University of Strathclyde
Department of Electronic & Electrical Engineering

Combining Knowledge Based Systems and Machine Learning for Turbine Generator Condition Monitoring

by

Martin Todd

A thesis presented in fulfillment of the requirements for the degree of Doctor of Philosophy

2009

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

Acknowledgements

I would like to thank all of the staff and students within the Institute for Energy and Environment and in particular all of the members of the Intelligent Systems Team. There are a multitude of people who played a role in helping me develop my research throughout my 3 years at Strathclyde. At the risk of using up half the pages of this thesis naming each of you individually, or worse still leaving someone out, I will rely on the fact that you all know who you are. Thanks for the support.

There are however a couple of individuals I'm sure I will be forgiven for mentioning specifically. I would like to thank: Gordon Jahn for many hours spent helping me with my programming woes; Dr Graeme West for supporting me throughout my research but in particular during my first 6 months when I needed it most; Euan Davidson for spending many hours passing on his vast knowledge of Artificial Intelligence and in the process being instrumental in developing my own research; Robert Currie for being a great friend, both inside and outside of work, and being a motivation to put in the extra hours required. Lastly but not least I would like to say a special thanks to my supervisor Prof. Stephen McArthur who pushed me every step of the way and instilled the confidence within me required to get to this final stage.

Thank you to John Bettles for the work he undertook for my system's graphical user interface for his 4th year project. Thank you to British Energy for sponsoring my project and providing me with the resources required to complete my research. Thank you to the British Energy staff Alan Wattling and the late Duncan Williams who made time available for knowledge elicitation interviews. A special thanks to Sally Shaw who took time out from her heavy workload on many occasions to develop my understanding of the application and most importantly showed enthusiasm and support for my research.

I would like to thank my close friends McInnes, Campbell, Connell, Fitzy, Mike and my brother Chris. Whether it be through watching football, clubbing, DJ'ing or just having a chat over a pint, your friendship has enabled me to forget about the stress of research at the times when I had to get away from it all. Your support and encouragement over the years has also given me the ambition and determination to get to the end of what has been the biggest challenge of my life.

I would like to thank all of my family who have been a source of inspiration to me. In particular I would like to thank my Mum and Dad who have supported me emotionally and financially throughout my whole time at university. It is this unquestioning support which has allowed me to pursue my academic ambitions where many would not have been given the opportunity. I dedicate this thesis and all the work that has went with it to you both.

Finally I would like to thank my girlfriend Morag. It can't have been easy having to live with someone doing a Ph.D. for all of these years but you have never once complained. Now we will be able to plan our free time safe in the knowledge that I won't have to fit in a couple of hours work on my thesis. Thank you for all your love, support and encouragement throughout.

Contents

Abstract

This thesis reports on the design and development of a prototype condition monitoring system. The prototype system was developed for British Energy to assist the Rotating Plant and Dynamics Team in assessing the routine alarms triggered by their on-line condition monitoring system which continually monitor their turbine generators. The prototype comprises of two distinct modules. The first module is a Rule-Based Expert System which assesses the routine alarms using knowledge captured from the condition monitoring experts within the Rotating Plant and Dynamics Team. A Rule-Based Expert System approach was chosen so that there was a clear and transparent explanation provided with each assessment which allows the expert user to verify the result through following the assessment rationale. The second module is a learning assistant which was developed to assist the experts and knowledge elicitation engineer in capturing the explicit rule based knowledge used by a Rule-Based Expert Systems. This module uses a novel adapted version of the Machine Learning (ML) approach, Explanation Based Generalisation (EBG), to help derive knowledge from single training example and background causal behavioural knowledge of the turbine generator. This thesis outlines the rationale behind the selection of these approaches for the prototype system developed through a review of both Artificial Intelligence (AI) and ML approaches. A detailed description of the design approach and system architecture is given for both modules and a comprehensive review of the performance of both modules based on the results of system testing on genuine case study data is presented.

List of Figures

List of Tables

Chapter 1

1 Introduction

1.1 Introduction to Research

Condition monitoring approaches are being employed more regularly within industrial applications to assist in asset management of strategically important equipment [Awadallah & Morcos, 2003], [Han & Song, 2003a] and [Cauvin et al, 1998]. The need for improved reliability and plant lifetime extension is particularly prevalent in sectors which are faced with the challenge of ageing infrastructure. One area in particular, which is heavily reliant on an ageing infrastructure largely built in the 1960's, is the electrical utility and generation sector. Strategic assets such as transformers, generators, nuclear reactors, circuit breakers and switchboards are approaching or have already passed their expected design lifetime. With a growing energy demand and reduced personnel to operate and maintain such systems, companies are exploring more and more approaches to effectively and efficiently manage their assets. One company which has such aspirations and is actively exploring approaches to more effectively manage their most strategically important assets is British Energy.

Significant emphasis is placed on the condition monitoring of turbine generators at each of British Energy's nuclear power stations within the UK. A core team of condition monitoring experts continually assess the behaviour of each turbine generator. This assessment is achieved by analysing key signals captured from transducers on the turbine generator sets which when correctly interpreted can indicate how the equipment is behaving. Beran monitoring systems as described in section 2.2 have been installed in each station to capture and feedback the raw transducer data to the condition monitoring experts for inspection. Typical signals captured by the Beran system from the transducers are operational parameters such as load, rotor current, temperature etc along with eccentricity and vibration measurements. Alarm limits are set within the Beran system by the condition monitoring experts to continually monitor important signals which can indicate a change in equipment behaviour. The purpose of the alarms are to alert staff to a potential behavioural change on the turbine generator which may constitute a problem which requires further investigation. Therefore, it is the job of the condition monitoring expert to diagnose the cause of the alarm and from this determine if it constitutes a problem and/or determine any remedial actions which must be taken in relation to the assessment. All of the alarms triggered by the Beran system are audited by an external body on a quarterly basis to ensure that all events occurring on the turbine are being thoroughly and accurately assessed. Advanced signal processing tools such as FFT's and magnitude and phase plots are provided by the Beran system which allow experts to view the data in various representations which assist in determining the alarm cause and can provide more detail to the assessment.

A problem associated with this approach is that the large volume of condition monitoring data required to assess each alarm can greatly intensify the workload of the relatively few experts qualified to interpret the data when many alarms are triggered. This problem is compounded in instances where the alarm is triggered but has no further operational consequence hence providing little or no information on the equipment behaviour. This research aims to support British Energy by creating novel data analysis and learning methods leading to an automated system to reduce the burden of analysis currently imposed upon the small team of turbine generator condition monitoring experts. The system also aims to standardise the analysis approach over all of the British Energy locations throughout the UK to provide consistent and accurate assessments for the external quarterly audits. In a strategically important application such as this it is important that the users build confidence in such a system. For this reason a significant emphasis has been placed on the explanation provided by the system's reporting facility. Rule-Based Expert System technology has been chosen specifically due to its ability to provide a clear and logical rationale through its explicit knowledge base. An additional benefit for choosing such an approach was the opportunity to capture the knowledge of a small number of turbine generator condition monitoring experts within British Energy. Acquiring the domain knowledge for the system's knowledge base during the development stage of the Rule-Based Expert System was an intensive and time consuming process, taking key personnel out of daily duties while it happened. Therefore a learning module was also developed during the course of the project to assist the experts in deriving explicit heuristic knowledge to improve the performance of the Rule-Based Expert System. The Explanation Based Generalisation (EBG) approach, which the learning module is based upon, was selected due to its compatibility with the learning problem encountered in this type of domain in that there is access to relatively few training examples but a multitude of domain knowledge exists.

In terms of the novelty of the research undertaken, three primary contributions can be identified:

- **Augmentation of the existing condition monitoring approach through the introduction of intelligent automated processing.**
  The application of the Rule-Based Expert System developed for this project is novel. The system had to augment a well established approach to turbine generator condition monitoring within British energy. It had to be designed to interface with the existing condition monitoring system and provide the British Energy experts with the well defined information required for them to perform their assessment.

- **Novel use of graphical approaches to provide explanation of Rule-Based Expert System rationale.**
  Central to the ethos of the Rule-Based Expert System design was the effectiveness of how the assessment explanation was fed back to the user. In addition to the use of rule-based explanation, novel approaches to graphically highlight features used within the assessment have been incorporated into the developed prototype to assist the user during verification.

- **Novel semi-autonomous approach to diagnostic condition monitoring knowledge derivation.**
  The learning module developed to assist in deriving explicit heuristic knowledge for the Rule-Based Expert System is novel in both its approach and application. The learning module designed and developed uses an adapted approach of EBG that utilises causal fault and behavioural models. The module aims to reduce the burden associated with capturing knowledge for Rule-Based Expert System applications.

1.2 Thesis Outline

Chapter 2 outlines the area of research covered in this thesis namely condition monitoring. A general overview of why condition monitoring is applied in many industrial applications and the various approaches employed in different applications are discussed. In addition a detailed description is given of the turbine generator condition monitoring system used within all British Energy locations throughout the UK, which is the Beran monitoring system. Chapter 3 reviews the area of AI. The first part of the chapter focuses on the 3 approaches, Rule-Based Expert Systems, Artificial Neural Networks (ANNs) and Model-Based Diagnosis (MBD), commonly used within diagnostic tasks. The second part of the chapter gives an overview of the Machine Learning (ML) domain and reviews approaches such as analytical learning, rule induction, Case Based Reasoning (CBR), Bayesian approaches and Hidden Markov Models (HMMs). Chapter 4 describes in detail the Rule-Based Expert System designed and implemented for British Energy. This includes a review of the knowledge engineering approach used to capture and document the expert knowledge, an in-depth description of the final design and an assessment of how the prototype performed in assessing genuine historical alarms from the Beran system. Chapter 5 reviews in detail the learning module developed for British Energy. This includes a full description of the causal knowledge modelling approach developed specifically for this project, the novel approach developed for the module and its performance when tested on historical case studies of genuine faults which occurred on British Energy turbine generator sets. Chapter 6 details the conclusions and contributions from the research and outlines areas of further work that can be undertaken as an extension to this research.

1.3 Associated Publications

M. Todd, S. D. J. McArthur, G. M. West, J. R. McDonald, S. J. Shaw, J. A. Hart, "The Design of a Decision Support System for the Vibration Monitoring of Turbine Generators," 39th International Universities Power Engineering Conference, vol 1, pp 433-437, 6-8 Sept 2004.

M. Todd, S. D. J. McArthur, J. R. McDonald, S. J. Shaw, "A Machine Learning Based Intelligent System for the Condition Monitoring of Turbine Generators," Artificial Intelligence in Energy Systems and Power, AIESP 2006.

M. Todd, S. D. J. McArthur, J. R. McDonald, S. J. Shaw, "A Semiautomatic Approach to Deriving Turbine Generator Diagnostic Knowledge," IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews, Issue 5, pp 979-992, Sept 2007

Chapter 2

2 Condition Monitoring

2.1 Condition Monitoring Applications

Condition monitoring is a commonly used approach within a wide range of sectors to allow organisations to more effectively manage their assets. The approach relies upon access to some form of data which can either directly or indirectly infer some condition or behaviour of the equipment being analysed. From this the asset owner can make a more informed decision on how the equipment should be effectively operated or maintained based on its predicted state. In practice it has been employed to assist in the monitoring of expensive or strategically important assets to plan maintenance outages, make operational adjustments based on the predicted condition and determine fault locations. Condition monitoring is used in cases where an accurate assessment of components central to the operation of strategically important or expensive plant items is difficult to achieve through direct human inspection. Therefore an important aspect of most condition monitoring approaches is that they are non-intrusive and don't require an interruption of the equipment's normal operation.

There are a wide range of applications where condition monitoring approaches have been applied over numerous sectors. Factors which influence the deployment of condition monitoring approaches are the cost of implementation versus the return on reduced maintenance costs and/or increased production. Generally a significant emphasis is placed on the monitoring of larger and more expensive items such as large turbine generators, transformers and reactor cores. The condition monitoring of large turbine generators historically relies on the monitoring of alarms generated from the continuous monitoring of equipment parameters such as vibration, steam temperature and operational parameters such as load [Cauvin et al, 1998 and Gonzalez et al, 1986]. These parameters are normally measured through transducers positioned at strategically important positions on the equipment. How the data is interpreted is dependent on the sophistication of the condition monitoring approach itself. The most basic approach would be for an alarm to trigger based on the absolute value of some measured parameter. This alerts the operator to some anomaly on the system so that an analysis of the condition of the equipment can be performed based on the various parameters at his or her disposal. The analysis could be performed on the absolute values of the parameters captured but more sophisticated approaches exist which are able to dissect these parameters in an attempt to extract more information from the raw data signals, particularly from the vibration signals. A commonly used approach would be to dissect the vibration into individual frequency components through the use of an FFT [Lynn & Fuerst, 1998]. This can allow common behaviours or fault types to be diagnosed based on the predominant frequency components normally associated with those behaviours. One such turbine generator condition monitoring system which has incorporated an FFT analysis tool is the Beran system which is described in detail in section 2.2. The analysis of temperature is largely based on expected operating temperature and any deviations from that which may indicate a particular behaviour or problem. Another common data type used to monitor generators in particular is the electrical data derived from the generating equipment. One simple approach is to match the power output of the generator against the input power to the turbines. This must take into consideration the designed operating characteristics of the equipment itself such as the efficiency of the turbine and generator. A difference in the expected performance against the actual performance may indicate some form of problem which has to be further investigated to determine its cause. More sophisticated forms of analysing electrical data exist where the signal is dissected into individual frequency components using an FFT. This is a similar approach to that applied to vibration signals in order to uncover underlying behaviours which are not apparent in the raw signal. The name given to this analysis is Current Signature Analysis (CSA) [Royo & Arcega, 2007]. No examples could be found of the CSA approach being applied to large turbine generator sets but an example of it being applied to squirrel cage wind turbine generators is given in [Royo & Arcega, 2007]. Here the authors demonstrated that through analysing the frequency spectrum of the current signals broken rotor bars, turn to turn stator faults and bearing damage could be detected.

Electrical transformers found within the distribution network and to a lesser extent within industrial facilities are another type of high cost and strategically important device which are commonly the subject of condition monitoring procedures. One widely adopted condition monitoring approach used to determine the condition of the device's insulation, both paper and oil, is Disolved Gas Analysis (DGA) [CIGRE, 2003]. DGA is an approach where oil samples are taken from the insulating oil of a transformer and then chemically analysed to determine whether the internal insulation is of a good condition. Among some of the conditions, which can be identified through DGA, are high levels of water within the oil insulation itself which can significantly impact on the dielectric strength of the device. DGA can also identify breakdown in the paper insulation through the identification of a high Degree of Polymerisation (DP) in the given oil sample. This measures the level of degradation in the cellulose by interpreting the generation of Furans in the oil [CIGRE, 2003]. The presence of high DP can also be used to indicate that partial discharge or arcing, which are common modes of transformer failure, are taking place [CIGRE, 2003]. Another approach for detecting partial discharge within transformers is through the use of Ultra High Frequency (UHF) signals [Judd et al, 2002]. UHF signals are captured via probes positioned externally on the transformer body. The signals are captured and then processed through advanced signal processing approaches to produce a spectra. Highly skilled personnel are then able to analyse this spectra to identify the presence of any partial discharge activity within the transformer. Systems have also been developed to automate the interpretation of the UHF signals such as COMMAS [McArthur et al, 2004].

