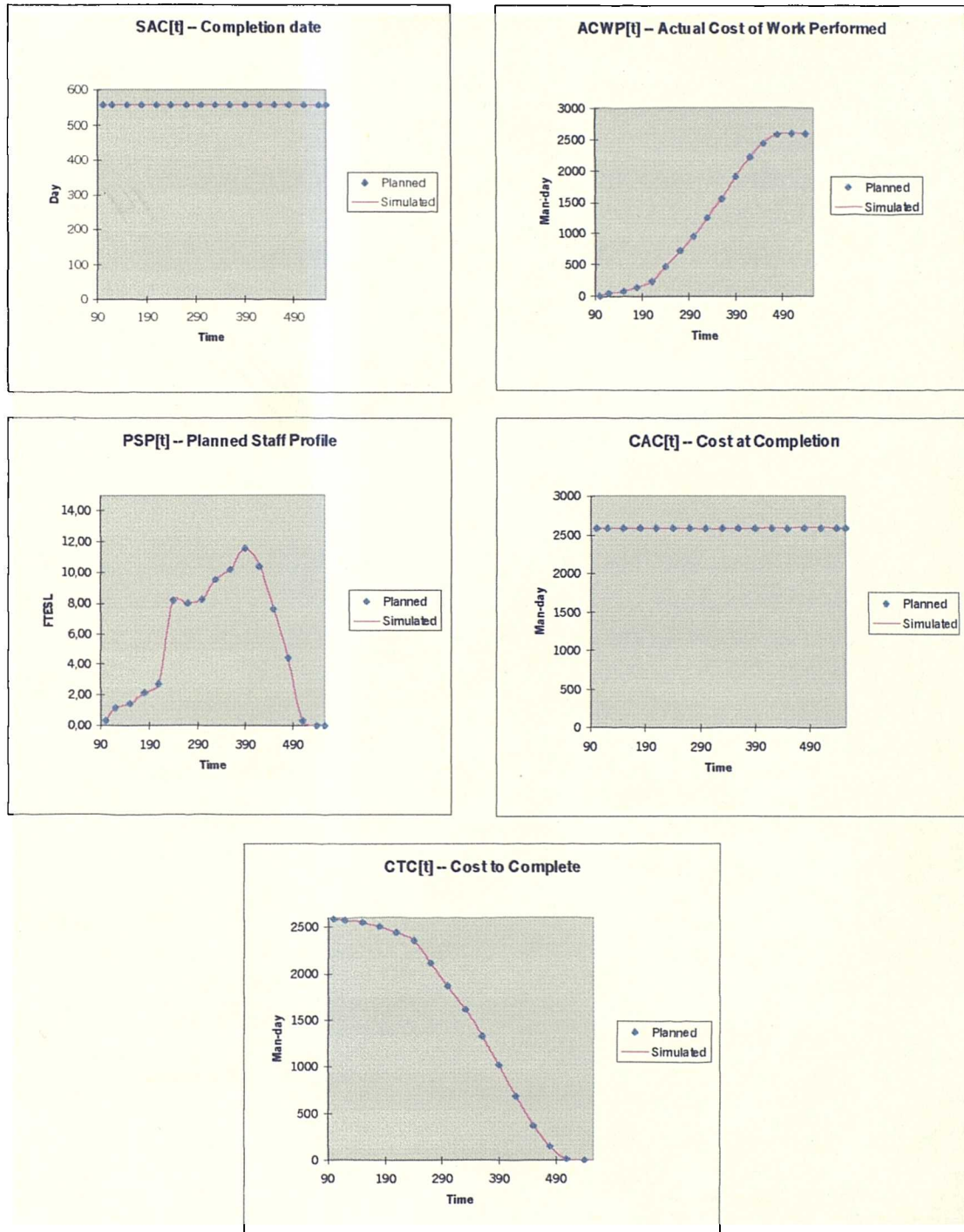


Figure 8.27 – Example of some patterns of the “PERT/CPM future behaviour” extracted from the initial plan and reproduced by the SD model

"Goodness of fit" of reproduced behaviour (summary)

Metric	Correlation	RMSPE	Bias	Trend	Point Prediction
1. Schedule - SAC[t]	1,00	0,31%	5,88%	94,12%	0,00%
2. Effort					
2.1 ACWP[t]	1,00	0,13%	27,91%	50,43%	21,66%
2.2 CAC[t]	1,00	0,17%	23,26%	76,74%	0,00%
2.3 BCWP[t]	1,00	0,36%	3,32%	7,48%	89,19%
2.4 CTC[t]	1,00	0,47%	3,32%	7,48%	89,19%
3. Staff - PSP[t]	1,00	2,22%	6,15%	7,67%	86,18%

Figure 8.28 – Indices of "goodness of fit" for some behaviour patterns of the "PERT/CPM future behaviour" reproduced by the SD model: correlation coefficient and Theil's statistics

"Goodness of fit" of reproduced behaviour (detail)												
	SAC[t]		ACWP[t]		CAC[t]		BCWP[t]		CTC[t]		PSP[t]	
	(s-a) ²	((s-a)/a) ²	(s-a) ²	((s-a)/a) ²	(s-a) ²	((s-a)/a) ²	(s-a) ²	((s-a)/a) ²	(s-a) ²	((s-a)/a) ²	(s-a) ²	((s-a)/a) ²
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,00	0,00	0,00	0,00	0,00	0,01	0,00	0,01	0,00	0,01	0,00	0,00	0,00
0,00	0,00	0,00	0,00	0,00	0,01	0,00	0,01	0,00	0,01	0,00	0,00	0,00
0,00	0,00	0,00	0,00	0,00	0,02	0,00	0,02	0,00	0,02	0,00	0,00	0,00
0,00	0,00	0,00	0,00	0,00	0,02	0,00	0,02	0,00	0,02	0,00	0,00	0,00
0,00	0,00	0,00	0,00	0,00	0,03	0,00	0,03	0,00	0,03	0,00	0,00	0,00
0,00	0,00	0,00	0,00	0,00	0,03	0,00	0,03	0,00	0,03	0,00	0,00	0,00
0,00	0,00	0,00	0,00	0,00	0,14	0,00	0,13	0,00	0,13	0,00	0,00	0,00
0,00	0,00	0,01	0,00	0,00	1,61	0,00	1,45	0,00	1,45	0,00	0,00	0,00
0,00	0,00	0,14	0,00	0,00	6,10	0,00	4,39	0,00	4,39	0,00	0,00	0,00
0,00	0,00	0,66	0,00	0,00	2,19	0,00	0,44	0,00	0,44	0,00	0,00	0,00
0,00	0,00	0,45	0,00	0,00	77,79	0,00	66,46	0,00	66,46	0,00	0,01	0,00
0,00	0,00	29,72	0,00	0,00	106,50	0,00	23,70	0,00	23,70	0,00	0,01	0,00
0,00	0,00	43,32	0,00	0,00	42,64	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,00	0,00	38,41	0,00	0,00	37,33	0,00	0,01	0,00	0,01	0,00	0,00	0,00
0,00	0,00	32,44	0,00	0,00	32,26	0,00	0,00	0,00	0,00	0,00	0,00	0,00
49,00	0,00	32,38	0,00	0,00	28,73	0,00	0,12	0,00	0,12	0,00	0,00	0,00
Averages	2,88	0,00	10,44	0,00	19,73	0,00	5,69	0,00	5,69	0,00	0,00	0,00
RMSPE (%)		0,31%		0,13%		0,17%		0,36%		0,47%		2,22%
UM bias		5,88%		27,91%		23,26%		3,32%		3,32%		6,15%
US - trend		94,12%		50,43%		76,74%		7,48%		7,48%		7,67%
UC - point prediction		0,00%		21,66%		0,00%		89,19%		89,19%		86,18%
Mean actual		556,00		1162,77		2585,00		1162,77		1422,23		5,08
Mean simulated		555,59		1164,48		2587,14		1162,34		1422,56		5,09
Dev actual		0,00		1005,1		0,00		1005,1		1005,1		4,1
SDev simulated		1,65		1008,42		3,89		1005,48		1005,48		4,14
Correlation Coefficient		1,00		1,00		1,00		1,00		1,00		1,00

Figure 8.29 – Calculation of the Theil's statistics of "goodness of fit"

Accuracy of final results

Metric	Planned	Simulated	Deviation	
			Absolute	Percentage
1. Schedule	556,00	549,00	-7,00	-1,26%
2. Effort				
2.1 Development	1551,00	1553,42	2,42	0,16%
2.2 QA	517,00	524,26	7,26	1,40%
2.3 Rework	517,00	513,01	-3,99	-0,77%
2.4 Total	2585,00	2590,69	5,69	0,22%
3. Productivity (LOC/man-day)				
3.1 Overall	23,41	23,36	-0,05	-0,22%
3.2 Development	39,02	38,96	-0,06	-0,16%

Figure 8.30 – Indices of "goodness of fit" for some behaviour patterns of the "PERT/CPM future behaviour" reproduced by the SD model: absolute and percent deviations of cumulative results

Figure 8.30 shows the absolute and the percent deviations of the most relevant patterns, as suggested in the formal description of SYDPIM activity (P4a) see

chapter 7. The deviations from the planned cumulative results are minor thereby suggesting a good level of fit.