Another important, expensive plant item where methods of condition monitoring are becoming more common is the reactor core used to house the radioactive material within nuclear power stations [West et al, 2006]. The life of the reactor core essentially determines the life of the nuclear power plant. The importance of such an item is twofold due to the implications associated with lost revenue during unplanned downtime and perhaps more importantly the risk to human life and environment if such a piece of infrastructure were to catastrophically fail. A type of degradation is the cracking of the graphite bricks which comprise the reactor core. One of the more traditional approaches used to monitor and diagnose such deficiencies is through visual inspections via cameras which are fed down the refuelling tubes embedded within the reactor but this can only be performed during equipment outages. A more advanced approach to detecting the same type of deficiency is to capture and analyse the variations in frictional pull exerted by the pulley system otherwise known as a Fuel Grab Load Trace (FGLT) when refuelling the core [West et al, 2006]. Variations in frictional pull can be used to identify areas where the refuelling cylinder is not uniform, which in turn indicates some form of deformity in the core. The main cause of these deformities is cracking of the graphite bricks.

The nuclear reactor core application described above is a good example of where existing data is re-used for the purposes of condition monitoring. This is in contrast to the transformer and turbine generator applications where sensing equipment is installed specifically for condition monitoring purposes. A result of installing sensing equipment specifically for condition monitoring purposes is the increase in cost. As a result, condition monitoring has traditionally been applied to equipment which has a high enough cost, or is strategically important to the company's business needs, to merit the high cost of implementing such approaches. There are, however, condition monitoring approaches which utilise some form of data which already exist as a by-product of the operation or maintenance of the equipment such as the FGLT described above. A good example of this type of application is the condition monitoring of induction motors through CSA [Thomson & Fenger, 2001] & [Culbert & Rohdes, 2007]. CSA has been applied to induction motors to monitor their electrical and mechanical behaviour. The approach dissects the motor current signal using FFTs to determine their frequency distribution. The frequency distribution can then be interpreted to determine if the motor is exhibiting any degraded and/or faulty behaviour. One of the main benefits of this type of approach is that, with the exception of the device which captures and processes the current signals, there needs to be no installation of any additional transducers which can be expensive. The current drawn by the motor is accessible from the starter cubicle within the supplying substation meaning that monitoring devices such as these are relatively cheap and easy to install. The ability to install cheaper devices on less expensive but larger volumes of equipment provides the potential to increase the efficiency of plant wide processes further by introducing more condition based maintenance plans for parts of the process which were traditionally left to routine maintenance schedules.

One of the primary benefits of using condition monitoring to assess plant equipment is that it allows companies to move from a routine based maintenance schedule, whereby outages are planned based on the time elapsed since the previous shutdown, to a schedule based on the condition or performance of the equipment. The advantage of this is that the cost of an outage is only incurred when some form of maintenance is required. The business can have parts which have long lead times ordered in advance so that the outage is not extended anymore than it needs to be. The net effect is a reduction in maintenance costs where the outage is planned and where there is prior knowledge of problems present on the equipment. In addition, the safety of employees, customers and the public is increased where operators have a better understanding of the condition of the equipment being operated.

A consequence of the growing need for sophisticated forms of condition monitoring is the increased workload placed upon company experts to analyse and interpret the data produced by these approaches. Much work has been done to help minimise the effort required to assess this data using automated systems. Automated systems exist which assist humans in some form to help monitor the condition of equipment or systems. At the lowest level are systems which continually monitor raw data signals and trigger an alarm or warning when pre-defined limits are breached, eliminating the need for continuous human monitoring of the raw data streams. A level up from this is provided by packages which in addition to triggering alarms provide the user with advanced processing tools which allow the raw sensor data to be viewed in various formats. The additional information made available by these techniques enables the operator to uncover events which otherwise may not have been found using only the raw data.

British Energy is a company that is affected by the problem of having to process ever increasing volumes of data collected by various condition monitoring approaches. One division within the company which is particularly affected by this problem is the Rotating Plant and Dynamics Team which is responsible for the safe operation and maintenance of some of British Energy's most strategically important assets, such as the 16 turbine generator sets throughout its eight UK nuclear power locations. The rotating plant team has developed a process for the condition monitoring of its turbine generators to help standardise the approach employed by each team member, who is responsible for his/her own particular location within the UK. The process developed has only been made possible by the installation of an on-line condition monitoring system (the Beran system) which is fully described in the following section.

2.2 Beran System

The online condition monitoring system used by British Energy to monitor their turbine generators is developed by Beran Instruments Limited and is referred to within British Energy and throughout this thesis as the Beran system. The specific version used throughout all of British Energy's locations is the Beran 766. The Beran system is an online condition monitoring tool which can be used for the analysis of any type of rotating equipment providing there is access to raw data signals. The Beran system captures raw data from transducers positioned on the equipment to allow condition monitoring experts to analyse and interpret the signals both in real time or retrospectively. Typical signals captured by the Beran system from the transducers are operational parameters such as voltage, stator current, rotor current, steam and bearing temperatures along with eccentricity and vibration measurements. From this the system can infer useful parameters such as the system load components such as the real, reactive and apparent power. The Beran system is able to dissect the overall vibration signals into vectors which specify the behaviour of the magnitude and phase of each harmonic. A first order vibration would imply that it occurs once every cycle, i.e. the fundamental frequency or first harmonic. Therefore for a turbine rotating at the rate of 3000rpm, first order vibration would be at a frequency of 50Hz. Similarly a second order vibration, or second harmonic, would occur at twice the frequency of the first order, therefore in this instance occurring at a rate of 100Hz. The orders of vibration are each identified by two parameters. These are magnitude and phase. The magnitude parameter as the name implies represents the peak to peak amplitude of the vibration signal. In addition to this the phase angle, that is the value by which the signal is leading or lagging the keyway reference point on the rotor shaft, is also specified.

Alarm limits can be set to monitor the value of any of the parameters monitored by the system. These can be parameters measured directly by the system, such as absolute vibration levels, steam or bearing temperatures. Alarm limits can also be set to monitor derived parameters such as first and second order vibration magnitude and phase, generator load, reactive loads etc. The system allows each of the monitored parameters to be defined as either an alert or an alarm. Alarms are normally parameters which are of greater importance than alerts and therefore require immediate investigation if triggered. For example, an alarm may be set to trigger when a critical level of vibration, which may impact on the safe operation of the equipment, if reached. Alerts can be set for less critical parameters which may indicate a change in equipment behaviour but may not necessarily represent a critical state. For example changes in first and second order magnitude and phase can indicate a change in behaviour without the equipment reaching a state which impacts on the safe operation of equipment. Therefore, such parameters are typically set to alerts, so that the experts are made aware to further investigate the cause of such changes. The Beran system indicates any alarms or alerts which have occurred on the system in a grid format as shown in figure 2.1. The alarm status and alarm display screen given in figure 2.1 indicates that there are two alerts active on the monitored equipment. Both of the alerts are on the order 1 parameter which is either the first order vibration magnitude or phase of the rotating equipment on both channels 8 and 10. All alerts are highlighted in yellow. There are no alarms in this example which are highlighted in red. If the parameter is coloured green then this indicates that there is neither an alarm or alert present.

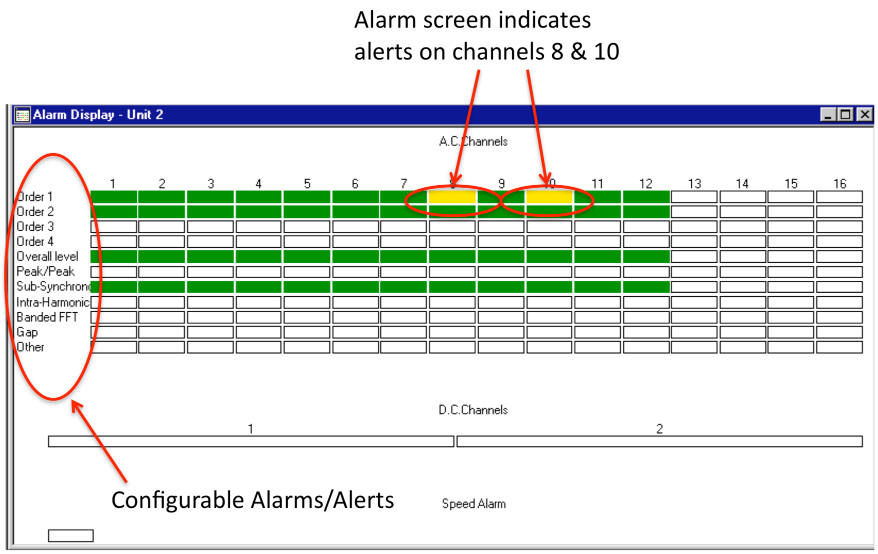


Figure 2.1: Alarm status and alarm display screen captured from the Beran on-line condition monitoring system.

The Beran system provides numerous signal processing tools to allow condition monitoring experts to investigate the cause of any alarms or alerts present on any of the channels. One of the signal processing tools provided by the Beran system is the ability to plot any of the measured signals in time series. Any signal from any channel can be selected and the time period over which the signal is viewed is specified. An example of the time series plot is given in figure 2.2.

In this example the overall vibration amplitude and first and second order magnitude and phase for channel 8 has been plotted. The data has been plotted over a period of an hour on 08/06/04. The data can be viewed in two different resolutions. The highest resolution is where a sample is captured every 5 seconds over a period of an hour. This type of data is referred to as buffer data and is only captured when an alarm or alert is activated. The higher resolution data is captured 30 minutes before and after the alarm triggering. This is achieved by a rolling buffer, where the system continually saves 30 minutes worth of high resolution data. Therefore, when an alarm is triggered, the 30 minutes worth of data contained in the rolling buffer is saved to memory in addition to a further 30 minutes worth of high resolution data following the alarm. The time series data plots given in figure 2.2 are examples of the buffer data type. The other type of time series data available is of a lower 10 minute sample rate resolution. This type of data is constantly saved to the hard disk therefore the period can be defined over any date and time the system has been operational.

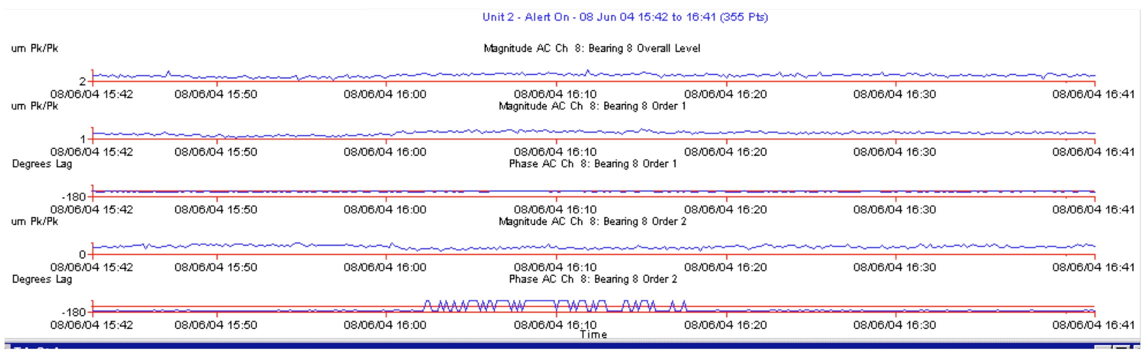


Figure 2.2: Time series data display captured from the Beran on-line condition monitoring system.

An additional signal processing tool provided by the Beran system to assist the condition monitoring experts in their analysis is the ability to decompose vibration signals into their individual frequency components using the FFT function. An example of an FFT plot taken from the Beran System is given in figure 2.3. An FFT plot depicts the distribution of the magnitudes of a particular signal over a certain frequency spectrum. This allows experts to quickly determine at what frequencies significant magnitudes are present. For example the first FFT plot in figure 2.3 has a significant magnitude present at the first order frequency (50Hz) along with smaller magnitudes at the second and sub-synchronous frequencies. However the second FFT plot contains a much higher proportion of second order frequency magnitude (100Hz) which may be indicative of a particular behaviour such as misalignment faults.

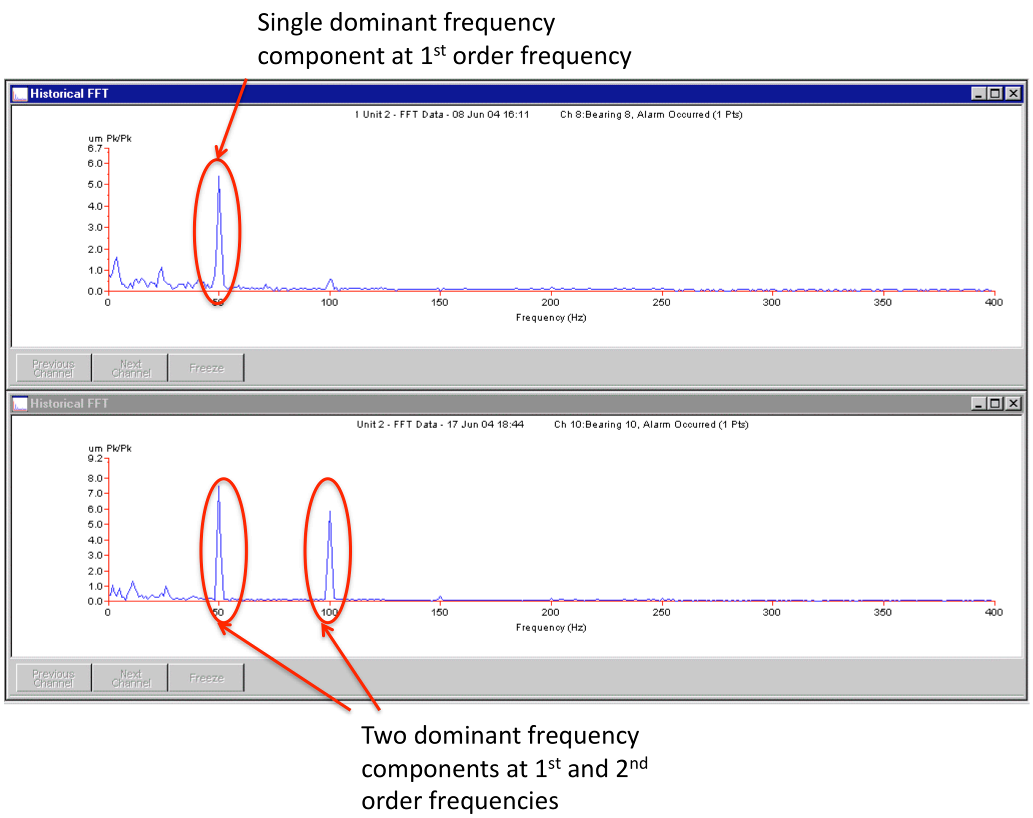


Figure 2.3: FFT plots captured from the Beran on-line condition monitoring system.