Once the SD model was reproducing the initial PERT/CPM plan for SWB2 within an acceptable level of accuracy, the process followed to activity (P1b) where the project future is analysed. This includes assessing whether the underlying assumptions in the plan are realistic, and assessing the impacts of various risks. In the following activities, the plan is eventually improved within the SD model and is finally transferred back to the PERT/CPM model. The implementation of these activities within SYDPIM planning at the beginning of SWB2 is not described here in detail. However, the process followed was similar as described in the previous examples. In reality, various risks and scenarios were raised by management, according to the specific situation.

8.3.6 SYDPM planning: assessing the impacts of introducing inexperienced staff

This application took place after the successful completion of SWB2 on schedule and first release of the C2 SW system (early 1998). The work at the KDCOM project then moved to a second batch (called Batch II), aimed at delivering an improved release of the system. As part of the contractual agreement regarding technology transfer, staff from the Client would participate in this batch. Management wanted to assess and quantify the impacts of introducing this inexperienced staff and how to best mitigate the potential problems. Management wanted to analyse the impacts first within the development phase (including design and coding up to system integration), and then for the full life-cycle of the project. The application of SYDPIM was requested for this purpose.

The SYDPIM analysis took place up-front at the beginning of the batch, where an initial PERT/CPM plan was available but implementation had not started yet. In terms of SYDPIM planning process logic, this implies that activities (P1a) and (P2a) had already been carried out. The next step is activity (P5), where the analytical links are created in order to map tasks, resources and dependencies into the SD model. The following activities (P3a) and (P4a) consist of extracting the future behaviour from the PERT/CPM model and calibrating the SD model to reproduce

this behaviour. The initial PERT/CPM plan contained the following information shown in table 8.21.

Planned targets	Full life-cycle	Development	Unit
Estimated size	100 K	100 K	DSI
Schedule	12	7	Quarters
Budget	48	28	Person-Quarter
Staff profile (constant)	4	4	FTS

Table 8.21 – Initial planned targets extracted from the PERT/CPM model

This plan did not consider the involvement of any staff from the Client. The four engineers were fully experienced members of the project team. Figure 8.31 below shows the *steady* behaviour produced by the SD model at the end of activity (P4a), when calibrated to reproduce the plans for the development phase. Figure 8.32, shows the same behaviour regarding the full project life-cycle.

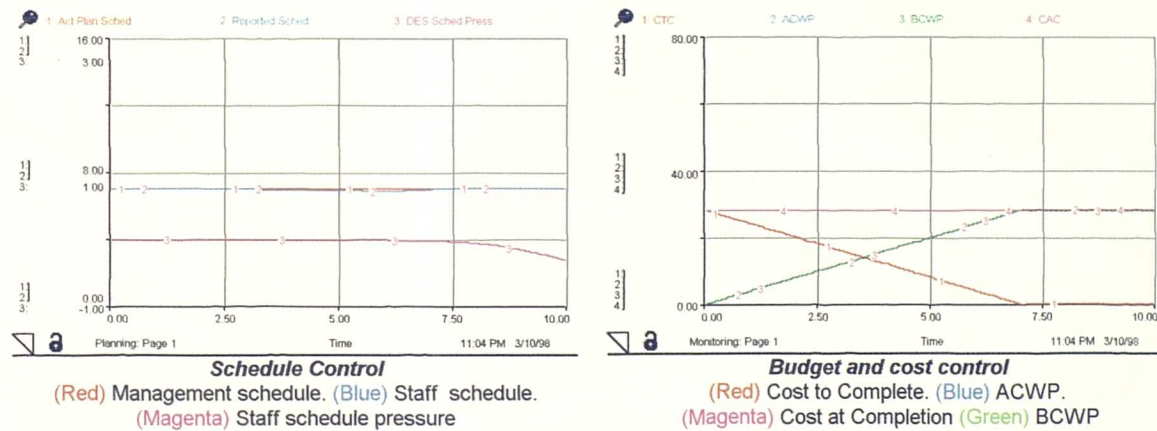


Figure 8.31 – Steady behaviour produced by the SD model when calibrated to reproduce the initial PERT/CPM plan for the development phase

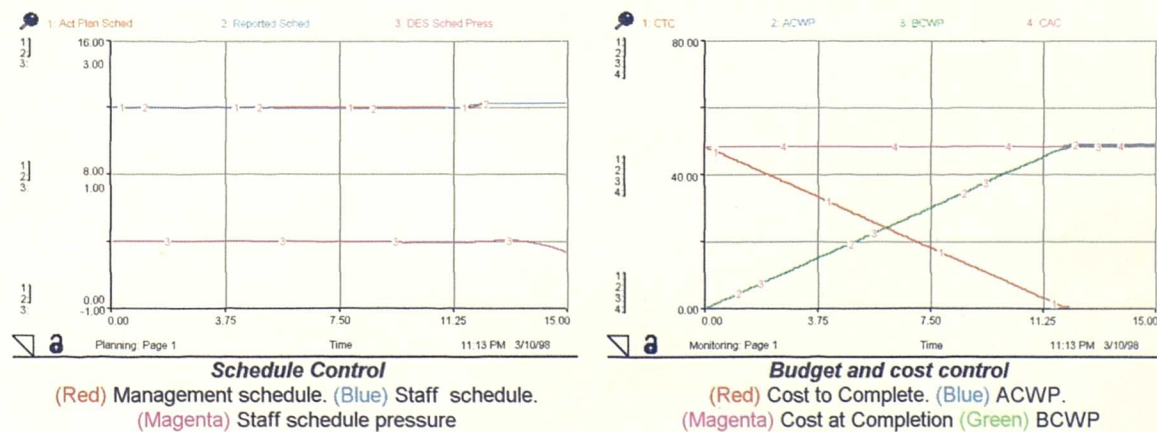


Figure 8.32 – Steady behaviour produced by the SD model when calibrated to reproduce the initial PERT/CPM plan for the full life-cycle

The following SYDPIM activity (P1b) is aimed at analysing the performance of the project plan under various scenarios within the SD model. This where the required scenarios were tested within the SD model. The calibration for the initial PERT/CPM plan, with no involvement of staff from the Client, was considered as the base-case. Two other scenarios were then tested:

- *scenario A* – two BAeSEMA engineers are replaced by inexperienced Client engineers. The total budget and planned schedule are not readjusted.
- *scenario B* – two Client engineers are introduced in the project team, but the plans are changed to compensate the potential negative effects of introducing inexperienced staff. The planned schedule is not changed, but two extra BAeSEMA engineers are added to the team, thus increasing the total budget.

It was known by management that the introduction of Client engineers would have direct impacts on the productivity learning curves of the BAeSEMA engineers. Furthermore, the Client engineers themselves would have their own slower learning curves. These assumptions regarding the *direct* impacts were considered as an input to the model. Tables 8.22 through 8.24 illustrate these assumptions as described by management.

Time (Quarter)	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Total MY
Base Case													
All BAeSEMA													
BAeSEMA 1	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	3.00
BAeSEMA 2	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	3.00
BAeSEMA 3	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	3.00
BAeSEMA 4	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	3.00
Total Effort													12.00

Table 8.22 – Productivity learning curves of BAeSEMA engineers in the base-case

In the base-Case, The four BAeSEMA engineers are assumed to work equally at a best nominal productivity level.

In scenario A, BAeSEMA engineers will divert part of their effort to support Client engineers (referred to as SEC engineers). One BAeSEMA engineer is more experienced in providing training, and one SEC engineer is also more experienced

in receiving the training. The estimated impacts on the productivity learning curves are shown in tale 8.23.

Time (Quarter)	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
Scenario A												
BAeSEMA/SEC Co Devpt												
SEC 1 (Exp)	40%	60%	80%	80%	80%	80%	80%	40%	60%	80%	80%	80%
SEC 2 (Inex)	20%	30%	40%	50%	60%	70%	70%	20%	30%	40%	50%	60%
BAeSEMA 1 (Exp Sup)	60%	75%	90%	90%	90%	90%	90%	60%	75%	90%	90%	90%
BAeSEMA 2 (Inex Sup)	20%	35%	50%	65%	80%	80%	80%	20%	35%	50%	65%	80%

Table 8.23 – Productivity learning curves in scenario A

In the third scenario – i.e. scenario B – two SEC engineers are introduced in the project team, but the plans are changed to compensate the potential negative effects of introducing inexperienced staff. The planned schedule is not changed, but two extra BAeSEMA engineers are added to the team, thus increasing the total budget. This is described in the table 8.24 below.

Time (Quarter)	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
Scenario B												
BAeSEMA/SEC Co Devpt with re-planning												
SEC 1 (Exp)	40%	60%	80%	80%	80%	80%	80%	40%	60%	80%	80%	80%
SEC 2 (Inex)	20%	30%	40%	50%	60%	70%	70%	20%	30%	40%	50%	60%
BAeSEMA 1 (Exp Sup)	60%	75%	90%	90%	90%	90%	90%	60%	75%	90%	90%	90%
BAeSEMA 2 (Inex Sup)	20%	35%	50%	65%	80%	80%	80%	20%	35%	50%	65%	80%
BAeSEMA 3	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
BAeSEMA 4				100%	100%	100%	100%	100%	100%			

Table 8.24 – Productivity learning curves in scenario B

The planning changes in this scenario resulted in new targets being set within the SD model, as described in table 8.25 below.

Planned targets	Full life-cycle	Development	Unit
Estimated size	100 K	100 K	DSI
Schedule	12	7	Quarters
Budget	66	38	Person-Quarter
Staff profile (average)	5.5	5.43	FTS

Table 8.25 – New planned targets in scenario B

Given these scenarios, the main questions to be addressed by the SD analysis were:

- what would be the impacts in of introducing inexperienced engineers (scenario A)? Will the project over-run? By how much?
- would the re-planning changes be effective in coping with the disruptive impacts of introducing the inexperienced Client engineers (scenario B)? What are the remaining negative impacts?

The evaluation of each scenario was based on the following forecasts produced by the SD model:

- the cost and schedule over-runs against the targets (of both development phase and full life-cycle);
- the cost and schedule performance against the base case (of both development phase and full life-cycle);
- the overall quality problems in the full life-cycle: how many defects are generated and needed to be reworked;
- the quality impacts in development: how many defects escape to integration.

The SD model was calibrated to simulate scenarios A and B for the development phase and for the full life-cycle. The results achieved for the development phase are summarised in table 8.26 below.

Development Phase	Base Case	Scenario A	Scenario B
1. Schedule (Quarter)			
1.1 Planned	7.00	7.00	7.00
1.2 Actual	7.05	7.55	7.00
1.3 Slippage	0.05	0.55	0.00
1.4 Slippage (%)	0.7%	7.9%	0.0%
2. Cost (Person-Quarter)			
2.1 Budgeted	28.00	28.00	38.00
2.2 Actual (CAC)	28.20	30.66	37.81
2.3 Over-run	0.20	2.66	-0.19
2.4 Over-run (%)	0.7%	9.5%	-0.5%
3. Quality (Defect/KDSI)			
3.1 Defects Generated	50.44	90.40	72.64
3.2 Defects Detected	28.89	13.05	28.59
3.3 % Defects Detected	57%	14%	39%
3.4 Defects Escaped	23.81	78.43	46.41
3.5 % Defects Escaped	47%	87%	64%
4. Comparison with Base Case			
4.1 Schedule	0.00%	7.09%	-0.71%
4.2 Cost	0.00%	8.72%	34.08%
4.3 Quality	0.00%	229.40%	94.92%

Table 8.26 – Summary of results for the development phase

In the base-case, the model was calibrated with the required metrics so that the planned targets of schedule, cost, and quality, were all achieved within minor deviations. Replacing two BAeSEMA engineers by two Client inexperienced engineers (scenario A) causes cost and schedule over-runs in the order of 7% to 9%, for the development phase, against the base case. The worse effect however, is in the number of defects escaping to integration, which more than triples (a 229% increase). In this scenario, serious problems could emerge in integration.

Adjusting the plan to accommodate the two SEC engineers (scenario) recovers the original schedule but at the expense of a higher budget. The quality problems are considerably reduced, although the number of defects escaping to integration still doubles compared with the base case. The worsening of the schedule and quality results in scenario A, and the recovery in scenario B at the expense of a greater budget, is represented in the graph of figure 8.33 below.

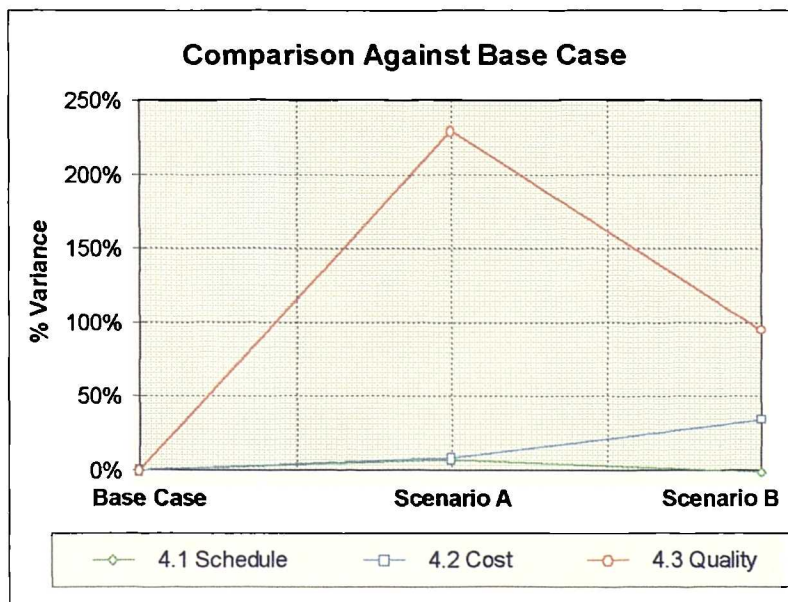


Figure 8.33 – Impacts of scenarios A and B in the development phase

Clearly, while scenario B is more expensive than scenario A, the gains in quality are substantial. Given the potential reduction of serious problems in integration, these quality gains are likely to compensate and hence are worth the cost.

The results achieved for the full life-cycle are summarised in table 8.27.

Full Development Life-Cycle			
	Base Case	Scenario A	Scenario B
1. Schedule (Quarter)			
1.1 Planned	12.00	12.00	12.00
1.2 Actual	12.20	18.60	13.65
1.3 Slippage	0.20	6.60	1.65
1.4 Slippage (%)	1.67%	55.00%	13.75%
2. Cost (Person-Quarter)			
2.1 Budgeted	48.00	48.00	66.00
2.2 Actual (CAC)	48.60	74.20	74.29
2.3 Over-run	0.60	26.20	8.29
2.4 Over-run (%)	1.25%	54.58%	12.56%
3. Defects Removed (Defect/KDSI)			
	79.25	164.46	122.27
4. Comparison with Base Case			
4.1 Schedule	0.00%	52.46%	11.89%
4.2 Cost	0.00%	52.67%	52.86%
4.3 Defects Removed	0.00%	107.52%	54.28%

Table 8.27 – Summary of results for the full life-cycle

Again, for the base-case the model was calibrated with the required metrics so that the planned targets of schedule, cost, and quality, are all achieved within minor deviations. Replacing two BAeSEMA engineers by two SEC inexperienced engineers (scenario A), causes a significant schedule slippage and cost over-runs, both in the order of 53% against the base-case. This is consistent with the previous analysis of the development phase: serious problems in integration stem from a high number of defects escaping from development. While most of the defects are removed in integration (so that system testing is passed), the overall generation rate was twice larger than in the base case (i.e. a 107% increase). This indicates that most of the extra effort is spent on detecting and reworking defects (the actual breakdown between development, QA, and rework activities can be extracted from the model).

Adjusting the plan to accommodate the two SEC engineers does not eliminate the schedule delay in full, with a 13% overall project slippage still remaining (about 4 months and 1 week). The total project cost over-runs in the same order of magnitude, against the adjusted target of 66 person-quarter. These persistent over-runs are due to the inevitable increase in error generation and all the extra rework required (54% increase against the base case).

The increase of schedule and cost along with quality problems in scenario A, and the partial schedule recovery in scenario B at the expense of more budget, is represented in the graph of figure 8.34 below.

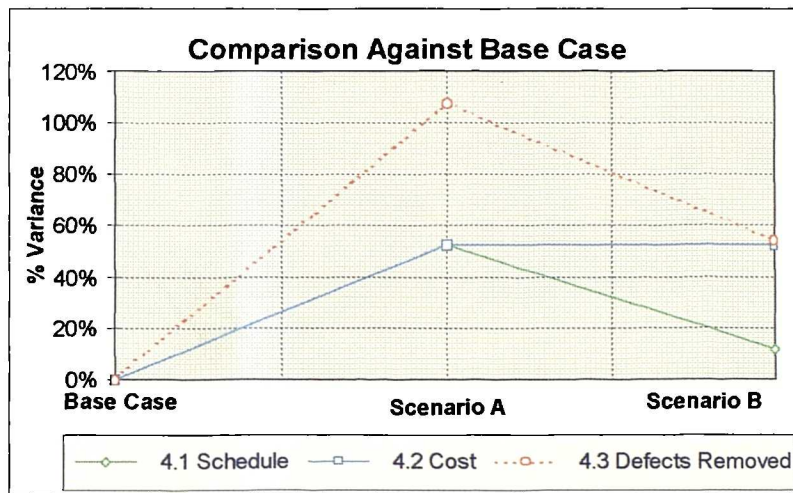


Figure 8.34 – Summary of results for the full life-cycle

The main conclusion from this SD analysis was that introducing the SEC engineers without adjusting the plans (scenario A) is unrealistic and can be dangerous: the final cost will be the same as if the plan had been readjusted (i.e. 74 person-quarter; scenarios A and B), but with a much worse schedule overrun. Overall, the mitigating actions in scenario B do not solve the cost problem, but reduce considerably the potential schedule slippage from 55% down to 12%.

Once the SD analysis is complete in activity (P1b), the next step is to implement the selected decisions in activity (P2b), thereby improving the project plan. The new plan is then converted back to the PERT/CPM model in activity (P3b), and the target metrics are updated in the SYDPIM objects as appropriate in activity (P6). For reasons of confidentiality, the actual decisions selected by management are not identified here and hence the implementation of these SYDPIM activities is not described. Overall, this application of a SYDPIM was a case of risk analysis and mitigation within the planning function. An initial project plan was converted from the PERT/CPM model to the SD model where the risks were assessed and mitigated. Depending on management final decision, a new plan would be produced in the SD model and then would be converted back to the PERT/CPM model.