Another signal processing tool provided by the Beran system allows the user to monitor the variation in amplitude and phase of any order of vector over time with respect to the alarm limits set for that vector. The user is able to view this information using a magnitude and phase plot. An example of a magnitude and phase plot taken from the Beran system is given in figure 2.4. The plot depicts the movement of the first order vector over time. The alarm limits are denoted by the circle. As can be seen in figure 2.4 the vector position highlighted by the dark red colour starts within the alarm limits and over time moves outside of the limits, which is denoted by the transition in colour to yellow. The remaining colours on the scale have been omitted from the plot.

The Beran system also provides additional tools such as alarm logs which allow the user to view the sequence of alarms which have been triggered through an alarm log. A file list screen indicates file types which have been saved for a particular channel. Typical files saved for a channel are associated with the run-up and run-down of the turbine generator, which enables the expert to easily determine its state, i.e. whether the it is off-line or on-line.

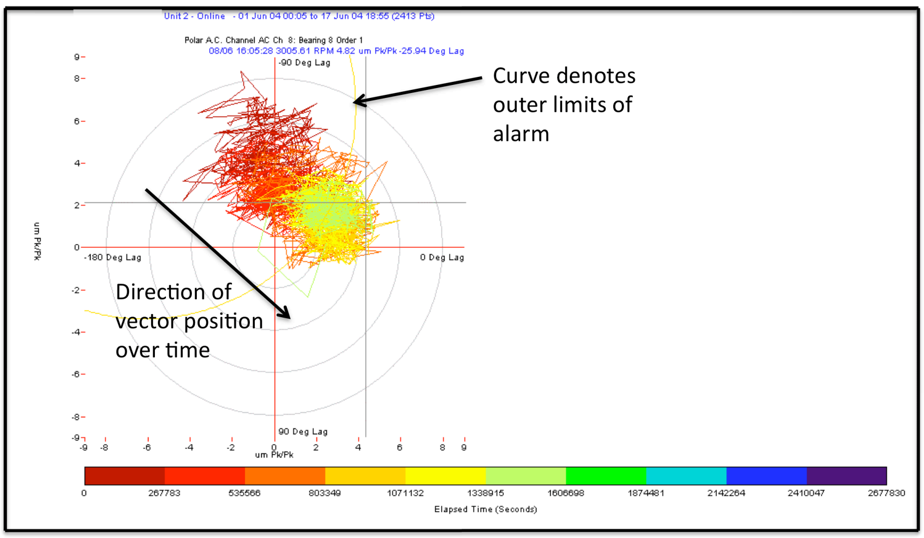


Figure 2.4: Magnitude and phase plots captured from the Beran on-line condition monitoring system.

2.3 British Energy Turbine Generator Condition Monitoring Approach

A significant emphasis is placed on the condition monitoring of turbine generators at each of British Energy's nuclear power stations within the UK. The condition monitoring of each turbine generator aims to continually assess the behaviour of each plant item to ensure that incipient faults are diagnosed as early as possible. Early fault detection can allow plant operators to take remedial action in the form of operational changes which will help avoid or minimise the damage caused by the fault and assist in the effective planning of outages. British Energy has a core team of three full time employees and five part time contractor staff who continually assess the behaviour of each turbine generator. This assessment is achieved by analysing key signals captured from transducers on the turbine generator sets which, when correctly interpreted, can indicate how the equipment is behaving. The Beran system described in section 2.2 has been installed at eight British Energy UK nuclear power locations to capture and feedback the raw transducer data to the condition monitoring experts for inspection. In addition the Rotating Plant and Dynamics Team has developed a process to maximise the facilities provided by the Beran system to ensure that each alarm triggered is fully assessed. The process also helps standardise the condition monitoring process across the whole of the rotating equipment group, ensuring that the analysis performed at all locations is consistent.

The first stage of the process is to set alarms or alerts on the Beran system, which continually monitors important signals and can indicate a change in equipment behaviour. The purpose of these is to alert staff to potential behavioural changes on the turbine generator, which may constitute a problem requiring further investigation. There are a standard set of signals which are monitored across all British Energy locations. In addition, the experts are able to set additional alarms outside the standard set if they feel it will assist them in the assessment process. The standard alarms set across all locations are as follows:

- Subsynch High – Alarm triggers when the magnitude of the sub-synchronous frequency components are too high.
- 1X Vector – Alarm triggers when the first order vector moves outside of the alarm zone. A movement in either the magnitude, phase or both in the first order component can initiate this alarm.
- 2X Vector – Alarm triggers for the same reasons as the IX Vector alarm except the second order component is being monitored.

- 1X Step – Alarm triggers when there is a significant change in the first order component vector. This could be triggered by a significant change in the first order magnitude, phase or both.
- Zone 2 – Alarm triggers when overall vibration or eccentricity falls within the predefined zone 2 limits. These limits indicate the severity of the vibration with zone 2 being the lowest level of vibration out of the 3 alarms.
- Zone 3 – Same as for zone 2 except the predefined zone 3 limits are used. Zone 3 limits represent a higher level of vibration than the those given by zone 2 limits
- Zone 4 – Same as for both zones 2 & 3 except the predefined zone 4 limits are used. Zone 4 limits represent a severe level of vibration which would indicate serious problems on the turbine generator.

The sub-synchronous high alarm can alert the condition monitoring expert to the presence of various fault types which can be characterised by a high magnitude at one of the sub-synchronous frequencies. The triggering of this type of alarm may prompt the expert to investigate the sub-synchronous frequencies further to determine which one is at a high magnitude and then relate this to a particular behaviour. Common fault types which can be represented by higher than normal sub-synchronous components are rotor rub and oil instability. A rotor rub fault is commonly represented by higher than average frequency magnitude at between 1/2X and 1/3X the running speed although the precise location of the frequency component is dependent upon the position of the machine critical speed in relation to the running speed. Oil instability can be highlighted by higher than average magnitudes at between 0.42X and 0.47X the running speed. Again the precise location of the frequency component cannot be precisely specified for all examples of oil instability since it is dependent on a number of variables.

The 1X Vector, 2X Vector and 1X Step alarms alert the expert to changes which have occurred in first or second order vibration. These parameters are of particular interest, given that many fault types produce changes in both of these components which may be masked if the expert were to only analyse overall amplitudes. Common faults, which can be diagnosed by large magnitudes at 1X the running speed are various forms of imbalance such as static imbalance and rotor imbalance. Both of these conditions will always have a 1X component which dominates the spectrum. Therefore the triggering of the 1X vector alarm can alert the condition monitoring experts to investigate the presence of such faults. Misalignment fault types such as angular, parallel or bearing are highlighted by large frequency magnitudes at both 1X and 2X the running speed. Therefore the triggering of either the 1X Vector or 2X Vector alarms can lead the condition monitoring experts to investigate such faults. The triggering of a 1X Step alarm can be indicative of a serious fault such as the loss of a piece of material from the rotor which would cause a sudden change in balance and in turn cause a sudden change in the 1X Vector. The Zone type alarms monitor the overall amplitude primarily to alert the experts to when the turbine generator may be reaching problematic levels of vibration.

The condition monitoring expert must diagnose the cause of any triggered alarm and from this determine if it constitutes a problem and/or determine any remedial actions which must be taken in relation to the assessment. Each alarm must have an alarm checksheet completed which ensures that the expert explores the majority of the potential causes of the alarm. An example of the alarm checksheet is given in figure 2.5.

OA Amp	Zone	Non-synch. Amp/Freq	1x Amp	1x Phase (*lag)	2x Amp	2x Phase (*lag)	OA Genuine?	OA ≈ Σ(1x+2X)?	Step Δ1x?	Signif. Δ2x?	Operational change?	Commentary (likely cause, action taken or recommended)

Figure 2.5: Alarm checksheet completed for every Beran alarm triggered.

The first 7 entries in the checksheet are concerned with capturing data from the overall amplitude, sub-synchronous 1X and 2X vector signals in the Beran system. The first entry OA Amp requires that the approximate value of the overall amplitude signal for that particular channel is recorded. The Zone entry asks the expert to enter what zone the overall magnitude level falls within i.e. 1, 2, 3 or 4 with 1 being the lowest level of vibration and 4 being the highest. The most dominant sub-synchronous magnitude including the frequency it falls within is recorded in entry 3: ‘Non-synch. Amp/Freq’. The 1st and 2nd order magnitude and phase levels are recorded in the ‘1x Amp’, ‘1x Phase (*lag)’, ‘2x Amp’ and ‘2x Phase (*lag)’ entries. The next 5 entries are questions which guide the expert into exploring particular alarm causes. The ‘OA Genuine?’ entry prompts the expert to assess whether the overall amplitude signal is genuine or not. If the signal is not genuine then this can indicate that there is some form of signal fault on the channel and therefore explain the cause of the alarm. The next ‘entry OA ~ ∑(1x+2x)?’ asks the user to consider if the overall vibration amplitude level seems to be made up of mainly the 1st and 2nd order frequency amplitudes. If they are not then this would prompt the expert to investigate other ordered frequencies to determine what additional significant frequency components there are, either at sub-synchronous or multiples of the operating speed. The presence of other significant frequency components apart from 1st and 2nd order frequency components can indicate the presence of faults such as rotor rub and oil instability mentioned previously. The next entry, ‘Step Δ1x?’, prompts the expert to determine if there has been a step change in the 1st order vector in either the magnitude, phase or both. Such an event can be indicative of a serious fault such as a loss of material from the rotor. The ‘Signif. Δ2x?’ asks the user to determine whether there has been a significant change in the second order vector in either the phase, magnitude or both. Any such change can alert the expert to faults such as a cracked shaft, looseness or misalignment. The final question, ‘Operational change?’, is to determine if an operational change has occurred which could be related to the alarm. A common cause of alarms is a change in operation which has some impact on one of the monitored parameters, in particular ones which are operating close to their alarm limits.

In some instances, the expert can conclude the root cause of the alarm based solely on the data captured to complete the checksheet entries, allowing the ‘Commentary’ entry of the checksheet to be completed without any further investigation. The Commentary entry contains the cause of the alarm and any remedial actions required based on the given assessment. The cause of some alarms cannot be concluded following the completion of the checksheet entries, since additional information is required to determine the cause. In this situation the expert will investigate the alarm further using other signals associated with the affected channel or data relating to neighbouring channels. Once the cause of the alarm is concluded the Commentary section of the checksheet is completed,

The results of these analyses can be monitored to determine if any faults are developing on the equipment and to subsequently plan any actions which may have to be taken based on these results. Much emphasis is placed on monitoring and reporting of these behavioural changes as a result of the strategic importance of plant items such as turbine generators. Regulatory bodies undertake quarterly inspections of all alarms triggered by the Beran system in each station to ensure that all events are being thoroughly and accurately assessed.

Experience has shown that many alarms are commonly caused by faulty signals, signal drift or changes in operational parameters which cause the vibration signals to temporarily move outside their limits. Generally these signals do not provide the experts with information on the health or state of the equipment and so have no further operational consequence. However each alarm must be inspected by one of the three full time staff and five part time contractors within British Energy who are qualified to do so. This effectively intensifies the already substantial daily workload on this small team.

This problem prompted British Energy to commission a project which aimed to develop a system capable of automatically diagnosing the cause of each alarm triggered by the Beran system. The system had to enable an expert to select an assessed alarm, review the system analysis and then sign off the alarm. This would allow the expert to quickly confirm alarms of no further operational consequence and focus their time and expertise on diagnosing incipient faults which may impact on the health and operation of the turbine generator. In addition the system should assist in standardising the analysis performed across all British Energy locations so that the results are accurate and consistent for the external quarterly audit.

This research has employed approaches from the area of Artificial Intelligence (AI) to develop an automated system for the turbine generator condition monitoring application detailed above. In addition, this research has used an approach from a subgroup of AI called Machine Learning (ML) to develop a novel approach to the problem of capturing the required knowledge for an automated system. The next chapter reviews the area of AI and in doing so justifies the approaches chosen for the project.

Chapter 3

3 Artificial Intelligence for Diagnosis & Learning

3.1 Introduction

Artificial Intelligence (AI) is defined by [Barr and Feigenbaum, 1981] as "the part of computer science concerned with designing intelligent computer systems, that is, systems which exhibit the characteristics we associate with intelligence in human behaviour – understanding language, learning, reasoning, solving problems and so on." As outlined in chapter 2 the aim of the project was to develop a system capable of automatically assessing the Beran system alarms to assist the experts in the alarm assessment process. Therefore one of the primary aims of the automated system is to perform the analysis which is already undertaken by the human experts at present. AI approaches would therefore seem suited to this task due to their ability to reason with "characteristics we associate with intelligence in human behaviour" and "reasoning." Additionally the system developed has to be able to perform the analysis automatically which again is a trait of AI approaches since they are "computer systems." It is for these reasons that the area of AI was researched to find solutions to the British Energy turbine generator condition monitoring problem.

AI techniques which have been applied to condition monitoring as a tool for diagnosis include Model-Based Diagnosis (MBD) [Davis and Hamscher, 1988], Rule-Based Expert Systems [Jackson, 1999], Case-Based Reasoning (CBR) [Kolodner, 1993], statistical based approaches such as Artificial Neural Networks (ANNs) [Haykin, 1999] and Hidden Markov Models (HMMs) [Rabiner, 1989]. These techniques use some form of knowledge to solve a pre-defined task and the type of knowledge utilised varies depending on the adopted approach. MBD, Rule-Based Expert Systems and CBR utilise a more explicit form of symbolic knowledge which is more easily interpreted and understood by humans whereas the statistical based knowledge used in ANNs and

HMMs are less so. How this knowledge is derived or captured is also dependent on the approach. Typically the knowledge adopted by Rule-Based Expert Systems is captured and formalised through a knowledge engineering approach such as CommonKADS [Schreiber et al, 2000]. The models used by MBD approaches may already exist as part of the design package for a piece of equipment or may have to be developed specifically for the system. The knowledge used by a CBR system is derived from a collection of examples associated with a particular subject of interest, therefore it is common to adopt this approach where an adequate collection of examples exist and are accessible. Approaches such as ANNs or HMMs derive knowledge from the statistical distributions contained in a collection of examples on a particular subject and therefore rely on having access to that collection of examples.

A growing area of AI which is concerned with the automated acquisition of knowledge is Machine Learning (ML). ML can be defined as "any change in a system that allows it to perform better the second time on repetition of the same task or on another task drawn from the same population" [Simon, 1983]. ML techniques offer the potential for automated systems to learn knowledge from training data or in some cases already existing knowledge used in conjunction with training data. ML approaches include ANNs, CBR and HMMs which were mentioned in the previous paragraph when discussing AI approaches. Other ML techniques include Bayesian learning [Mitchell, 1997], rule induction algorithms such as C4.5 [Mejia-Lavelle & Rodriguez-Oritz, 1998] and its successor C.5 [Strachan 2005], Explanation Based Generalisation (EBG) [Mitchell et al, 1986] and Explanation Based Learning (EBL) [DeJong & Mooney, 1986]. As already discussed ANNs, CBR and HMMs can be used as AI approaches to diagnosis. It was also stated that all three of these techniques derive their knowledge in some form from a population of examples relating to the subject of interest. It is this derivation of knowledge from a collection of examples which make all three of these approaches examples of machine learning. Bayesian approaches are also statistical based and can learn knowledge from the statistical data contained within a body of training examples. Rule induction algorithms such as C4.5 and C.5 use a collection of training examples to derive explicit rule based knowledge which is in a form to that used by Rule-Based Expert Systems. EBG and EBL are both ML approaches which use existing knowledge of the area of interest to derive explicit rule based knowledge using few or in some cases single training examples.