8.4 Summary conclusions and final discussion

This chapter described some examples of practical implementation of both the SYDPIM Model Development Method and the SYDPIM Project Management Method. These two methods were continuously tested and refined throughout the case-study, in an iterative manner. The examples here presented are illustrative and are not aimed at providing an exhaustive and complete step-by-step implementation of the methods, since this would be extremely lengthy. Instead, different examples focused on different aspects of the SYDPIM methodology.

The SYDPIM Model Development Method was illustrated through the description of the design phase. For the sake of simplicity and clarity, this focused on the design of the prototype model. The SYDPIM steps proposed in chapter 6 were followed and some of the most important formal design elements were described. An overview of the implementation phase was also presented, where the development of the core process structure of the prototype model was described. This description was aimed at illustrating how the formal design elements of the SYDPIM method can be developed and used to support the model implementation phase.

Regarding the SYDPIM Project Management Method, five examples of application were described. They all refer to different scenarios where management explicitly requested the use of SYDPIM as a means to analyse and help solving a specific problem. The first example refers to the diagnosis of the design phase of a specific software component. While there were some cost savings, the completion was delayed. Although this was not a major problem, the SD model was used to explain why this situation occurred. This first application was primarily aimed at demonstrating the usefulness of SYDPIM and at refining the methodology. According to the SD analysis, varying work productivity and cost to rework levels were the main cause for the outcome. The second example of application referred to re-planning and was more important under a management perspective. Management needed to know with some urgency whether the quality of the designs could be improved through the implementation of a formal inspections technique called "Fagan inspections", and in what conditions. The SD analysis showed that significant quality benefits could be achieved if the completion date of the design phase was extended. The third case of application also took place within planning

but refers to a case of risk analysis. Management needed to know what were the impacts of requirements changes being imposed by the Client, towards the end of the design phase. The SD analysis suggested that, in order to mitigate considerable adverse effects on the quality of the designs, a certain level of schedule extension would be required in proportion to the changes requested. The fourth example of application refers to the calibration of the SD model to reproduce the initial plans of the second increment of the C2 SW project. This took place at the end of the case-study, when the SD model was delivered to the project team, as planned in the beginning of the case-study. The last example refers to a re-planning scenario, where management needed to assess the impacts of introducing inexperienced Client staff in the project team. Management also wanted to identify effective mitigating re-planning actions. This took place after the first release of the C2 SW system being installed in the Client, and referred to the development of a second release, which was about to be initiated.

In all of these examples of practical application there were two important issues that highlight the usefulness of the SYDPIM methodology: the SD model provided a distinctive contribution to the project management process, supporting a certain type of analyses, which would be difficult to implement using the traditional tools alone; and secondly, the use of the SD model was formally integrated with the PERT/CPM model, as project plans were transferred between the two models based on the establishment of analytical links. Furthermore, the fact that these applications were explicitly requested by management throughout the project shows that the value of SYDPIM was perceived and recognised by managers within a real life project management context.

Without a formal framework to integrate the use of the SD model within the traditional project management process in place, it would have been difficult to achieve these results by just using a SD project model in isolation. The formal process logic provided by SYDPIM has proven valuable in guiding the use of the SD model, as it identified the sequence of steps to be taken. The formal description of each SYDPIM activity provided a detailed description of the work to be carried out in each step. The formal specification of the SYDPIM objects and of the analytical links clarified all the data storage requirements as well as the data exchange operations between the two models.

The SYDPIM model development provided the required framework to develop in a progressive manner an appropriate model for the C2 SW project. The design elements allowed the model structure to be clearly understood, so that improvements were easier to conceptualise and could be implemented without sacrificing this same understanding.

The most relevant difficulties encountered in implementing SYDPIM throughout the case-study, had to do with the following issues:

- implementation of the SYDPIM objects;
- effort required to specify and maintain the analytical links;
- automation of the data exchange between the two models;
- calibration of SD model to reproduce past behaviour.

As discussed in chapter 7, the effectiveness of the SYDPIM process depends, in great part, on the technological platform chosen to implement the SYDPIM objects and analytical links. In particular, this affects the level of automation for the data exchange between the two models. A purely manual implementation can become time-consuming. At the other extreme scenario, a specialised software tool can support or even implement most of the operations. In this case-study, a spreadsheet tool was used to implement the SYDPIM objects. This provided an acceptable solution, although the information needed to be updated regularly for consistency with the elements of the already existing information system in the project. In some cases, this was a laborious effort-consuming task. The calibration of the SD model to reproduce observed past behaviour did not benefit from any semi-automated algorithm and hence in some cases it was also an effort-consuming task. As also discussed in chapter 7, the author believes that the practicality of SYDPIM can benefit considerably from developments in the software platform that supports the methodology. The development of such platform was not the purpose of this research.

It is important to note that this experience took place within a time-pressured real project management environment. As the SD model was progressively developed and used according to SYDPIM, the methodology itself was refined and improved. Therefore, every application benefited from past experiences and served as the

basis for further readjustments. This resulted in an interesting but challenging research scenario, where a methodology is being tested and improved, while used at the same time to help managers solve real problems. From this experience, it is the author's opinion that, overall, the various practical applications of SYDPIM that took place throughout the case study demonstrated the distinctive value of the methodology.

9. Conclusions

9.1 Overview

Most of the relevant findings drawn from the present research were identified and discussed in detail throughout the previous chapters. In a sense, this thesis is continuously conclusive. In great part, this results from the nature of the research approach adopted. In action research, understanding is progressively built, often in an iterative fashion. The theory being developed is continuously conceptualised and re-organised. The result is the cumulative generation of conclusions.

Presenting and discussing the conclusions drawn from this research only in a final chapter, at the end of this thesis, would not reflect the way in which knowledge was actually developed, and thus it would be “artificial”. Perhaps more importantly, it could prevent these conclusions to be properly related to the underlying achievements in the field. In the author’s view, this would not be an effective means to deliver the understanding gained from this research, nor of its achievements. As an alternative, the logical flow of the thesis was purposefully designed to reflect the evolutionary process of the research, and in most chapters the relevant conclusions drawn at that stage were identified and discussed “locally”. Naturally, some of them were revisited and reviewed in later chapters.

Being a long thesis, it is however more than appropriate to summarise and compile all the main conclusions at this final stage. This is the purpose of this chapter. Since most of these conclusions were previously discussed in other chapters, such discussions will not be repeated here in detail. Instead, this chapter will focus on an overall integrative view.

At this final stage, it is also important to cross-check the conclusions achieved against the initial research aim and objectives. Overall, both were achieved with success. The research aim was proposed in chapter 3 as follows:

Research aim:

To conceptualise the integrated use of System Dynamics modelling within the traditional project management process, formally linked with the PERT/CPM based models. To test this concept in practice, identify possible distinctive benefits, and, if any, further develop a formal methodology capable of guiding implementation in other projects.

The conceptual integration of System Dynamics formally linked with PERT/CPM was successfully achieved, as described in chapter 4. This high-level concept was tested within a real project environment, as also described in chapter 5. Distinctive benefits were identified and hence a detailed formal methodology was developed (SYDPIM), which was described in chapters 7 and 6. Chapter 8 further presents examples of practical applications of SYDPIM, where some of the distinctive benefits are illustrated. In order to successfully accomplish the research aim, a set of main objectives was proposed. These were identified in chapter 3 as follows:

Research objectives:

- *develop a conceptual high-level framework (to apply System Dynamics within the traditional project management process, in an integrated manner with the PERT/CPM based models);*
- *test this conceptual framework in practice (through implementation in a real project);*
- *throughout this practical testing, explore alternative ways of applying System Dynamics and identify the more effective ones so that overall benefits are achieved;*
- *throughout this practical testing, if benefits are identified, readjust and refine the conceptual framework into more detail (in particular the formal links between the SD model and the PERT/CPM models), so that a formal methodology can be developed. After this practical fieldwork, develop a final complete specification of the methodology;*
- *gather evidence that this methodology has potential to provide distinctive benefits, thereby leading to performance improvements in project management.*

The development of a conceptual framework was the first step towards the achievement of the research aim. As just mentioned, this objective was successfully achieved, as described in chapter 4. This framework was exposed to the review and critique of specialised refereed journals and conference audiences, having received good acceptance and very positive feedback. As described in chapter 5, a real world complex project was used to test the framework. Throughout this fieldwork “case-study”, alternative ways of using SD in an integrated manner were tested. As also mentioned in chapter 5, this implied an iterative process wherein the researcher “stepped back” from the project, the framework was refined, reworked and further re-tested. The final SYDPIM Project Management Method, described in detail in chapter 7, reflects the outcome of this process, where the more effective ways of applying SD were selected (see SYDPIM process logic in chapter 7). The final methodology turned out to also include a model development method, which is presented in detail in chapter 6. The experiments carried out throughout the fieldwork provided evidence of the distinctive benefits of SYDPIM. As just mentioned above, examples of practical application are presented in chapter 8, illustrating some of these benefits.

As it will be discussed in this final chapter, the various conclusions derived from this work support the successful achievement of the research aim and of its objectives. Given the wide scope of this research, these important conclusions can be found in different areas or aspects of the work developed. While they are all strongly inter-related, for the sake of clarity they will be here divided into the following main topics: (i) the SYDPIM methodology, (ii) the research approach adopted, (iv) the main difficulties encountered, and (v) the proposed future developments. The SYDPIM methodology is itself very comprehensive and therefore the conclusions in this area will be further divided into three main sub-groups: (a) the conceptual integration of SD within the project management process, (b) the SYDPIM Project Management Method, and (c) the SYDPIM Model Development Method. Here, the conclusions will focus on the usefulness of SYDPIM and mainly on the critical issues of practical implementation.

Before proceeding into these individual areas, it is appropriate to summarise the main overall conclusions of this work. These are as follows:

- (1) *SD distinctive contribution* – System Dynamics has the potential to provide distinctive benefits to the traditional project management process. It can play new roles within this process, and support the traditional techniques in their own roles;
- (2) *benefits of integration* – the benefits of SD can be better achieved by integrating it within the project management process, as a complementary modelling technique of project analysis, rather than as alternative to replace the traditional techniques;
- (3) *benefits of formal integration* – this integration can be implemented formally, and various levels of formality are possible. Increasing formality improves the consistency between the SD model and the other traditional models, thereby enhancing the overall distinctive benefits of SD;
- (4) *benefits of analytical links SD-PERT/CPM* – the formal integration of SD can be based on linking a SD project model with a PERT/CPM model. This has proven a valid, feasible and useful concept. While the two types of models provide different representations of the same project, focusing on different aspects and at different levels of detail, their formal integration can be achieved and effectively implemented through the establishment of analytical links. These links ensure consistency and support the exchange of data and information between both models. Links with the PERT/CPM model are important because this model is at the core of the traditional project control process;
- (5) *the SYDPIM solution and usefulness* – the SYDPIM methodology proposed in this research provides a method to formally integrate the use of SD within the traditional project management process, linking a SD project model with a PERT/CPM model. The implementation of this methodology has proven beneficial in practice, as it improved project management performance in the fieldwork conducted. It is likely that most of these benefits and improvements will be achieved in other projects although this needs further fieldwork;
- (6) *importance of SD model development and SYDPIM solution* – to implement the SYDPIM methodology it is crucial to have a valid SD project model available, representing the specific project. The SYDPIM methodology

- comprises a structured model development method to support this requirement. This method has also proven useful in the fieldwork;
- (7) *SYDPIM main requirements* – the successful practical implementation of SYDPIM is based on some important requirements. Of particular relevance are the level of structuring of the traditional project management process adopted by the organisation (in particular the use of a PERT/CPM model), data collection procedures, management / staff support and commitment, management openness to the pro-active and pragmatic SD perspective of problems, and the availability of appropriate software platforms and supporting tools;
 - (8) *Importance of software tools to implement links SD-PERT/CPM* – the implementation of the analytical links between the SD model and the PERT/CPM models are crucial to ensure that the two models are representing the same project (and thus the same work plan). The “transfer” of the project representation from one model to the other can be a complex task, which can benefit from the availability of software tools capable of automating part of this process (mainly data-transfer and consistency-checking activities). Of particular relevance are the (semi)automated calibration of the SD model to reproduce a PERT/CPM scenario, and the (semi)automated readjustment of the PERT/CPM model to represent a SD project scenario;
 - (9) *importance of gradual SYDPIM implementation* – introducing SYDPIM in a organisation who has never used SD before can be a difficult task. An iterative implementation, as followed in the fieldwork undertaken in this research, is highly recommended as it will provide a progressive approach to solving the difficulties and facing the complex issues which will always emerge;
 - (10) *appropriate research approach* – the “action research” approach adopted in this work has proven very effective in supporting the aim of the research. In the author’s view, the direct involvement in the problem for a considerably long period of time was essential to develop the required in-depth understanding. It is unlikely that any other research approach would have provided this essential achievement;

- (11) *difficulties encountered* – there were some important difficulties encountered throughout this research, which should be addressed in future developments. Of particular relevance was the difficulty in finding an end-point for the fieldwork and subsequent conceptualisation of the methodology, maintaining support and commitment from the project team throughout the fieldwork, preventing disruption to the case-study project while conducting the fieldwork, availability of required data and information during the fieldwork, the need to simultaneously conceptualise and implement a novel methodology while keeping pace with the progress of the case-study project, the lack of familiarisation of the case-study project team with System Dynamics, the lack of software tools to support the implementation of both the analytical links and steps of the SYDPIM process, the limitations of the SD modelling software tools used in controlling the various versions of the SD models and various calibration scenarios, and in supporting the model calibration process;
- (12) *importance of other fieldwork experiences* – the SYDPIM methodology here presented is based on one single fieldwork experience. While it was conceptualised under a generic perspective, it is naturally influenced by the particular issues and experiences that took place within the KDCOM software project. It is highly desirable that further fieldwork experiments are carried out in other projects in order to improve the methodology's actual potential to be generic;
- (13) *relevant future developments* – there are some potential developments which can support and benefit the practical implementation of SYDPIM in other projects. These developments can be accomplished individually and include the development of supporting software tools, refining the analytical links to automate as much as possible the transfer of a project scenario from one model to the other, integration of a SD project model within a PERT/CPM planning tool available in the market, and further refinement of the SYDPIM Model Development Method to ensure that good quality models can be developed in short periods of time (in particular through the re-use of generic sub-structures).

The conclusions (2), (3) and (4) above stress two of the core novelties and research findings of this work: (i) the added value of formally integrating a SD

project management model with the traditional PERT/CPM model, and (ii) while different levels of formality can be adopted, analytical integration is technically feasible. These are important findings because the formal integration had never been attempted before and thus there was no knowledge about whether this would be technically feasible, given the rather different nature of the two quantitative models. The analytical links developed and presented in this work provide this evidence and the technical means to achieve formal integration. While to some extent understanding and implementing these links can be complex, it has been proven that this is possible and of practical value.

SYDPIM integration takes place at two levels: at the process level and at the technical level. The latter is reflected in the establishment of analytical links between the two models, which support the exchange of data and formal consistency. The process level is reflected in the SYDPIM process logic, as described in chapter 7. This process logic articulates the integrated use of the two models, creating a rich and dynamic dialect among them. Perhaps the most fundamental finding and conclusion from this research is that this novel dialect raises a number of questions, problems and issues regarding the management of a project, which otherwise would not be identified and addressed, were the two models employed separately (even if simultaneously) – similar conclusions regarding synergies of “hybrid modelling” have been reported in related literature (e.g. Eden 1994, Ackerman et al 1997). SYDPIM has proven to provide an effective means to establish this valuable dialect, through which critical problems for project performance can be identified *and* effective solutions can be devised. Interestingly, the main causes of current project failure relate precisely to this type of problems. By raising new questions and problems, SYDPIM happens to address these causes, which the traditional approach alone has been unable to cope with.

The formal integration provided by SYDPIM leads to a continuous interleaving of the two types of models throughout project implementation. This interleaving leads to a process where the two models “talk” to one another, exchanging important information, and thereby validating a common project plan and description of its past, under rather different perspectives. The SD model ensures that the project PERT/CPM plan will not suffer from escalating problems, which typically result from

the various and ever present feedback dynamics. It also uncovers important monitoring information about the project past and current status. Both in monitoring and planning, it provides a means to continuously diagnose the project behaviour and thereby understand the causes of project of performance. The resultant continuous learning is the basis to identify better solutions and thus achieve performance improvements. The formal integration of a PERT/CPM model with the SD model is crucial to this process, as it provides the *only* means through which effective strategic solutions devised in the SD model can be validly translated in a consistent manner to an operational plan, capable of leading to the expected results. In monitoring, this formal integration ensures that the SD enhanced vision of the past is also effectively translated to the operational level, thus allowing understanding from performance diagnosis to range from the strategic level down to the operational level. Without this formal integration, an inconsistent relationship between the strategic solutions, lessons learned and insights gained from the SD model, and the operational specifications in the PERT/CPM model, could hardly raise new questions and ensure that strategies are effectively put into practice.

The following sections discuss some particular relevant issues and other conclusions in more detail.

9.2 The SYDPIM methodology

9.2.1 Conceptual integration of SD within the project management process

The persistent overruns reported in chapter 2, along with the new promising features of SD, have led some SD modellers to suggest the replacement of the PERT/CPM and earned value techniques (Cooper 1993). This was a relevant issue for the purpose of this research. Therefore, both the validity and the usefulness of the conceptual integration of SD with the traditional techniques (as opposed to replacing them), were a main concern of this research. It was concluded that this conceptual integration provides distinctive benefits, as SD has potential to play its own roles within the project management process. At the same

time, SD on its own does not address properly the operational issues. Therefore, it should be used as powerful complementary addition to the traditional approach.

Initially, the literature review and analyses of chapter 2 concluded that the traditional approach to project management was focused on the operational issues. On the other hand, the main causes for current failures are of strategic nature and have to do with social human factors. It was further concluded that SD provides the required systemic view to address these issues. It could therefore become a valuable addition to the project management process. This early conclusion based on literature review was confirmed by the fieldwork experience, as described in chapter 5. As a result, the potential distinctive roles of SD, which had been identified in the proposed conceptual framework (see chapter 4, table 4.1), were further refined and included in the formal specification of the SYDPIM Project Management Method (see description of SYDPIM in chapter 7). These roles cannot be performed effectively by the traditional techniques on their own and, conversely, SD is not appropriate to address the operational issues. The benefits from integration with the traditional techniques stem from their complementary nature. The SD distinctive roles result primarily from two main features of a SD model: (i) a wide range of complex scenarios can be conceptualised and tested quickly, and (ii) the model has the ability to explain the underlying causes of the outcome. This turns the SD model into a powerful learning tool. In addition, formal integration provides the benefit of supporting the consistency and the exchange of data and information between the two types of models (an important requirement for the integrated use of SD within the project management process). An effective way of achieving this formality is through the establishment of analytical links between the SD and the PERT/CPM model. This research has shown that this is possible, and such links were therefore developed and specified as part of the SYDPIM integrated methodology – the “theory” underlying the analytical links is presented in chapter 7 and appendix D.