The aim of this chapter is to review AI approaches, which have been used to develop both automated systems for condition monitoring and diagnosis applications and ones which are capable of deriving the knowledge used by these systems. The first half of this chapter will review AI techniques which have been applied to condition monitoring and diagnosis problems. This will focus primarily on Rule-Based Expert Systems, ANNs and MBD approaches since a significant proportion of academic literature for industrial applications has reported on these approaches. Examples of other AI approaches, which have been used in the area of diagnosis, will be discussed where appropriate. A detailed discussion on the CBR and HMM approaches is left to the second half of the chapter which reviews AI approaches associated with ML. This discussion also includes a discussion on Bayesian learning, EBG, EBL and rule induction approaches such as C4.5 and C.5. Rule-Based Expert Systems, MBD and CBR are all commonly referred to as examples of Expert Systems due to their use of explicit knowledge. However, the remainder of this thesis will use the term Expert Systems to refer only to Rule-Based Expert Systems and not MBD or CBR.

3.2 Expert Systems

The approach employed by Expert Systems is based around two key aspects of human reasoning:

1. Some body of domain knowledge which can interpret input data to derive conclusions.

2. A reasoning approach which determines how this knowledge is applied in order to derive a solution.

A high level overview of a typical Expert System is illustrated in figure 3.1. The input data is fed into the signal to symbol transformation module to derive qualitative symbolic data from the raw input signal. The transformed qualitative data is uploaded to working memory, sometimes referred to as the interpreter, which controls how the input symbols are matched with the individual fragments of knowledge contained within the knowledge base. This can be thought of as a local level of control which tasks include resolving conflict where more than one item of knowledge can be activated and whether the knowledge should be activated using known data or alternatively reasoned backwards from the goals to be achieved. A global level of control is facilitated by the inference mechanism which is used to determine the order in which tasks are undertaken by controlling the flow of knowledge uploaded to the working memory. The conclusions drawn from the input data by the applied knowledge are then fed to the end user by means of an appropriate interface.

Four important considerations when designing an Expert System are:

- Knowledge Capture
- Knowledge Representation
- Inference Mechanism
- System Maintenance

The first consideration is concerned with how the knowledge used by the Expert System is captured from the domain experts. Once this knowledge has been captured it must be transformed into a suitable representation which allows it to be utilised. The third consideration deals with the reasoning approach which should be used in order to maximise the performance of the domain knowledge in order to achieve the required task. Finally, the designer must consider how the system, once implemented, can be maintained and who is responsible for its maintenance. Each of these considerations are discussed in the following sections.

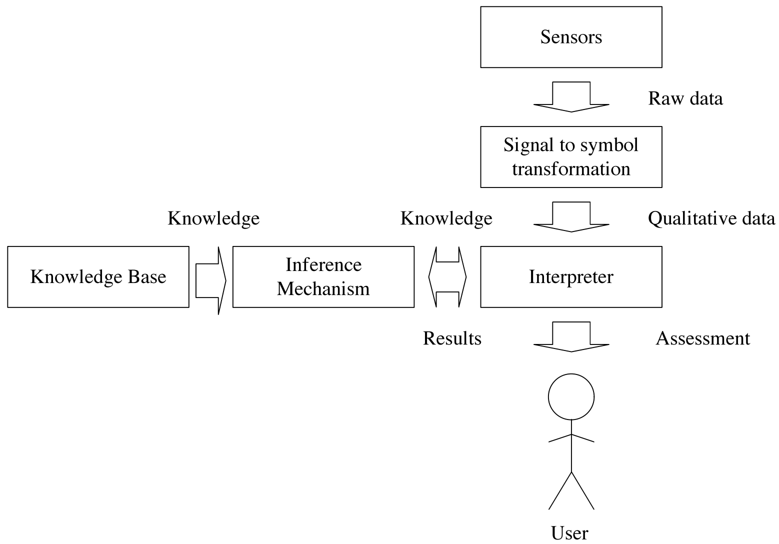


Figure 3.1: Overview of the main Expert System components

3.2.1 Knowledge Capture

The most common approach to capturing expert knowledge is through a knowledge engineering approach which in general can be divided into the following areas:

- Knowledge elicitation – The process of capturing the expert knowledge.
- Knowledge modelling – The process of recording & modelling the knowledge captured at the elicitation stage.
- Knowledge Analysis – The process of reviewing and correcting the knowledge captured and modelled at the knowledge elicitation stage.
- Knowledge Structure – Determining the most suitable representation for the captured knowledge within the Expert System.

One of the first attempts at knowledge elicitation was protocol analysis [Newell & Simon, 1972] where experts were encouraged to think aloud when performing tasks so that both their approach and knowledge could be observed and recorded. This technique still forms the basis of most approaches to knowledge elicitation which typically involve an interview between the domain experts and the knowledge engineer. It is common for the first interviews to take the format of protocol analysis to allow the knowledge engineers to gain a general understanding of the problem area and allow the experts to get used to 'thinking aloud.' Once the knowledge engineers have gained a more firm understanding of the area then the interviews can become more focussed on particular aspects of the problem in a bid to make the captured knowledge as detailed, correct and error free as possible. These interviews are normally recorded using audio and video recording media.

The knowledge captured during the interviews is recorded then transcribed and finally transformed into knowledge models. The transcripts are structured documents which detail both the expert's reasoning approach and domain knowledge. The transcript is then reviewed by the expert and his/her peers to find and correct errors or areas where more knowledge is required. The reviewed transcript is corrected and this process is continued until the knowledge contained within the transcript is correct.

The knowledge must then be transformed into a format whereby it can be used as the knowledge base for the software implementation of the Expert System. It is common for the knowledge to be modelled in some format which reflects how it can be utilised by the Expert System prior to the knowledge being encoded into a knowledge base. The modelling formalisms selected at this stage should be understandable to the experts so that they can be further validated and at the same time bear some relation as to how the knowledge will be represented and reasoned with by the system.

There is no de facto approach to modelling the captured knowledge, but one notable formal methodology is CommonKADS [Schreiber et al, 2000]. CommondKADS offers a complete modelling formalism for representing both the domain knowledge and the reasoning approach applied to any particular problem. The knowledge is divided into three main categories; task; inference; and domain. The task knowledge defines the overall objective of the problem, for instance a common objective of a condition monitoring system might be to identify and/or 'diagnose a fault'. This overall objective can then be dissected into smaller sub-tasks, which when performed, achieve this overall goal. Therefore the 'diagnose a fault' task may be divided into 'locate anomaly', 'capture data', 'extract features', 'identify events', 'apply diagnostic knowledge', and finally, 'select or construct diagnostic conclusion'.

The inference knowledge models the reasoning approach used to achieve the overall objective. In effect, the inference approach assigns in what order the tasks are completed and highlights the flow of information including input data, derived data and domain knowledge. The domain knowledge models any concepts, theories, rules or any other piece of relevant information which is particular to the problem being analysed. For instance, if the above diagnosis task is in the field of transformer condition monitoring, then the domain knowledge might include details on how to analyse the information derived by ultra high frequency partial discharge signal analysis or a gas in oil sample.

CommonKADS offers a formal modelling mechanism to model the domain knowledge which is compatible with the Unified Modelling Language (UML), a standard used within the computer science community for modelling object-oriented data for programming languages such as Java (http//www.uml.org). The models are also intuitive for experts, with no background in computer science, to understand the tasks undertaken by the systems, the flow of data and how the knowledge is represented which assists at the knowledge verification stage.

It is vital that experts with the required expertise for the domain of interest exist in order to capture the necessary knowledge for an Expert System using a knowledge engineering approach. This requirement was fulfilled for the turbine generator condition monitoring project since certain members of the Rotating Plant and Dynamics Team were willing to participate in the project. The above knowledge engineering process described so far has been purely manual, in that the interview process and the creating of knowledge transcripts and models is undertaken by humans. This process has been shown to be time consuming [Fenton et al, 2001], [Awadallah & Morcos, 2003] and is therefore only beneficial in projects where there is a high return on the associated high cost in using such a technique to develop an Expert System. Due to the operational importance of the turbine generators to British Energy's core business and the already time consuming nature of the existing manual analysis process it was felt that the use of a knowledge engineering approach in such an application would be beneficial.

There have been some attempts at developing on-line tools which work with the domain expert to assist in capturing knowledge. One example stemmed from the development of ONCOCIN [Shortliffe, 1981], an Expert System which develops treatment plans for cancer patients. The knowledge acquisition tool developed is known as OPAL and its approach to assisting with knowledge capture is based mainly on its user interface. Entities and relationships corresponding to drug treatments are entered via graphical forms where the user selects items from a menu of alternatives. These forms are then translated into frames (section 3.2.2.3) and are linked with other objects within the knowledge base through a hierarchy. The user is also allowed to enter procedural knowledge which relates to plans for administering combinations of drugs. The acquisition of this knowledge is again facilitated by the programmes graphical user interface, which allows the user to create icons standing for plan elements and arrange them into a graphical structure. The user can then position and draw connections between these elements to create charts which mimic the control flow. These can then be converted into a format which the system is able to utilise. OPAL's main asset was its graphical user interface and its use of forms and icons which allowed the user to enter the domain entities and relationships along with the procedural knowledge. In addition, OPAL's ability to translate this knowledge into a form which could be utilised by an automated system was also very important. It is worth noting that OPAL's functions were mainly passive in that there was no assistance given to the expert in deriving the domain knowledge unlike manual knowledge elicitation where the interviewer will encourage the expert to consider different scenarios. OPAL has lead to the development of a general purpose system called PROTÉGÉ and its subsequent successors [http://protege.stanford.edu/]. Although some research has been undertaken in developing tools, which assist with the knowledge capture process, knowledge engineering for Expert Systems is still largely an intensive human based process. It is for this reason that as part of the British Energy turbine generator condition monitoring project, methods of introducing machine learning based AI approaches were researched as discussed in sections 3.6 to 3.11 which lead to the development of a semi-automated learning module as described in chapter 5.

3.2.2 Knowledge Representation

A system developer must determine how the knowledge captured at the knowledge elicitation stage is represented within the knowledge base. Certain approaches are more suited to problems with particular characteristics than others, but no definitive rules exist to determine what representation to choose. Three approaches to knowledge representation are formal logic representations, production rules and frames.

3.2.2.1 Logic

A formal approach to representing the knowledge of a particular domain is through logic approaches [Russell & Norvig, 1995] such as propositional logic. Propositional logic allows sentences to be constructed from atomic sentences which consist of single indivisible propositional symbols such as ‘bearing vibration high’ or one of the truth values ‘true’ or ‘false’. Complex sentences are constructed by applying logical connectives to the atomic sentences. These logical connectives are negation ($\neg$), conjunction (∧), disjunction (∨), implication (⇒) and bi-conditional (⇔). A formal grammar can then be defined as shown in figure 3.2.

A knowledge base can then be constructed by defining complex sentences for the domain of interest such as 'bearing A vibration high' ⇒ 'bearing A out of balance'. It should be apparent that in using this approach each propositional symbol must be explicitly defined and therefore no general statements can de defined.

Sentence: AtomicSentence, ComplexSentence

AtomicSentence: **True**, **False**, P, Q. R, …

ComplexSentence: (Sentence),
Sentence Connective Sentence,
¬Sentence

Connective: ∧, ∨, ⇔, ⇒

Figure 3.2: Propositional logic grammar

First order logic is a more expressive form of knowledge representation which allows objects and relations to be defined. For example the statements "rotor of turbine A has fault stiction" and "rotor of turbine B has fault stiction" would each be represented as a single proposition in propositional logic. First order logic allows statements like this to be defined in a way which is more closely related to natural language processing. It achieves this by expanding the syntax of propositional logic to include symbols which represent objects, relationships and functions. Objects are represented by constant symbols, relationships are represented by predicate symbols and functions are represented by function symbols. Therefore 'turbine A', 'turbine B' and 'stiction' could be represented by object symbols, 'rotor' could be represented as a function symbol and 'has fault' could be represented as a predicate symbol to construct the 'HasFault(Rotor(TurbineA), Stiction)' and 'HasFault(Rotor(TurbineB), Stiction)' first order logic sentences. These atomic sentences can then be used to form complex sentences using the same approach and logical connectives as used by propositional logic.

First order logic also provides a method of defining variables which allows whole collections of objects to be reasoned with as opposed to defining each one individually. This is achieved through two standard quantifiers called the universal and existential quantifier. The universal quantifier ($\forall$), meaning “for all”, allows sentences to be constructed which refer to a whole body of variables. For example, the statement $\forall x$ $Rotor(x) \Rightarrow Rotates(x)$ reads as “for all of x, if x is a rotor then x rotates.” This allows general statements to be made without having to name each rotor individually and state that it rotates. The universal quantifier ($\exists$), meaning “there exists an”, allows sentences to be constructed which refer to some objects within a body of variables. For example, the statement $\exists x$ $SupportedBy(x, PedestalA) \wedge Bearing(x)$ can be read as “there exists a bearing which is supported by pedestal A.” This gives permits generality through allowing a statement to be made about some object without naming that specific instance.

Knowledge relating to a particular domain can be represented using the rich syntax and semantics offered by first order logic. The full expressiveness offered by this representation has found popularity in mathematical theorem proving [Newell et al, 1963], [Newell & Simon, 1963] and acted as the basis for the general problem solver PROLOG [Clocksin & Mellish, 1984]. Logic is also a popular form of knowledge representation for symbolic Machine Learning (ML) approaches such as Explanation Based Generalisation (EBG) [Mitchell et al, 1986]. These applications are well suited to this type of representation because the laws governing these domains are well defined and complete, meaning that any conclusions drawn are correct. However, it is more difficult for the domain knowledge of real world problems to be represented in this way. This is because real world problems can only be modelled using theories which are approximate and don’t always hold true in every conceivable situation. Therefore, an accurate representation of the domain knowledge is either not possible or prohibitively complex. The turbine generator condition monitoring application is an example of where it is very difficult to construct a domain theory using logic due to the complexity of the domain.