There can be some difficulties in implementing this conceptual integration. As mentioned before, the more important ones relate to the effort required to implement the analytical links without the support of appropriate software tools (e.g. extracting the project past behaviour patterns from a series of historical project

plans stored in the PERT/CPM model). Furthermore, the implementation of all the analytical links cannot be fully automated. A semi-automated process is therefore required, where human judgement is a crucial input.

Integrating the use of a SD model within the traditional project management process leads to a new process, where new steps need to be considered. The SYDPIM process logic provides a formal and objective description of this new process, which is aimed at guiding the use of the SD model in the field. An important requirement of the SYDPIM process logic, is the use of a PERT/CPM model within the existing project management process. Nevertheless, SYDPIM provides some flexibility regarding this, by considering alternative paths within the process logic (see figures 7.6 and 7.7).

Overall, the SD and the PERT/CPM model provide a different view of a same project. They consider a different range of factors and at different levels of detail. Nevertheless, their integration down to the quantitative formal level is a valid concept, which *can be* implemented. This has been implemented in this research and, as concluded in chapters 5 and 8, it has proven beneficial by contributing to the improvement of project management performance.

9.2.2 The SYDPIM Project Management Method

The SYDPIM Project Management Method is the main element of the SYDPIM methodology. It was developed throughout the fieldwork, based on the continuous feedback received from the tentative applications of SD in an integrated manner. This SYDPIM method was specified in great detail in chapter 7, and also in appendices C and D. It was presented as a formal model, comprising four main inter-related elements: objects, process logic, analytical links, and activities. The method is aimed at articulating the use of the SD model within the project management process, formally linked with the PERT/CPM model. The process logic specifies in detail the flow of management activities within the two main functions of project control: planning and monitoring. The objects provide a formal specification of the SD project model, the PERT/CPM model, of a metrics database (SYMDB), and of other elements. The analytical links establish a formal

relationship between the SD project model and the PERT/CPM model. There are various important conclusions about this core element of the SYDPIM methodology.

In the first place, the SYDPIM Project Management Method does not require a major re-engineering of the traditional project management process. This is an important issue regarding the practical implementation of SYDPIM, which thus becomes easier. The recommended gradual implementation of SYDPIM also becomes easier to achieve. A radical break-away with the traditional approach was not the perspective adopted from the beginning of the fieldwork. This is because it was clear from the early literature review that the SD approach was not capable of addressing all the needs and issues of the project management process.

The SYDPIM Project Management Method introduces some novel elements into the project control process. It changes the process logic of both planning and monitoring functions, by adding new steps and making them more elaborated. It requires the development and maintenance of specific objects, in particular of the SD model itself. It also requires the development, maintenance and use of the analytical links established between the SD model and the PERT/CPM model.

An important characteristic of the SYDPIM Project Management Method is that it can be implemented at different levels of formality. Once more, a gradual implementation within the organisation becomes easier. This type of implementation is based on a gradual change from the conventional framework to the SYDPIM framework. Most likely, the vast majority of the potential SYDPIM users is not prepared for a full implementation of SYDPIM. For this reason, a "basic mode" for implementing SYDPIM was proposed in chapter 7. This "basic mode" scenario considers a standard use of the PERT/CPM model, and does not require a complex SD project model.

There are some important requirements so that an effective implementation of the SYDPIM Project Management Method can be achieved. In the first place, an appropriate SD project model must be available. The validity of this model is crucial. This requirement is handled by the SYDPIM Model Development Method,

another element of the SYDPIM methodology, which was specified in detail in chapter 6. Conclusions about this method are presented in the following section. Another important requirement, also related to model development, is the need of some basic SD expertise. Throughout the project, someone will have to have the required know-how to maintain the SD model and use it to run analyses. While an extremely skilful and experienced SD expert is not required, this is not the type of expertise which can be developed quickly, as with the PERT/CPM technique.

The use of the SD model also requires that a considerable amount of data is available in a timely fashion. Sometimes, expert judgement can be used as an alternative to data collection, but in that case careful attention is required to validate possible conflicting views (e.g. using the Delphi method). Throughout the project, the SD model will have to be re-calibrated many times. This continuous exercise of model calibration is effort-consuming. It is therefore important that this is explicitly accounted for in the budget of the management team. Poor and thus "invalid" calibrations carried out under time-pressured conditions will probably lead to unhelpful or even counter-productive analyses. The ideal solution to the calibration problem would be an automated process, which would ensure both valid and quick calibrations. This "technology" is not available (and perhaps conceptually it is not desirable), and therefore human judgement is required. Nevertheless, semi-automated algorithms can be used to support the process. Similarly to SD calibration, the PERT/CPM model will have to be readjusted many times to reflect different project scenarios developed in the SD model. This can also be a laborious process. The required dis-aggregation of information leads to various possible solutions, and therefore human judgement is again required. Some automation of this process, through the use of specialised software tools, would be valuable.

Overall, the SYDIPM project management method is more suited to environments wherein the traditional project management process is implemented in a basic structured manner. The SYDPIM Project Management Method is grounded on some basic managerial discipline. The more structured the existing project management process, the easier it will be to implement SYDPIM. An important requirement of such structured process is the use of a PERT/CPM model.

Preferably, historical past versions of the project plan should be recorded, and a future work plan should also be available in this model. The PERT/CPM model should also be updated regularly regarding schedules, effort and resources information. Another important feature of a structured project management environment is the implementation of metrics programs. These will handle the effort-consuming task of data collection, and will often provide much of the data required by SYDPIM.

The explicit support from management and staff to the required process and cultural changes imposed by SYDPIM is also an important requirement. The SD perspective of project management is pragmatic and realistic (e.g. estimating undetected defects). This might not be easy to assimilate within a time-pressured environment – realistic news are bad news (the so-called “killing the messenger” syndrome).

Finally, the software platforms and tools used to implement the SYDPIM objects, and to support the overall consistency-checking and exchange of data among them, is also critical. The implementation of the SYDPIM metrics database (SYMDB) is important for the effectiveness of the SYDPIM process. An appropriate software tool is required to implement this database. The manual implementation of the analytical links is very effort-consuming, because most of the data-exchange and consistency-checking requirements cannot be implemented with some level of automation. The technological platforms and software tools are therefore a major factor for the timeliness of SYDPIM, which in practice is imperative to achieve. As discussed in chapter 7, the ideal technological platform for the SYDPIM Project Management Method would be a fully integrated application, supporting the implementation of all the SYDPIM objects and activities. Because such application is non-existent, the alternative is a mix of manual processes and using existing software tools like spreadsheets.

9.2.3 The SYDPIM Model Development Method

SYDPIM requires that an appropriate and valid SD project model is available. In chapter 6, a review of the more relevant past developments showed that there was no standard method available to develop SD project models. A structured

development method was therefore conceptualised throughout the fieldwork, and became an integral part of the SYDPIM methodology. The SYDPIM Model Development Method has four main novel features: (1) an explicit design phase, which comprises the specification of various formal design elements, prior to model quantification; (2) a generic project feedback structure, which is used to guide the overall modelling process; (3) a structured validation framework, comprising various stages, and which is integrated into the implementation process; and (4) an explicit overall life-cycle process, comprising a sequence of well defined phases and stages. There are some important conclusions drawn from this research regarding its practical implementation.

First, this method was proposed to support the development of a project model within the context of implementing SYDPIM. Its primary aim is therefore to ensure that the project model developed will comply with all the SYDPIM requirements. However, this method can also be applied to develop a project model to be used for another purpose (e.g. post mortem analysis of a past project).

The main potential difficulty in implementing the SYDPIM model development method has to do with the effort required to implement the design phase. Various formal design elements need to be developed, and then updated throughout the implementation phase. Furthermore, these formal design elements must always be consistent one another. Because they are highly inter-related, this can be a laborious process. In order to cope with this difficulty, the availability of an appropriate software tool is important. The ideal would be a SD modelling tool being capable of supporting the implementation of the design phase in an integrated manner with implementation and validation. To the author's knowledge, such tool does not exist.

The quality of the SD project model is crucial to the usefulness of SYDPIM. While a careful implementation of the SYDPIM model development is likely to ensure the required quality, this can be an effort-consuming development. An explicit trade-off between effort and quality is required. In this context, it is important to note that the disadvantages of low quality can be disastrous. The effort required to implement the SYDPIM Model Development Method must therefore be explicitly

accounted for in the management budget. In case time and effort are scarce resources, it is recommended that the scope of the model is adjusted to accommodate a proper implementation of the SYDPIM Model Development Method. Prototyping and incremental development can also be appropriate solutions when a model needs to be made available in a short period of time.

Overall, throughout the field work the SYDPIM Model Development Method has proven useful in guiding and supporting the continuous activity of model development and refinement.

9.3 The research approach adopted

A discussion of the research approach adopted in this work was presented in some detail in chapter 3. This included the elements taken from the positivist and phenomenological philosophies, the reasons why such an approach was appropriate, its potential limitations, validity issues and the research methods employed. The approach followed was predominantly phenomenological in the form of action research. Given the aim of the present research, this approach was aimed at taking advantage of the strengths of phenomenology, preventing the weaknesses of positivism, and complying with practical constraints. There are some important conclusions regarding the implementation of this research approach.