3.2.2.2 Production Rules

A simplified form of propositional and first order logic is that of horn clauses [Luger & Stubblefield, 1998]. Horn clauses represent knowledge in a form which relates more closely to the type of knowledge used by humans to reason about complex domains. Horn clauses are implications where the antecedent is composed of positive literals (P_1, P_2, …, P_n) and the consequent is a single positive literal (Q) as in equation 3.1.

$$P_1 \wedge P_2 \wedge \ldots \wedge P_n \Rightarrow Q \tag{3.1}$$

Horn clauses were the pre-cursor to production rules [Jackson, 1999] which have become a mainstay in knowledge representation in Expert Systems. Production rules are in the form of if <condition> then <conclusion>. The condition in the statement can be made up of multiple conditions whereas only a single conclusion is permitted as is the case for horn clauses. Production rules have become widely used because they allow the knowledge to be represented in a format which resembles more closely the way humans store their knowledge. For instance a turbine generator condition monitoring expert, when attempting to diagnose a fault, does not under normal circumstances revert back to the fundamental laws governing the operation of turbine generators such as thermodynamics, mechanics, electrical theory, etc. Instead the expert will refer to experience of the domain where observations of the equipment behaviour relate to a particular state or fault. For example an expert may conclude that if there is a step change in either the first order magnitude or phase then this may imply that a significant piece of the rotor blade has become detached from the main body of the rotor. The expert can deduce this through his/her own experience of the domain and is unlikely to go back to first principles to derive the same conclusion. This is an approximate form of reasoning because the observation mappings are acting as a shortcut to the conclusion by directing search as opposed to exhaustively searching the potential solution space. Knowledge represented as production rules is commonly referred to as heuristic knowledge since the actual knowledge prunes the search space as opposed to relying purely on heuristic search methods.

A further advantage of simplifying the knowledge in this way is that more simplified inference approaches can be adopted to search the knowledge to derive conclusions than those adopted in logic. One such inference algorithm is forward chaining which takes the known facts of a problem and applies them to the knowledge base. A conclusion is activated and added to the existing facts whenever a fact or group of facts matches with the rule's conditions. This process is repeated until the query is answered (if one exists) or there are no new facts to be derived. The direction of the reasoning in forward chaining is highlighted in figure 3.3. An alternative to forward chaining is backward chaining which selects the goal from the consequent and chains backwards through the clauses to determine if the facts support the proof. The direction of the reasoning in backward chaining is highlighted in figure 3.3.

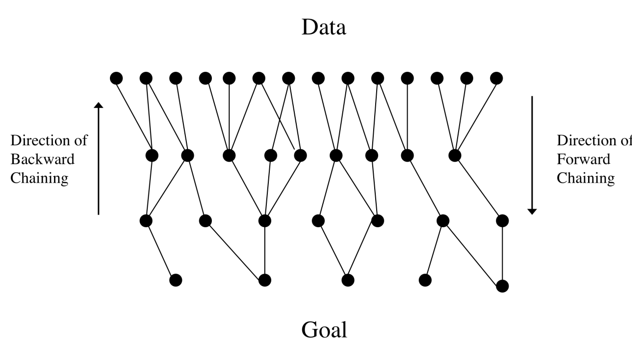


Figure 3.3: Forward and Backward reasoning inference approaches.

Production rules are well suited to representing knowledge in domains which can move from the states of a particular problem towards the solution or vice versa. The turbine generator condition monitoring application is an example of a problem where the expert analyses the state of the equipment using the information provided by the Beran system and utilises this to deduce its behaviour. Production rules do not explicitly represent the properties and interrelationships of complex objects found in real world domains. One knowledge representation approach which can represent this associative knowledge is a frame-based representation.

3.2.2.3 Frames

Frames [Luger & Stubblefield, 1998] attempt to organise knowledge into categories in a similar way to how humans organise their knowledge of similar objects which share common characteristics. It achieves this by grouping together all of the properties associated with a particular object using a single data structure. The object being represented could be that of a typical object found in the real world such as a car, a species of animal such as bird or could even be used to represent a group of faults which are closely related. A frame has a single slot which is used to store the entity that it represents. For example a frame structure which represents a car would have a slot to indicate this. Further slots contain the common attributes associated with the object, or procedures which can be used to derive additional information. Multiple related frames are normally arranged into a hierarchy, where frames lower down the network can inherit values for slots from higher up the hierarchy. The fundamental idea is that the properties and procedures represented higher up the frame system represent things that are typically true about the entity of interest, whereas the frames at lower levels contain slots that are particular to specific types of the entity represented. For example, a frame structure could be constructed for a 'bearing fault'. The primary frame structure at the top of the hierarchy may contain a slot to indicate that a high vibration is normally associated with the fault. Further down the hierarchy various types of bearing faults may be specified such as oil whirl which will specify that the bearing would have to be lubricated by oil and that an increase in vibration would be expected at 0.42x and 0.48x the running speed. Alternatively a inner race type bearing fault could be specified which would indicate that the bearing would have to be of a ball bearing type and would exhibit a frequency distribution which is dependent on the number of ball bearings and the running speed. Frames are useful in instances where objects must be represented that share many similarities, but also have unique differences. As is outlined in section 4.3.1, there are only nine alarm causes identified for the turbine generator condition monitoring application and most of them exhibit distinct features from one another. Therefore, using frames as a form of knowledge representation for this particular problem was not necessary. However, there is a case for using such an approach if a system had to be developed to diagnose fault types where there is commonality between various faults.

3.2.3 Inference

The inference approach of an Expert System is concerned with the process undertaken to achieve the overall goal of the system. The overall task of a system, e.g. diagnosis, classification, etc., can be dissected into smaller sub-tasks. Once the overall task has been dissected then the order in which these smaller sub-tasks are undertaken must be determined, in addition to the flow of information between tasks and any additional knowledge required to achieve the goals. Therefore the inference approach is effectively the reasoning process undertaken to achieve the overall goal.

The CommonKADS methodology [Schreiber et al, 2000] attempts to use general inference approaches to common tasks which have to be undertaken by a Knowledge Based System (KBS). For example the CommondKADS methodology suggests that the diagnosis task is divided into the following:

- Cover – Determine all possible causes of complaint.
- Select – Select a single possible cause from a whole population of causes.
- Specify – Choose an observable entity which could be used to confirm or rule out hypotheses.
- Obtain – Acquire the actual value of the observable entity specified in the previous step.
- Verify – This step checks the candidate cause to determine if it should remain as a potential hypothesis for the complaint.

These tasks can then be represented within an inference structure like the one given in figure 3.4 which is suggested by the CommonKADS methodology in [Schreiber et al, 2000].

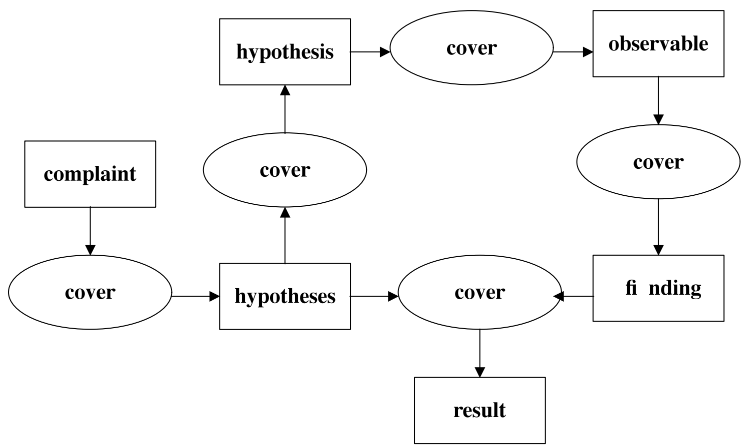


Figure 3.4: Generic CommonKADS inference structure to the task of diagnosis.

An alternative to using generic structures suggested by knowledge modelling approaches such as CommonKADS is to develop a specific approach for a particular task. This can be achieved using the results obtained from the knowledge elicitation process. The transcripts resulting from the knowledge elicitation exercises, are normally structured to describe more manageable sub-tasks individually, to reflect the natural approach adopted by the expert. This information is used firstly to develop the task model. The information contained within the task model then defines the tasks associated with the inference approach. In addition, the transcript will also specify the ordering in which these tasks are carried out and any knowledge required to achieve each. All of this information can then be combined to construct a specific inference approach for the overall task. This method was used to develop the inference approach utilised for the turbine generator condition monitoring application and is described in detail in section 4.3.2.

3.2.4 System Maintenance

The designer must also consider how the Expert System is to be maintained once implemented. This is an especially important consideration for Expert Systems since it is common that the knowledge contained within the knowledge base will have to be updated and/or revised, based on the performance of the system and further experience gained in the field. Therefore, some issues that the designer must consider are:

- Who will maintain the knowledge base?
- How will feedback on the system performance be recorded?
- Will any tools be provided to assist in updating the knowledge base?

The system designer must determine up front who is expected to maintain the knowledge base. Traditionally it has been the job of the knowledge engineer and/or the system maintenance engineer to perform this task. However, there may be scope to include the expert user(s) in this process to varying degrees. For instance, it is beneficial that the performance of the system is recorded, especially for instances where the system has failed to provided an accurate diagnosis. Given that the expert user is best placed to identify when inaccurate assessments have occurred then it is sensible for the system designer to provide facilities for the expert to record these instances. This then allows the knowledge engineer and/or the system maintenance engineer to analyse this information and determine what updates are required in the knowledge base. It may also be desirable to provide tools for the expert user, to assist them in deriving new knowledge in instances where the system does not perform adequately, such as, the one described in chapter five of this thesis.

3.2.5 Practical Issues for Expert System Applications

The main disadvantage of Expert Systems is the difficulty associated with acquiring the necessary knowledge for a particular subject/area and transforming this into a suitable format which can be utilised by the system. The knowledge engineering approach is in most cases time consuming and labour intensive which therefore makes it expensive. This has meant that within industry Expert Systems have mostly been developed for applications which merit the initial large investment. Such applications may be where the system is developed to monitor strategically important or expensive plant items. Another difficulty, which can arise with Expert Systems, is that there may be no way of acquiring the knowledge of a particular area due to the lack of expertise. This problem is particularly relevant where the application involves some form of recent emerging technology or a newly developed device.

Applications where this difficulty in acquiring knowledge is offset by the importance of the system being monitored and where the domain is well understood are well suited to the use of Expert Systems. The explicit symbolic nature of the knowledge used by Expert Systems allows a rationale of the assessment to be built up by referencing the rules whose triggering led to the conclusion. This allows the users to build confidence in the system assessments because they can clearly understand the rationale behind each decision. This is also a benefit in instances where an incorrect assessment is constructed so the user can follow the reasoning and determine where the knowledge must be updated.

There are however drawbacks associated with expert systems which are present regardless of the application. The first is the problem known as “conflict resolution.” This problem refers to cases when the input data to the Expert System results in multiple conclusions or in a diagnostic sense the symptoms are consistent with more than one possible diagnosis. This problem has more impact in areas where the system is being designed to supply users, who have little or no knowledge in a particular domain, with a definitive answer to some query. It is not as much of an issue where the system is designed as a decision support system for a user who is expected to have a certain level of knowledge for a given application and where explanation of the assessment rationale is provided. Another problem inherent in Expert Systems is that the knowledge remains static unless further knowledge engineering is undertaken to manually update the knowledge base. In effect there is no automated learning undertaken which allows the knowledge to be updated based on prior performances. This therefore makes Expert Systems expensive and time consuming to maintain or expand.

3.2.6 Applied Expert Systems

Expert Systems have been in use for the condition monitoring support and fault diagnosis of systems and equipment in numerous applications within the power systems domain. An alarm processing and fault diagnosis Expert System is reported in [Protopapas et al, 1991] which interrogates the non-expert user to enter information such as power system protection flags and maintenance information to determine the cause of faults on a distribution network. The system utilises diagnostic rules which have been captured from field experts and are represented using a tree structure. The information entered by the user is added to the relevant nodes at the root of each tree to determine if all of the conditions within the tree are satisfied. If all of the conditions within the tree are satisfied then the system concludes that the fault represented by that fault tree has occurred. The tree also allows the user to visualise the rationale produced for any of the system conclusions since the information input by the user can be traced through the tree back to the conclusion.

Another alarm processing Expert System for fault diagnosis on a power distribution network is described in [Minakawa et al, 1995]. The system was developed for and implemented on one of Japan's power distribution networks. The system required, on average, three engineers to investigate and develop a prototype system over a period of 27 months and 10 engineers to develop the prototype to the implementation stage over a further 12 months. This demonstrates the significant effort required to design and construct an Expert System, meaning that only applications where there is the required economic return are considered for such systems. The authors also discuss the problem of conflict resolution where multiple conclusions exist for a single event. The system deals with this problem by trying to add additional knowledge to the Expert System which attempts to differentiate between competing hypotheses. This additional knowledge requires additional information from the protection relay event recorders which is not available in all cases due to some areas of the network possessing less modern and advanced technology. The authors explain that this lack of detail in some of the information recovered from the system explains that when the system is tested, only 34% of correct conclusions have only a single hypothesis, whereas 56% have more than one. The authors predict that the solutions where conflict exists could be reduced by improving the information fed back to the Expert System. The system employs explanation by taking the system diagnosis and simulating it back to the user using the power network single line diagram which the operators use to monitor the network on a day to day basis. This therefore relays the rationale back to the operators using a graphical user interface and one with which they are familiar and is easily understood.

An Expert System is reported in [Strachan et al, 2008] which diagnoses faults on HV Power Transformers. The system utilises Ultra High Frequency (UHF) data captured from probes placed on the outer casing of the transformer. This UHF data can be used to construct what is known as Phase Resolved Partial Discharge (PRPD) patterns which transformer condition monitoring experts can interpret to diagnose and locate Partial Discharge (PD) behaviour. The Expert System attempts to extract the same features from the PRPD data using statistical analysis which the human experts would in their own analysis. The statistical description of the PRPD is analysed using a rule based approach to determine the physical discharge behaviour being exhibited, potential discharge sources, the failure type and its location within the transformer. The authors report that the system's performance benefits from the explanation provided with each assessment due to the explicit nature of the knowledge, in contrast to other pattern recognition techniques which generally provide little or no explanation. The knowledge utilised by the Expert System was captured from experts using a knowledge engineering approach but there is no indication given as to the level of resources required to capture the knowledge.

One of the earliest examples of an Expert System application in the domain of turbine generator diagnosis was the on-line diagnostic support tool described in [Gonzalez et al, 1986]. The system captured data from multiple sensors placed on the turbine generator and from this performed an analysis on the data to determine if there was a problem on the equipment. A confidence factor was calculated for each diagnosis to deal with the issue of conflict resolution from probabilities associated with that fault captured at the knowledge elicitation stage and probabilities associated with the reliability of the sensor data captured. The confidence factors associated with the sensor reading were derived using knowledge on the failure characteristics of the sensors themselves. The users were able to view a rationale for the diagnosis in the form of a tree structure which also included confidence factors associated with sensor readings and with the diagnoses themselves. The system also provided the operator with suggested tests which could be undertaken on the equipment to provide a higher confidence in the diagnosis or disprove it. An input facility was therefore provided where the operator could enter additional data captured from the tests performed. The system would also advise the operator on any action which should be taken based on the assessment of the equipment state. Despite the system being implemented on-line with fully operational turbine generators, none of the major faults occurred on any of the turbine sets which could adequately test the system. At the time of writing the paper, the authors were only able to verify that the system was able to diagnose sensor faults which had occurred on the equipment. The authors also reported on the large amount of knowledge elicitation exercises which were performed over a 6 month period with a knowledge elicitation engineer and multiple employees from the Westinghouse Electric Corporation

The system reported in [Gemmell, 1995] was developed to provide ScottishPower with an on-line turbo alternator condition monitoring tool to be used in real time by the equipment operators. As well as providing diagnostic assessments based on the measured data, it also provided a module which validated the raw data signals using knowledge on cross sensor corroboration. The author reports on how a structural model of the turbo alternator was developed using an object-oriented programming approach. These models could also be used to store condition monitoring data associated with each component. The diagnostic knowledge within the rule base was developed using a production rule knowledge representation. Similarly to [Gonzalez et al, 1986], a lack of data to test all of the rules in the knowledge base is reported due to major faults rarely occurring. However, the system performed well when tested for two major fault groups using historical case studies.

The use of a tree structure explanation in [Protopapas et al, 1991] and [Gonzalez et al, 1986] demonstrated that Expert System approaches were useful in applications where the system is required to provide an explanation of the rationale to the user. It was also demonstrated in [Minakawa et al, 1995] that alternative approaches to explanation could also be employed such as simulation and the use of graphical formats with which the operators/experts are familiar. The condition monitoring experts at British Energy specifically requested that any system developed would have to provide the user with an explanation of the assessment. Therefore the Expert System approach would appear well suited for providing the necessary explanation for this application either through the reporting of the knowledge used to derive the conclusion or through novel forms of graphical explanation or both. The degree of human resources required to develop the Expert System reported in [Minakawa et al, 1995] and [Gonzalez et al, 1986] demonstrate that applications where there is a strategic importance or high capital cost should only be considered for the application of Expert Systems. There is no doubt that the British Energy turbine generator condition monitoring project is such an application due to the high capital cost of the equipment and the high operating losses which would be inflicted upon the company if the generators were to experience a forced outage. However the time consuming nature of the traditional methods of knowledge capture do indicate that a particular area of research is in the development of approaches to assist in this process. Another important issue highlighted in [Minakawa et al, 1995] and [Gonzalez et al, 1986] was that Expert Systems are required to deal with the issue of conflict resolution. The approach in [Minakawa et al, 1995] to dealing with conflict was to use more detailed data on the events and in turn more detailed knowledge in order to differentiate between conflicting hypotheses whereas [Gonzalez et al, 1986] employed a from of approximate reasoning which used probability factors captured from the experts themselves. If an Expert System were to be developed for the turbine generator condition monitoring application, then the issue of conflict resolution would have to be addressed. Finally both [Gemmell, 1995] and [Gonzalez et al, 1986] allude to the fact that genuine faults on the equipment are relatively rare, meaning that it is difficult to acquire any form of training data within this application which would allow more data intensive techniques to be employed, as opposed to more knowledge based techniques such as Expert Systems.

3.2.7 Further Reading

For a more detailed discussion on the historical development of Expert Systems, the various approaches and technologies employed by Expert Systems and detailed descriptions on seminal systems in areas outside of the power systems domain the reader is guided to [Jackson, 1999], [Luger & Stubblefield, 1998] and [Russell & Norvig, 1995].

3.3 Artificial Neural Networks

ANNs are primarily a biologically inspired attempt to recreate intelligence using an approach similar to the way in which the human brain processes the vast quantities of data captured by human senses. The data in an ANN is processed by a densely interconnected network of artificial neurons which store knowledge implicitly through interconnecting weights. This is in contrast to the symbolic based AI methods, which represent problem spaces using symbols that model certain characteristics in the domain of interest. ANNs also have the ability to learn as well as interpret data by adapting the weights between interconnecting neurons through the use of learning algorithms.

One of the earliest examples of using neurons in the field of computing was by McCulloch & Pitts [McCulloch & Pitts, 1943]. They demonstrated how any logic function could be realised using a simple neuron which consisted of two logic inputs, a bias input and a single output. This work was important in demonstrating that these neurons were able to implement computational functions, although interest in the technology only started to grow with the development of learning algorithms such as that used by [Rosenblatt, 1958], which used perceptron neurons as shown in figure 3.5.

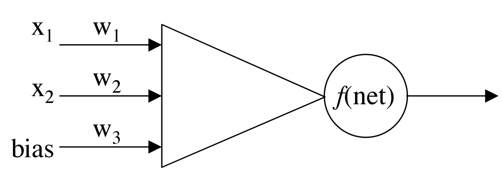


Figure 3.5: A perceptron net which can learn linearly separable functions

Where bias = 1, x_1 and x_2 are the two variable input values, and w_1, w_2 and w_3 are the weighting factors used to multiply inputs x_1, x_2 and bias respectively. Therefore, f(net) is as given in equation 3.2.

$$f(net) = f(w_1*x_1 + w_2*x_2 + w_3*1) \tag{3.2}$$

In equation 3.2, if f(net) is above or equal to 0 then the output is set at 1 otherwise the output is set to –1. If the signs of both the f(net) output and the training data are the same then the weights remain unchanged. If the signs are not the same then the weights are updated using a learning function. Perceptrons were able to learn linearly separable functions by comparing the network's output with the desired result and then feeding back the error between each to adapt the weights within the network. A limitation of these single layered networks was their inability to learn non-linearly separable functions. This limitation was only overcome with the introduction of multilayered networks and the use of continuous threshold functions which were differentiable such as that used by back-propagation learning algorithms [Haykin, 1999]. Back-propagation networks are, like perceptron learning, are a form of supervised networks which means that the data used to train the network is labelled to indicate the classification of each example. In addition to this type of supervised approach, unsupervised approaches such as the self organising map (SOM) [Haykin, 1999] were developed. Both of these approaches will now be described in the following sections.

3.3.1 Back-propagation Artificial Neural Network

The basic unit in a back-propagation ANN is the Sigmoid unit as shown in figure 3.6. The Sigmoid unit computes the linear combinations of its inputs and then applies this value to a continuous threshold function. The continuous nature of the threshold function allows it to be differentiable, as opposed to the discrete non-differentiable threshold function used by the perceptron given in figure 3.5. This attribute allows the Sigmoid unit to utilise what is known as the delta rule which enables multi-layer networks to learn non-linearly separable functions. This is a very powerful tool since a significant amount of data analysed in practical implementation problems normally exhibit non-linear characteristics. Before examining how a multi-layered back-propagation network is able to learn non-linear functions, it is necessary to consider how a single Sigmoid unit is able to learn linear functions using both the gradient descent algorithm and the stochastic approximation learning rule.

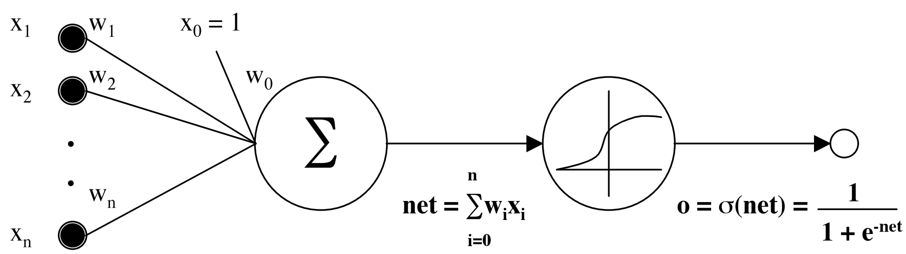


Figure 3.6: Sigmoid unit used in a back-propagation ANN.

The key idea behind the gradient descent algorithm is to search the hypothesis space to determine the weight vector which minimises the output error of a Sigmoid unit and hence find a function which best describes the training data. The unit output error is minimised by reducing the mean squared error of the single unit. The hypothesis space is searched to find the steepest gradient by differentiating the error with respect to each individual weight component on the unit input. This derivative can be expressed in terms of the node inputs, the expected output and the actual output. This allows the rule, which calculates by how much each individual weight in the network is updated, to be expressed in terms of the same parameters as in equation 3.3. A comprehensive explanation of how equation 3.3 is derived is given in [Mitchell, 1997].

$$\Delta w_i = \eta \sum_{d \exists D} (t_d - o_d)\, x_{id} \tag{3.3}$$

Where Δw_i is the change in weight, x_{id} denotes the single input component x_i for training example d, t_d is the target value for training example d, o_d is the output for training example d and η is a positive constant called the learning rate. The gradient descent rule learns the weight vector for a single unit by randomly initialising each weight. Each example is then applied to the network and the amount by which each weight is updated is calculated using equation 3.3. Each weight is then updated and the process is repeated until the error has been minimised and the solution weight vector is found. A variation of this is the stochastic gradient descent, which instead of finding the mean squared error over all the examples, finds the mean squared error of a single example and uses the results to update the weight vector. This is achieved by applying one training example at a time as opposed to running all training examples at once. This method provides an approximation to the minimisation of error, but is computationally much more efficient. The weights of the unit are updated using the delta rule, as shown in equation 3.4. Comprehensive explanation of how equation 3.4 is derived is presented in [Mitchell, 1997].

$$\Delta w_i = \eta(t - o)\, x_i \tag{3.4}$$

Where t, o and x_i are the target value, unit output, and i^{th} input respectively for the training example in question. Back-propagation networks are constructed using Sigmoid units as shown in figure 3.7. These are multilayered networks which typically consist of an input layer, a hidden layer and an output layer. The multilayered network configuration means that the solution space is multi-dimensioned which enables non-linear functions to be represented using such networks. This ability to represent non-linear functions is a very powerful tool in real-world domains which require the interpretation of non-linear data sets. The learning algorithm used by a back-propagation network is based on the same gradient descent or stochastic approximation approaches explained above. Back-propagation networks, however, can have multiple output nodes and contain at least one layer of nodes where the expected output of each unit is not known directly.

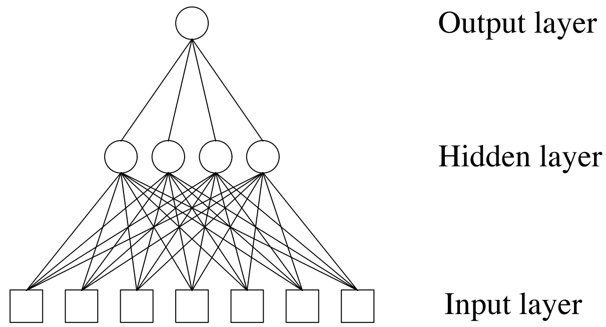


Figure 3.7: Multilayered back-propagation network consisting of input, hidden and single node output layer.

The back-propagation learning algorithm attempts to reduce the error of the solution space in the same way as a single Sigmoid unit. Since a back-propagation network can contain multiple output units then, the sum of the squared error over all of the units is minimised. The calculation of the magnitude by which each weight in the network should be modified is different in a multi-layered network due to the added complexity. A detailed discussion on the equations used to update the weights on both output and hidden layers can be found in [Mitchell, 1997]. The back-propagation algorithm searches the solution space using an approach identical to gradient descent as outlined earlier. The weights of the network are randomly initialised and all examples are applied to the network to determine the error of each unit. All of the training examples are then applied to the updated network and the unit weights are again updated accordingly. This process is repeated until the error margin is reduced to an accepable level.

3.3.2 Self Organising Map Artificial Neural Network

The objective of a SOM network is to learn a function which groups together input examples that exhibit some form of similarity. It achieves this by taking an input example, which is represented as some vector of an arbitrary length, and transforming this onto a 2-dimensional lattice. The network must therefore be capable of grouping together similar input vectors on the lattice in a topological fashion. The interconnecting weights between each input and output unit are trained so that a function exists which can cluster input examples with similar features. However, the training data used to train a SOM is not labelled to assist the training algorithm as is the case for back-propagation networks. Labelling of the data is not necessary since the network aims to group together similarly featured objects, not objects of a particular classification.

A SOM network is composed of a 2-dimensional lattice of output neurons. Each neuron has weighted connection to each component of the input vector. An example of SOM network with a 3 component input vector and a 3x3 output lattice is depicted in figure 3.8 [Haykin, 1999]. The input vector v has values A, B and C. Each output is labelled o_n and each weight vector is labelled w_{nv} where n = 1, 2, …, 8 and v = A, B, C.

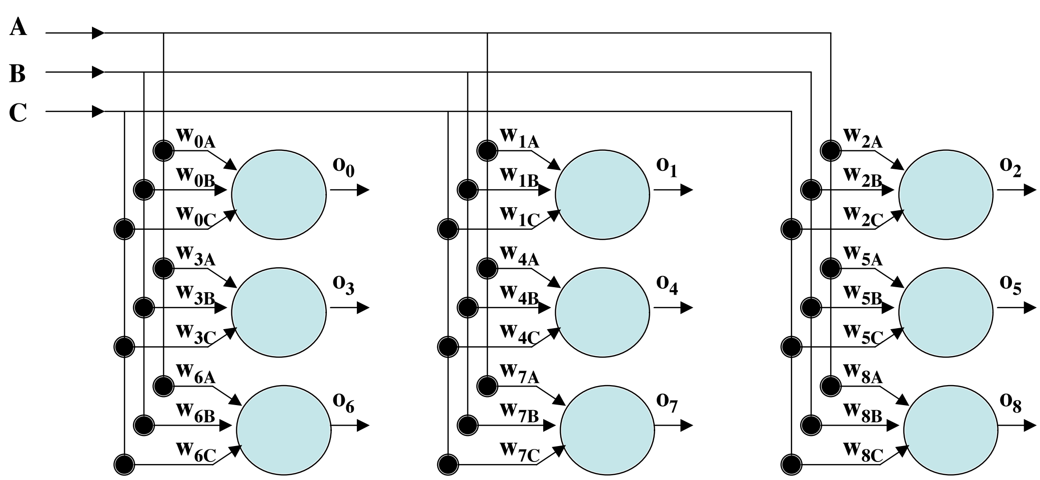


Figure 3.8: A SOM network with a 3 component input vector A, B and C and a 3x3 output lattice.

The algorithm, which trains the network, is composed of three main stages. These stages are:

- Competition
- Cooperation
- Adaptation

In competition each output neuron computes its output in relation to the input vector and the interconnecting weights. The neuron with the largest valued output is deemed the winner and this is commonly calculated by minimising the distance between both the input vector and the weight vector as described in [Haykin, 1999].

The winning output unit is found during the competition process, it is the task of the cooperation process to determine the location in the output lattice of a neighbourhood of nodes which will have their weights updated. These nodes which will have their weights updated are defined as the excited nodes. This neighbourhood is calculated using a Gaussian distribution as described in [Haykin, 1999] which shrinks with each iteration in the training process.

The final stage of adaptation updates the weights of each excited neuron within the neighbourhood so that the application of a similar input to the network would result in each updated node producing a higher valued output. A detailed description of the equation used to update each weight in the neighbourhood can be found in [Haykin, 1999]. This equation effectively moves the weight vector of the winning neuron and the excited neurons closer to that of the input vector.

To train a SOM the weights within the network are firstly randomly initialised. A training example is then selected from a population of training examples and is applied to the three step process of competition, cooperation and adaptation outlined above.

Another training example is then selected and the whole process completed until the SOM meets the required termination criteria.