There were three main aspects of action research which posed important challenges to the researcher. They have to do with the fieldwork conducted, and are as follows: (1) the fieldwork started with a poor understanding about the perceived problem (as it should). In general, this makes it difficult to specify a clear scope for the fieldwork, and thus to lay out an effective work plan; (2) throughout the fieldwork, it was difficult to identify an end-point to the research. By the nature of the approach, the researcher must seek a deep involvement in the problem. However, "living the problem" tends to shift the concern from developing a theory to solving the actual problems. In reality, problems in management systems are on-going events, which if ever completed require a long period of time. Most research projects cannot afford such a time-frame. Therefore the researcher will

have to abandon the problem half-way. This may create the feeling that an "incomplete theory" was developed; (3) success is often associated with completeness. In action research, this concept can be misleading. It is important to note that understanding is the primary value of action research. If a valid understanding about the problem is developed, in a way so that it can be transmitted to others and help them solving their own similar problems, the research can be considered as successful. It is therefore important that the researcher makes an effort to develop a good perception of the understanding being gained through the fieldwork, as well as an awareness of when this has become sufficient within the time and budget availability for the project, so that it can be compiled into a final theory.

In the present research, the problem of the fieldwork scope was in part attenuated by the development of an up-front conceptual framework, which has set clear boundaries for the research scope. However, the appropriate level of detail to be considered within these boundaries could not be identified as that stage. There was a clear scope growth in this aspect. The lack of a clear end-point for the fieldwork was also an issue, but there were two mitigating factors: the limited time and budget available for the research project, and the deadlines agreed with the organisation. The difficulty in finding an objective sign of success was motivated by the scope growth in the level of detail, and by the intrinsic complexity of the problem. The project plans were complex, the project team was large, large amounts of data and information were constantly changing. Finding self-contained, stable and updated scenarios to test the conceptualised ideas was not easy. However, the actual accomplishment of successful applications of SD in an integrated manner helped to overcome this problem. Ensuring that such events take place in action research, where the developed understanding is put at work to help managers, has proven to be an important strategy.

Regarding the methods employed, phenomenology suggests that structured qualitative methods of analysis should preferably be adopted, because they will support the validity of the results. However, it was experienced that such methods can also have the disadvantage of creating the impression that individuals within the target organisation are being "evaluated", or that their points of view are being

recorded for “unclear” purposes. More informal means of accessing data and opinions, and of involving individuals in the research, have proven more useful.

A critical issue in conducting fieldwork is the continuous involvement and support from the organisation. There are always political issues involved. Managing these issues without threatening the scientific nature of the work being developed is not easy. The understanding developed may counter existing beliefs and personal interests. Action research should not adopt the profile of a pure professional consulting service. Nevertheless, some useful analogies can be considered. The primary one are perhaps the ways of keeping the sponsor satisfied, while making the right alliances within the organisation. However, in action research valid understanding is not a means to an end (as in consulting), but rather it is the end itself. The action research project should therefore promise that the understanding to be developed will help the organisation, but its primary aim should not be stated as a solution to the problem. In the present research, the support from the organisation was promised and maintained throughout the fieldwork, despite the various difficulties always encountered (e.g. in having access to data and individual opinions). The awareness that the fieldwork was a tentative development of understanding about the problem of interest, was important.

In action research, a large amount of time is spent by the researcher within the organisation where fieldwork takes place. As understanding is developed and ideas are conceptualised, the individuals within the organisation are an essential audience regarding the practical issues. But their contribution regarding the conceptual and scientific aspects of the research is very limited. Feedback under a scientific and conceptual perspective is essential regarding two main aspects: the way in which the research is being conducted (i.e. can it lead to valid knowledge?), and the generalisation of the theory being developed (i.e. are issues not specific to the organisation being properly addressed?). In this respect, it was the author's experience that it is crucial to expose the conceptual understanding being developed to a scientific audience, *while* the fieldwork is underway. As reported in chapter 3, in this research the fieldwork was exposed to scientific audiences in several ways.

Finally, the validity of generalising the theory developed in action research is a crucial issue. This was discussed in some detail in chapter 3, and also briefly addressed in chapter 5. The strength of action research is the in-depth of the fieldwork experience. The potential weakness is the limited number of cases, which cannot ensure the statistical significance of the “generalability” of conclusions. There is always the inevitable feasibility constraint that no research project will be able to afford a large sample of fieldwork cases (say a hundred cases). It is the author’s view that the route to develop an objective criteria of validity is the establishment of “quality practices”, which must be followed in “action research” projects (something within the spirit of quality standards and certification). The generalisation of the theory developed in this work is supported by various aspects, which were discussed in chapter 3. While this generalisation cannot be objectively demonstrated in a positivist manner, its validity rests on the demonstrated usefulness and on the ability of being clearly explained to others (Eden and Huxham 1997). Pursuing further fieldwork cases is the proposed way ahead. Not only further experiences will explore the validity and generalisation of the theory here proposed, as they will also produce new findings.

Overall, it is the author’s opinion that not only action research has proven to develop a deep understanding about the integration of SD in project management, as no other alternative research approach would have provided this same level of understanding. If questionnaires, interviews, and data analysis, were all carried out at distance from what happens within a project, they could have produced interesting statistics on what managers thought about what they poorly understood. That would not have allowed the researcher to develop a field-grounded comprehensive theory, as the one here presented.

9.4 Main difficulties encountered

No research is fully complete. Future developments are always the way ahead. Under this evolutionary perspective, a crucial value of any research project is the identification of difficulties, so that future developments can concentrate on new aspects of the problem. There were various difficulties encountered throughout the

present research. As with the conclusions, most of these were identified and discussed through this thesis. They can be summarised as follows:

- (1) *scope of the research* – it was not possible to anticipate the extent to which the integration of SD within the project management process (as described by the conceptual framework), would grow a complex problem, requiring many issues to be addressed in a considerable level of detail. This has proven a major difficulty. It required the fieldwork to be extended considerably, and it has left the researcher with a set of complex issues to be solved, which required much more effort than expected. This difficulty was reinforced by the fact that this was a novel endeavour, which thus could not benefit from past developments. The level of detail was the main factor of scope growth. It would not be appropriate to develop a high-level solution (such as the conceptual framework), if at the detailed level the core concept was not valid. The SYDPIM methodology can be implemented at different levels of formality and detail, and a gradual implementation is even suggested. However, organisations can feel comfortable in doing so because the whole concept was validated at the detail level. With an already wide scope (the reasons for which were discussed in chapter 3), the researcher felt the need to deepen the work into considerable detail. Given the nature of the research aim, this was probably inevitable. However, it has proven a complex and effort-consuming endeavour;
- (2) *data collection during the fieldwork* – as discussed in previous chapters, the use of SD requires that a considerable amount of “SD-oriented” data is available. Again, because this was a first time experience, it was difficult to anticipate all the data requirements at the beginning of the fieldwork. Most organisations do not collect the type of data required by the SD model. Collecting this new data for the first time in an organisation is not an easy task. It requires time and effort from the project team, who will not be familiarised with the data anyway. The researcher has encountered some difficulties in this area. It is recommended for future developments that, as much as possible, the data required is specified up-front and that advantage is taken from possibly existing automated mechanisms of data collection;
- (3) *social aspects of fieldwork* – another difficulty of the fieldwork had to do with the social aspects. Not all project team members had the same view and expectations about the research being conducted. The need to please

everyone without disrupting the project, and at the same time investigating what was happening in the project, was a difficult task. It is essential to identify the key individuals and ensure their support as much as possible. Another difficulty related to social issues is the possible “isolation” of the researcher, as the only person who achieves a certain level of in-depth understanding about the problem and theory being conceptualised. While it is important to exchange ideas with others, the researcher might not find an audience;

- (4) *lack of supporting software tools* – a major difficulty had to do with the unavailability of supporting software tools. Again, this resulted in great part from the fact that this was a novel endeavour. Integrating the SD model with the PERT/CPM model requires an extensive handling of data and of various versions of both SD and PERT/CPM models. Implementing this task manually has proven a very effort-consuming and error-prone task. A major problem was that the SD modelling tool used did not support any configuration control of model versions and simulation scenarios, nor the SD calibration process. Other software tools which could have provided critical support would have been “interfaces” to exchange data between the models and check their consistency (this was discussed in chapter 7 in some detail; see figure 7.30);
- (5) *lack of past similar experiences* – there was no previously reported experiences on integrating SD within the project management process. This was not so much of a difficulty in itself, but it was a “catalyst” to other difficulties. There was no reference work to compare progress, there were no proven concepts to support the theory being developed, and there were no relevant past experiences to seek recommendations.

While these were the main general difficulties encountered, none of them has proven fatal to the aim of the research. They were all eventually handled. However, they identify several general issues that are important to address in future fieldwork developments. They also suggest useful topics for such developments.

9.5 Proposed future developments

As most research projects, the work here presented answered some questions and, in doing so, it raised many others. The theory here presented, in the form of the SYDPIM methodology, is not a final and complete piece of work. The integration of System Dynamics within the project management process has many challenges ahead that need to be faced.