3.3.3 Practical Issues for Artificial Neural Network Applications

ANNs have been applied to a considerable number of domains encompassing a vast array of problems. Arguably the biggest advantage of this approach is the ability of the network to acquire the necessary knowledge to perform a specific task using a learning algorithm like those described for the back-propagation and SOM networks. This ability to learn is a very powerful tool in domains where the acquisition of knowledge for a particular task is either costly, time-consuming or problematic to acquire. This provides system developers with a method of developing an intelligent automated system without the need to implement a full knowledge engineering approach to acquire the knowledge. This saves both on development time and excludes the need for a highly skilled expert in the domain of interest.

Another benefit of the ANN approach is the robustness of the learned functions which makes them robust to errors in the data. This is an especially powerful attribute when the network is used to interpret data taken from sensors, such as those in machine condition monitoring applications. This type of data will likely contain noise which may increase or decrease throughout the lifetime of a particular piece of plant. Complex real world domains will also have high degrees of non-linearity due to the complexity of the processes encountered in such applications. The high performance of ANNs in learning non-linear functions makes them particularly useful in such complex domains.

As well as the speed in which knowledge of a particular domain can be acquired through automated learning algorithms, ANNs are able to assess input data at very fast processing speeds. This makes the approach useful in applications where close to real time assessments must be made in order to implement actions in a timely fashion. An example of where this attribute finds particular favour is in control or protection problems where actions must be made quickly in response to the inputs and operating conditions of the network.

ANNs require the availability of a suitable data set which is representative of the problem being addressed. Supervised learning, which is required by back-propagation networks, means that this data must be appropriately labelled to provide performance feedback for the learning algorithm. The training data doesn't have to be labelled for unsupervised learning approaches used in SOM networks since, the similarities in the data are being derived as opposed to grouping the examples into pre-defined categories. It is therefore imperative that, regardless of the specific technique chosen, there must exist a suitable data set which is representative of the problem domain. As larger, faster and cheaper storage devices become available, adequate training data is becoming available for a significant number of applications. However, problems arise where access to suitable data is not permitted due to poor data storage approaches or the rarity of particular events.

A final consideration must be paid to the form in which the knowledge is represented. The knowledge acquired by an ANN is implicit in the interconnecting weights between neurons, inputs and outputs. Although this knowledge representation allows ANNs to quickly and accurately process noisy data sets, it doesn't provide any explanation or rationale which is easily understood by a system operator. The lack of explanation provided by ANNs means that user confidence can be lost, or at the very least reduced, when the network performs poorly.

3.3.4 Applied Artificial Neural Networks

ANNs have been applied to a wide range of condition monitoring and fault diagnosis tasks. One area is the condition monitoring of nuclear power plants [Embrechts &

Benedek, 2004] and [Steele et al, 2003]. In [Embrechts & Benedek, 2004] a back-propagation ANN is trained to determine if the monitored equipment is exhibiting signs of one of twenty pre-defined faults. Fuel grab load data from the refueling of nuclear reactors is analysed in [Steele et al, 2003] using ANNs. Features are derived from load trace data and used as inputs to an ANN for classification.

ANNs have also found application in the condition monitoring of power transformers [McArthur et al, 2004], [Wang et al, 2000], [Wang et al, 1998] and [Booth & McDonald, 1998]. The multi-agent system in [McArthur et al, 2004] uses a back-propagation ANN to detect and classify transformer faults from a feature vector made up of statistical measures derived from Ultra High Frequency (UHF) sensor data. A similar ANN is used in parallel with an Expert System in [Wang et al, 2000] and [Wang et al, 1998] to detect abnormal transformer behaviour through diagnosis of various pre-defined faults. Each fault type is assigned a dedicated ANN which applies oil sample information as inputs to the network. In [Booth & McDonald, 1998] a back-propagation ANN is trained to predict transformer vibration behaviour from thermal and current inputs. This provided a useful technique for comparing "healthy" modelled behaviour with actual vibration data captured from the test transformer. The paper also reported a SOM network which was successfully trained to classify instances of "healthy" and "unhealthy" transformer behaviour.

A SOM network was also used in [Wu & Chow, 2004] to detect faults generated by a three-phase induction motor test rig. The network used feature vectors derived from the frequency distribution of the vibration data and showed a high success rate at detecting mechanical and electrical fault types as well as normal behaviour. A similar approach is used in [Li et al, 2000], in that various frequency based features are extracted from the bearing time-series vibration data and fed into a back-propagation neural network. This technique was shown to perform well at diagnosing faults on test apparatus within the laboratory.

There were two issues associated with the turbine generator condition monitoring application which meant that ANNs were unsuitable. The first was that there was no training data in an electronic format which could be used to train a network. The data associated with the alarms generated for the Beran system were all stored in paper format and any digital data on the Beran system archive was still in its raw data format. It would have been possible to transfer the paper records to a digital format and transform the raw signal data into a format to be used as training data but the effort and resources required to do this were seen as being prohibitive. The other issue was the fact that the British Energy condition monitoring experts had specifically requested for there to be some form of assessment rationale so that any assessment could be verified. Therefore, ANNs which provide little or no explanation of their assessment, were not suited to this particular task.

3.3.5 **Further Reading**

For a more complete and in depth discussion on ANNs including its origins, seminal applications outside of the power systems domain and the varying algorithms adopted, the reader is guided to the following texts [Haykin, 1999] and [Mitchell, 1997].

3.4 Model-Based Diagnosis

Some of the earliest examples of MBD began to emerge in the 1970's, with the 1980's witnessing considerable growth in the technology [Davis & Hamscher, 1988]. MBD attempts to diagnose faulty behaviour using models of the domain of interest such as structural, functional, fault etc. Initially it was developed as a potential solution to some of the limitations which Expert Systems imposed. Central to the Expert System approach is the use of the empirical associations which associate observed symptoms with underlying faults in the system. The relationships were in most cases built up through experience of the device rather than knowledge of the structure or behaviour. This meant that Expert Systems were inherently device dependent; that is, a new rule set was required for each new device. This issue became especially prevalent in areas such as electronics where the rate at which new devices were being designed and manufactured created time constraints for collecting the required knowledge to develop such systems.

MBD is, by contrast, device independent due to the fact that it works from models of normal behaviour typically available at the design stage. Given a model of normal behaviour of the device, work can begin on diagnosing faults in a relatively short period of time. The advantages brought by such an approach are that where suitable models of the device of interest are available then there can be significant cost savings in the effort required to develop such a diagnostic device. MBD approaches are also more likely to diagnose faults within a system which have not been encountered before, as opposed to expert systems whose empirical associations rely on previous experience of encountered problems.

3.4.1 Model-Based Diagnosis Approach

The basic MBD paradigm is one of observation and prediction as shown in figure 3.9 from [Davis & Hamscher, 1988]. The actual device is typically some physical system whose behaviour can be observed. In addition to this, a model of the actual device can be used to make predictions about the expected correct behaviour. Any discrepancies between the observed behaviour (what the device is actually doing) and the predicted behaviour (what the device is supposed to be doing) are treated as an interesting event. A fundamental assumption in this approach is that the model accurately simulates correct behaviour of the device, which therefore implies that any deviation from this predicted correct behaviour is deemed as a fault.

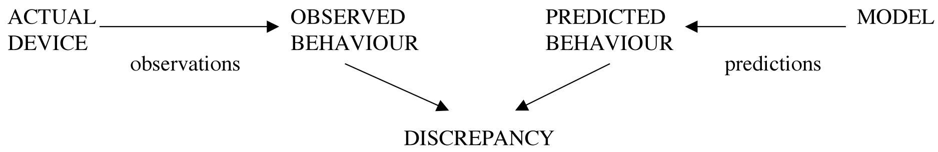


Figure 3.9: Basic MBD paradigm where the basis of diagnosis is the interaction of observation and prediction

On location of a discrepancy the task is to determine which component or components in the model could have failed in a way that accounts for all of the observed discrepancies. This is achieved by the following 3 stage approach:

- Hypothesis generation
- Testing
- Discrimination

Hypothesis generation is concerned with reasoning from a symptom to a collection of components whose misbehaviour could have caused that symptom. The next stage of the process is to test the candidate set to determine which components can account for all observations of device behaviour. The final stage of discrimination attempts to deduce what remaining candidates in the set are the most likely cause of failure through testing. The following sections will now explore some of the most common approaches in MBD employed to achieve the three aforementioned stages. A standard textbook example taken from [Davis & Hamscher, 1988] has been used throughout the following sections to assist in explaining each of these approaches.

3.4.1.1 Hypothesis Generation

The fundamental task here is that given a discrepancy, determine which components could have misbehaved in a way to produce that discrepancy. The simplest approach to achieving this would be to use a generator which simply nominates all of the components within the device. Figure 3.10 shows graphically a simple function.

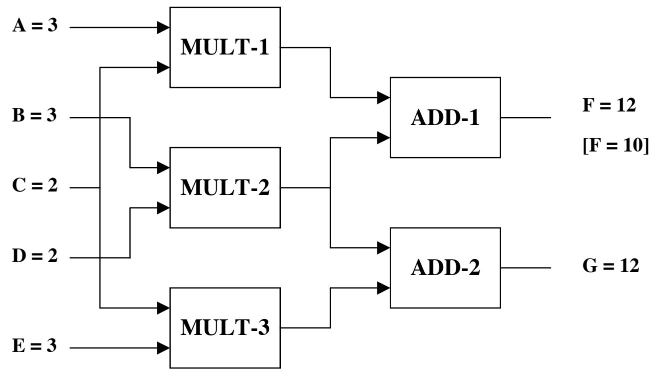


Figure 3.10: MBD example where potential candidates MULT-1, MULT-2, MULT-3, ADD-1 and ADD-2 could account for the discrepancy at F.

There are five numeric inputs A, B, C, D and E, three multipliers MULT-1, MULT-2 and MULT-3, two adders ADD-1 and ADD-2, and 2 outputs F and G. MULT-1 multiplies inputs A and C together, MULT-2, B and D, and MULT-3, C and E. Output F is calculated by adding together the output of MULT-1 and MULT-2, and output G is derived by adding together the output of MULT-2 and MULT-3. The outputs which are not bracketed are the predicted outputs whereas the outputs in brackets are those observed. It can be seen from the example in figure 3.10 that the output value F observed differs from the expected value. Therefore, a hypothesis approach, which simply nominates all of the components within the function, would identify components MULT-1, MULT-2, MULT-3, ADD-1 and ADD-2 as being potentially faulty from the device depicted in figure 3.10.

It should be apparent that this approach is not very intelligent but it can be improved by implementing a few simple rules to the search. These are:

- To be a suspect, a component must have been connected to the discrepancy.
- Only consider components upstream of the discrepancy as potential candidates.
- Only consider inputs which influence the output of a device to avoid following unnecessary inputs upstream.

- Information from more than one discrepancy can be used to further constrain the suspect generation.

The first rule would have no affect on the candidates generated from the previous example, since all components are connected to the discrepancy. The second rule would reduce the hypothesis set to MULT-1, MULT-2 and ADD-1. The third rule would not reduce the hypothesis set any further in the above example but could be used in other situations to effectively reason about a components behaviour to determine irrelevant inputs. The fourth rule is useful in cases where more than one discrepancy exists as shown in figure 3.11. Discrepancy F yields the candidates MULT-1, MULT-2 and ADD-1, whereas discrepancy G yields MULT-2, MULT-3 and ADD-2. Assuming a single point of failure, the hypothesis set could be reduced to the intersection point of both candidate sets i.e. MULT-2.

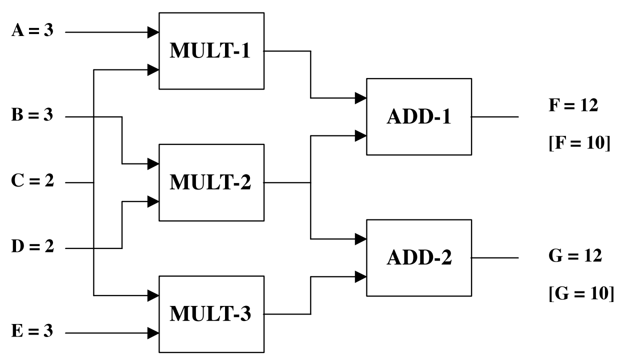


Figure 3.11: MBD example where the single potential candidate MULT-2 could account for both discrepancies at F and G assuming a single point of failure.

3.4.1.2 Hypothesis Testing

Once all of the candidate components in the system have been identified, it is required to test each hypothesis to determine if it can account for each of the observations made about the device. The simplest method to achieve this is to simulate in turn all of the ways in which each component in the hypothesis can malfunction using the original observed inputs. If the overall modelled behaviour is inconsistent with the observations, then the hypothesis can be discarded, whereas hypotheses which match with the observations can be retained. Another approach for hypothesis testing is constraint suspension. The basic idea behind constraint suspension is to test each component in the system to determine if any of them can account for the inconsistency. This is achieved by modelling each component as a set of constraints such as that given in figure 3.12 for an adder. Here the behaviour of an adder is represented by a set of expressions which capture the relationships between the values on the terminals of the component.

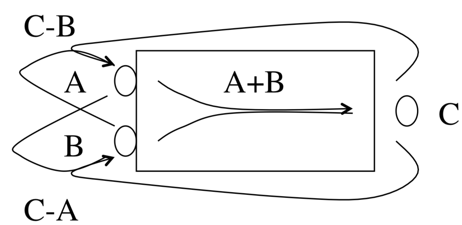


Figure 3.12: The behaviour description of an adder

Constraint suspension achieves hypothesis testing by removing the constraints for the suspect and leaving in place all remaining constraints. The observed values are then placed into the reduced constraints network. If no inconsistency is met during this simulation this would imply that the current suspect is consistent with all of the observations and is therefore a suspect hypothesis. If the network is still inconsistent with the constraint suspended then the suspect can be eliminated. One big advantage of this approach is that no assumptions are made regarding how any of the candidates could fail. It is in this sense that model-based approaches using a model of correct behaviour covers a broader class of faults than traditional techniques which pre-specify the modes of failure.

3.4.1.3 Hypothesis Discrimination

The next stage in the MBD process is that of hypothesis discrimination which attempts to distinguish between the multiple hypothesis that survive the hypothesis testing. This stage essentially entails gathering new information on the behaviour of the device using one of two approaches. The first approach is referred to as probing which involves making additional observations and the second is referred to as testing which requires that the inputs of the actual device are changed to obtain new observations. In both cases the goal is to gain the most information at the least cost.

3.4.1.3.1. Probing

The simplest approach to probing is to use the structural information to generate the set of all possible locations and pick any places which have not been measured yet. This basic approach can be refined by starting at the discrepancy and following it upstream to a component whose output is incorrect but its input is correct. This more advanced approach is referred to as the guided probe. For example the discrepancy in figure 3.13, where the value 5 is predicted as the output but 3 is observed instead, would be probed first at terminals A and Z since if these are measured to have their predicted values then this would imply that MAX-1 must be faulty. If Z has any number other than 5 then we probe upstream at both B and Y to see if they are 1 or 4 respectively until we find the culprit. The guided probe technique can be extended to use information about component behaviour to reduce the probes required.