The SYDPIM methodology was the first endeavour in this direction. It provide a practical basis of work: organisations interested in applying System Dynamics within their project management process can start by implementing the concepts and ideas proposed by SYDPIM. As previously suggested, these organisations should try to follow a gradual process, increasingly incorporating more SYDIM elements into their project management process. Until they implement SYDPIM at its full extent, there are several steps that need to be taken and important issues to learn about. SYDPIM is aimed at providing a practical working basis from where organisation can start using System Dynamics with the confidence that they are following the right direction.

Until System Dynamics becomes widely adopted at its full potential within the project management community, there is along way ahead. SYDPIM was designed to take full advantage of most of the potential benefits of SD. A future scenario where SD is temporary employed by visiting external consultants, is not the vision of SYDPIM. Nor it is a limited use of SD models, in an isolated manner from the project management process, well below its full potential. The reference to System Dynamics in the PMBOK (PMI 1996), while it is some good news, it portrays such a scenario.

For System Dynamics to become widely adopted at its full potential in project management, there are many aspects that need to be addressed. Of particular relevance is dissemination. System Dynamics needs to be much better known within the project management community. However, organisations which gain access to this knowledge and decide to apply System Dynamics will find some difficulties. The first one will most likely be to understand the role of System

Dynamics and its position within their project management process. SYDPIM provides a clear answer to this problem.

Organisations who decide to follow the “SYDPM way”, will soon be faced with a number of relevant questions. Those will probably be very similar to the ones that the author raised himself by the end of the fieldwork. Answering these questions requires further research developments, either in pure or applied forms. Most of these development result from using SYDIPM as a basis, and from there focus on particular issues in more detail. As mentioned in chapter 3, it was important to consider in SYDPIM a wide rage of project management issues, so that it could emerge as a global and consistent initial solution, upon which focused developments could take place safely.

The first future development here proposed is the implementation of similar fieldwork experiences using SYDPIM. The general aim is to verify whether its basic principles are valid and applicable to other projects, industries, and organisations. The outcome of such type of developments would be a more “genericised” and robust SYDPIM methodology. For example, it may be that other relevant paths need to be considered within the process logic proposed, to cope with circumstances not experienced in the fieldwork of this research. Similarly, it may well be that other relevant data-fields need to be considered in the specification of the models. The overall aim of this type of developments is to improve and validate SYDPIM, as a global generic solution for the formal integration of System Dynamics within the project management process.

The other developments here proposed focus on particular aspects of SYDPIM. These can be very relevant for the practical implementation of other fieldwork experiences, regarding time and effort required. They refer to the SYDPM model development method and to the project management method.

Regarding the SYDPIM Model Development Method, its primary aim is to support the development of valid project models in a controlled and timely fashion. Quality, in the form of validity, is the primary objective, and time is the primary constraint. As previously noted, the major limitation of the method here proposed is the effort

required to develop and document all the formal design elements, prior to model quantification, and then keeping them consistently updated throughout the implementation phase. If the design phase is implemented manually at its full extent, this can be very laborious, error-prone and effort consuming. The obvious future development to overcome this problem takes place in the arena of the software tools. It would be most valuable to develop software tools capable of supporting the implementation of the SYDPIM Model Development Method. The most important features of such tool are as follows:

- *support the implementation of the design phase* – this includes the development of the formal design elements, updating them and checking their consistency;
- *support the implementation of the validation framework* – this includes the generation of some test-cases based on the formal design elements, maintaining a historical record of test runs, identification of types of tests remaining, and the identification of required or advisable tests throughout the whole implementation phase;
- *integration of model design and implementation* – this includes checking the consistency between the model quantification and the model formal design;
- *identification and classification of feedback loops* – the SYDPIM Model Development Method proposes a generic classification framework regarding the feedback loops to be captured in the model (see figure 6.4 in chapter 6). This feature would include the automated identification and classification of feedback loops captured in the model structure. This can be considered as a particular case of the previous feature;
- *support the development of ID's* – during the design phase, various influence diagrams are developed. This feature would support the development and maintenance of these diagrams, in an overall consistent hierarchical structure. This can be considered as part of the feature “implementation of the design phase” mentioned above.

These features could be incorporated into an existing SD modelling tool. Another important aspect of the SYDPIM Model Development Method is its focus on the re-use of “generic structures” (see chapter 6). Another important future development in this area is the creation of a database of such structures, which could be easily

and quickly re-used in any project model (note that while some work has been done in this area, and some practitioners claim doing so, there is no scientific evidence of such available in the literature, nor any framework available to the public domain). Finally, another important future development could concentrate on improving the overall life-cycle of the SYDPIM Model Development Method (e.g. creating new stages and additional design elements). This would probably require some further fieldwork.

Regarding the SYDPIM Project Management Method, the future developments here proposed fall in two main areas: software tools and theoretical / conceptual developments. As with the model development method, software tools play a crucial role on ensuring the timeliness of the overall process, and on preventing errors and thereby promoting quality. This topic was discussed in some detail in chapter 7 (see section 7.5). Overall the ideal would be an integrated tool capable of supporting the implementation of both the SD model and the PERT/CPM model, as well as all the data-transfer, structural readjustment and consistency checking operations. These features can be summarised as follows:

- *implementation of the models* – the tool would support the simultaneous implementation of both models. This would be important in order to implement the analytical links. It also delivers an integrated view of the project, under both PERT/CPM and SD perspectives (similar developments have led to the integration of empirical estimating tools with PERT/CPM tools);
- *implementation of the analytical links* – this would include both the specification of the links and the implementation of the SYDPIM activities that use the links. For example, specifying which PERT/CPM tasks are mapped to the SD-Tasks (a structural link), and the transfer of the plan from one model to the other (a data-transfer activity). As specified in chapter 7, the analytical links include data-links, structural links and data-structural links. Regarding the activities that use these links, five types of elementary operations were also proposed in chapter 7: data exchange, data consistency checking, structural consistency checking, structural-data consistency checking, and structural readjustment. Even for those operations which cannot be fully automated and thus require human judgement, their implementation using a software tool will be quicker and much less error-prone;

- *configuration control of model versions and project scenarios* – another very important feature of a SYDPIM software tool would be to support an effective configuration control of the models' versions and project scenarios. Standard PERT/CPM tools in general support the configuration control of various project plans, as well as some scenario management. The same feature in the SD model is also important, in particular the control of various calibration scenarios. With the two models integrated, each possibly holding different versions and project scenarios, this configuration control becomes both more complex and more important (e.g. which scenario in a certain version of the model corresponds to what scenario in what version of the other model).

The development of such software tool could follow an incremental development process, where these features would be progressively incorporated into the product.

Regarding the theoretical / conceptual developments, they are important to overcome the practical difficulties of implementing SYDPIM and to improve the overall methodology. The more important ones are as follows:

- *conceptualisation of further analytical links* – the analytical links of SYDPIM basic model proposed in chapter 7 were supported by detailed rationales. However, some questions raised remain opened. A critical one is to what extent can the conversion of a project scenario from one model to the other, become more automated than proposed by SYDPIM basic mode. This requires further research which can benefit largely from the development of software tools proposed above. This is because the manual transfer of scenarios from one model to the other can be very laborious;
- *automated calibration of the SD project model* – the automated calibration of SD models is a general unsolved issue in SD (the current state-of-art in SD appears to be the use of automated algorithms to “perfect” initial solutions developed by human judgement). Its practical benefits are obvious, but there are also conceptual objections. The future development here proposed is based on the idea that, since SD project models have common characteristics (specially if they were developed using the SYDPIM Model Development Method), some level of automation is possible, provided the solutions would be

checked by human judgement. This development is particularly relevant to support the transfer of project scenarios from the PERT/CPM model to the SD model;

- *automated readjustment of the PERT/CPM model* – just like it is useful to have the SD model automatically readjusted to reflect a given PERT/CPM scenario, the reverse is also true. However, the transfer of a scenario from the SD model to the PERT/CPM level raises the critical issue of dis-aggregation without human judgement being unfeasible (this was discussed in some detail in chapter 7). However, some level of automated readjustment could be explored. For example, based on the idea of common “re-planning patterns”, which could lead to the identification of generic “templates” for structural readjustment (visual network aspects could possibly be taken into account). This is an interesting theoretical development that requires in-depth research;
- *improvement of PERT/CPM model towards the SD perspective* – it was concluded in the early stages of this research that the PERT/CPM and the SD models address different types of factors. However, the integration of the PERT/CPM model with the SD model would benefit if the two models would speak a “closer language”, by incorporating more common factors. In this way, the standard PERT/CPM model could be improved to consider SD factors like “work quality” (note: a preliminary development in this direction was undertaken by the author, and can be found in Rodrigues 1994b). This type of development will probably lead to the conceptualisation of further analytical links, and therefore should be related to the first topic mentioned above;
- *conceptualise SD structures oriented towards the PERT/CPM perspective* – another route to close the gap between the SD and the PERT/CPM model is to introduce PERT/CPM type of concepts into the SD model. For example, variables like “degree of concurrency” or “average total float” could be incorporated in the SD model. Again, this would probably lead to the conceptualisation of further analytical links.

Each of the theoretical developments identified above provides sufficient scope for individual research efforts. Some of them can be even split into sub-topics. Some may require further fieldwork experiments, while others can be accomplished on a more purely theoretical ground.

In the author's view, the many future developments suggested above reflect the complexity, conceptual and practical richness of the research topic that was the basis for the development of the SYDPIM methodology: the integration of System Dynamics within the traditional project management process. The author will continue to carry out research in this exciting and challenging topic, while pursuing further opportunities for fieldwork and practical applications of SYDPIM .