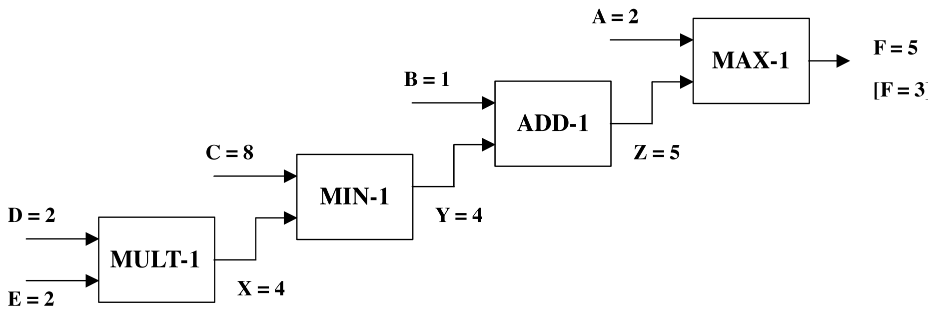


Figure 3.13: The guided probe approach to hypothesis discrimination.

The guided probe is a simple and effective approach but is linear in its search. This approach can however be refined further to produce a more efficient search. In the current example it should be apparent that the most effective position for the first probe would be at Y. If the value does not match the predicted one then half of the components can be eliminated from the search, that is, those upstream of the probe. The process of cutting the search space in half can be continued at each step producing the traditional binary search. In some cases there may be several places which appear equally informative. It is sometimes possible in instances such as these to eliminate candidates for probe points when there is information on failure probabilities of the considered components. This information can be used to test the component with the greatest chance of failure than those that are least likely to fail.

3.4.1.3.2. Testing

Testing is the second approach to hypothesis discrimination. This approach uses new inputs to the actual device to create further observations which can be used to gain more information on the behaviour of the device. A valid hypothesis should be consistent with the new observations created by the new test inputs. As with the probing approach the difficulty is to find the test which will provide the maximum amount of information for the discrimination process. If the set of tests is pre-set at a finite amount then the optimal approach is to select the test which will split the set in half. If the set of tests possible is infinite then the goal is to develop tests which will test each component.

3.4.2 Practical Issues for Model-Based Diagnosis Applications

MBD is a symbolic AI approach which utilises models of the domain of interest as its source of knowledge, as opposed to the empirical heuristic associations used by the Expert System approach. This makes MBD an attractive option in instances where an adequate model of the domain of interest exists since little or no effort is required to acquire the knowledge base. This is in contrast to the time consuming and expensive knowledge engineering process required to develop the knowledge base for Expert Systems. This is particularly useful in applications where devices become obsolete in short spaces of time but models are readily available. The short useful lifetime of such devices make Expert Systems prohibitive due to the length of time required to acquire the necessary knowledge since the device could become obsolete before the knowledge engineering approach is completed. With MBD however, a system can be developed relatively quickly given the access to adequate models. A good example of this arises in the electronics industry, where continuous developments in the technology mean that devices have increasingly shorter lifetimes but access to logical models is facilitated as soon as the product is developed. The use of models, which are built both on formal logical foundations, and correct expected behaviour, as opposed to known faulty behaviour, make it possible for such systems to diagnose novel behaviours or faults.

There are some limitations to the MBD approach. One limitation arises in domains which do not have access to an adequate model of the problem being addressed. In some of these situations it is possible to develop a model but this would only be feasible where the time and expense involved is merited by the application. It can also be the case that some domains are so complex or lack a clear logical foundation that the development of a useful model is not possible. Therefore the difficulty in acquiring expert knowledge is replaced by the difficulty in acquiring or developing a model. A further limitation of MBD systems is one which is shared with the Expert System approach. The MBD approach has no method of assessing its performance on a particular task and feeding this back into the system to make improvements to the model. Therefore, MBD is unable to learn knew knowledge as is the case for Expert Systems.

3.4.3 Applied Model-Based Diagnosis

There have been very few examples of MBD being applied to complex power system problems largely due to the difficulty in developing adequate models of the domain. However the most notable example of such a system being applied to turbine generator condition monitoring is TIGER [Trave-Massuyes & Milne, 1997]. The system consists of a rule-based and a model based diagnosis module. The model based diagnosis module CA-EN is a qualitative technique which seeks to compare the expected system behaviour with the observed behaviour in order to locate discrepancies between both signals. Knowledge is represented within the causal models either empirically as cause and effect relationships or by equations which represent the physics of the underlying system behaviour. When a discrepancy between the expected behaviour and the observed behaviour is detected over a predefined period then the fault diagnosis element of the MBD module is initiated to determine a component or set of components which can explain the discrepancy. TIGER has received much attention from industrial parties and has been successful in diagnosing various faults over numerous gas-turbines as reported in [Milne et al, 2001].

An MBD toolset based on the GDE [Davis & Hamscher, 1988] is described in [Davidson et al, 2003]. This toolset is aimed specifically at utilising powers system protection simulation models which are commonly used by protection engineers to design and maintain protection schemes on transmission and distribution networks. The MBD toolset is able to capture a model from various protection simulation packages available. The model is composed of two document types. The first document describes the overall structure and connectivity of the model components. The second document models the behaviour of the components within the model structure such as the inputs required and how to simulate the behaviour of the component given these inputs. The diagnostic element of the toolset uses a discrepancy based approach to identify candidate components which have misbehaved during a protection operation. A later version of the system described in [Davidson et al, 2005] adds abductive MBR methods to the toolset to determine how a component has failed once it has been identified by the consistency based approach. The abductive based approach relies on the availability of fault models which describe failure modes for each component based on previous experience. The advantage of this toolset is that it allows existing models of a protection system, which have been produced for the purpose of protection setting validation/grading and system design, to then be utilised for diagnostic purposes. MBD is particularly suited to this domain since existing models of protection systems already exist for many applications. The authors do point out that the diagnostic results produced by the toolset are dependent on the quality of the model and its suitability to the task at hand.

The application of MBD for the turbine generator condition monitoring application is limited since there are no detailed models of normal behaviour for the 660MW turbine generator sets within British Energy. One option explored was the development of such a set of models which detailed the normal behaviour of the turbine generator sets. British Energy had previously invested some resources in an attempt to develop models associated with certain functional areas of the turbines. To model only specific elements of the sets was demanding on resources plus the results drawn from the exercises were not as beneficial or consistent as had been initially hoped for. Given the limited resources at disposal for the turbine generator project it was felt that the scale and complexity of the task to develop such models was too prohibitive. Therefore MBD was rejected as an approach for developing an automated system for the turbine generator condition monitoring project.

3.4.4 Further Reading

No comprehensive textbooks currently exist for the area of MBD which are equivalent to those that can be found for ANNs and Expert Systems. A more complete discussion on the approaches covered in this section can be found in [Davis & Hamscher, 1988] and the collection of papers for which the aforementioned paper was included within [Hamscher et al, 1992] gives a more detailed overview of the area of MBD.

3.5 Condition Monitoring Approach Selection

From the three AI approaches so far described in this chapter, one had to be selected to develop an automated system which was capable of assessing the alarms generated by the Beran system. The British Energy condition monitoring experts had expressed that the system produced should provide some explanation of its assessment so that the results could be verified. As explained in section 3.3.3, ANNs are poor at providing explanation of their assessments since the knowledge utilised by this approach is difficult for humans to interpret. Additionally, the ANN approach requires training data to develop such a system which was not available for the turbine generator condition monitoring project. Therefore the ANN approach was rejected as an approach for the automated system. The MBD approach was also rejected since there was no existing model of normal behaviour of the 660MW turbine generator sets within British Energy. Additionally the development of such a model was viewed as being too large a task for the resources at the project's disposal. The chosen solution for the automated system was the Expert System approach. Expert Systems have been shown to provide good explanation through the utilisation of the heuristic knowledge used to derive assessment conclusions as described in section 3.2.6. Additionally, the knowledge required to develop the Expert System was available from the British Energy Experts. A more detailed explanation of the technique selection is given in section 4.2.

Another facet of the turbine generator condition monitoring project was the development of a learning module which was designed to assist in capturing the knowledge required by the Expert System. The learning module was developed using an ML approach. The area of ML is reviewed in the remaining sections in this chapter.

3.6 Machine Learning Taxonomy

A complete literature review of the area of ML is outwith the scope of this thesis. The second half of this chapter does however aim to give an overview of the area of ML which will assist in explaining the choice of technique for the learning module for the turbine generator condition monitoring project. An example of a high-level ML taxonomy is given in figure 3.14. It can be seen that the area of ML can be divided into four broad categories. These are:

- Symbolic based approaches
- Instance based learning
- Connectionist networks
- Statistical/probabilistic approaches

Symbolic based approaches are, as the name implies, techniques which utilise symbolic data like that used in rule-based systems. The two categories of symbolic approaches are instance based learning and analytical learning. Instance based learning techniques are data driven techniques which use the statistical properties of data sets to extract commonalities which can lead to rule-based relationships. Analytical methods are knowledge based techniques which use background knowledge of the particular area of interest in conjunction with a small amount of training data to derive heuristic expressions. Instance based learning approaches are commonly referred to as CBR systems. CBR systems do not derive explicit or implicit knowledge directly to be used in some form of knowledge base. Instead they store examples in what is referred to as a case base and use a similarity measurement to determine which case or cases match closest to the current example under investigation. An area of machine learning, which has already been covered in detail in section 3.3, is ANNs or connectionist systems. These approaches are again data driven in that they use large data sets to update the statistical knowledge implicit within the network neurons to improve the system performance. The final category encompasses a broad range of techniques which utilise the statistical properties within, mostly, large data sets to infer knowledge. Two approaches which fall within this category, and are discussed in the second half of this chapter, are Hidden Markov Models and Bayesian learning techniques.

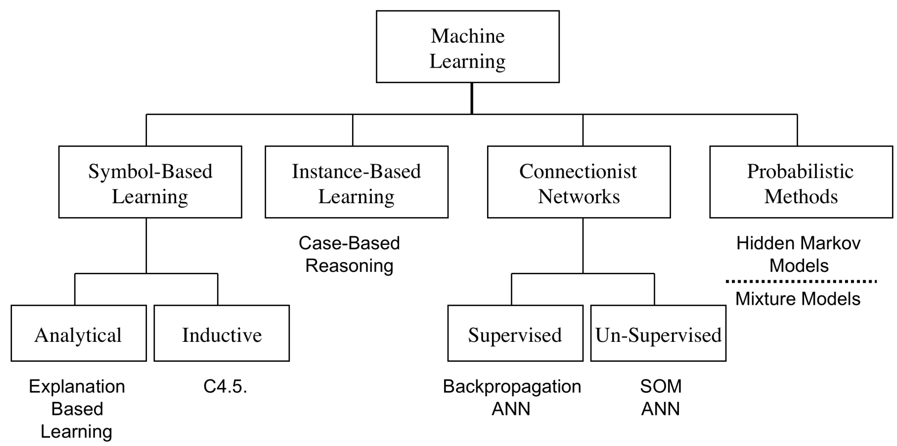


Figure 3.14: High level Machine Learning taxonomy.

The taxonomy given in figure 3.14 is by no means the definitive thinking on how the area of machine learning should be split. For example one line of thinking may divide the approaches into data driven and analytical. It could be argued that there is considerable overlap in the categories within the taxonomy. For example it could be argued that inductive symbolic methods along with connectionist networks should come under the statistical/probabilistic heading since they derive their knowledge from the statistical information contained within large training sets. Therefore the taxonomy given in figure 3.14 is a method of breaking down the vast area of machine learning. The remainder of the chapter will now give an overview of each of the categories outlined in the taxonomy which have not already been covered in previous sections and will describe some of the most common techniques within each particular area.

3.7 Symbol-Based Learning

Symbol-based learning techniques are primarily concerned with deriving the type of explicit symbol based knowledge such as that found in Expert Systems i.e. heuristic rule based expressions. The two primary approaches to deriving this type of knowledge are analytical or inductive based learning techniques. Analytical techniques utilise background information such as domain knowledge alongside a single or few training examples to derive heuristic rule based expressions. This approach can be thought of as knowledge driven, due to the use of some background knowledge of the area of interest. Inductive rule-based approaches use no background information on the domain but instead search for similarities within the data set to derive empirical associations. This type of approach requires large enough data sets to determine similar features between training data which is of a similar type. Therefore these techniques are effectively utilising the statistical distribution of the data set to derive the knowledge. This type of approach is commonly referred to as a data driven approach. Common approaches to both types of learning are explained in the following sections.

3.7.1 Analytical Learning

The two most common and widely researched analytical symbolic ML approaches are Explanation Based Learning (EBL) [DeJong & Mooney, 1986] and Explanation Based Generalisation (EBG) [Mitchell et al, 1986].

The EBG problem can be summarised as follows:

Given:

- Goal Concept: A concept definition describing the concept to be learned.
- Training Example: An example of the goal concept.
- Domain Theory: A set of rules and facts to be used in explaining how the training example is an example of the goal concept.
- Operationality Criterion: specifies the form in which the learned concept definition must be expressed.

Determine :

- A generalisation of the training example that is a sufficient concept definition for the goal concept and that satisfies the operationality criterion.

The EBG problem definition means that a system employing an EBG approach would be able to derive an expression which, in most condition monitoring cases, would be a diagnostic rule or heuristic. For example the turbine generator condition monitoring application may require a rule which is able to diagnose a rotor out of balance fault. Rotor out of balance would then become the goal concept. To derive this rule, the EBG approach would require an example of a rotor out of balance fault. This would be the training example in terms of the EBG problem definition. Initially this training example will contain facts which are not directly associated with the fact that it is an example of an out of balance fault. The EBG approach will use existing background knowledge on the domain of interest to sort out the information associated with the training example, so that only features relevant are used to construct the rule. This background knowledge relates to the domain theory in the EBG problem definition and would consist of knowledge of out of balance rotor faults for the derivation of the diagnostic rule. It is this use of background knowledge to determine the relevant features in the training example which encompasses the learning element of the EBG approach. The EBG problem definition also specifies that the expression learned must be expressed in a suitable format. This format is known as the operationality criterion. If, for example, the purpose of learning an expression for a rotor out of balance fault was to use this within an on-line diagnostic system, then the expression would have to contain terms which the on-line system would have access to. Therefore, if the on-line system never had access to temperature data of the turbine, temperature data would not be used in the operationality criterion.

The EBG approach used to solve the problem outlined above is:

1 Explain: Construct an explanation in terms of the domain theory that proves how the training example satisfies the goal concept definition.
2 Generalise: Determine a set of sufficient conditions under which the explanation structure holds, stated in terms that satisfy the operationality criterion.

Therefore, the EBG approach firstly proves the training example to be an instance of the goal concept by using the domain knowledge to prove it. The background knowledge would therefore have to logically prove that the training example fitted the criteria of that required to be an example of the concept definition. The logical explanation which proves that the example fulfils the goal concept criteria can then be transformed into an expression which can be used to identify other examples of the goal concept. It is the derivation of this expression which the second stage of the EBG approach is concerned with. Following the first stage the explanation is particular to the training example used to derive it. Therefore this explanation has to be generalised so that other examples which are of the same goal concept but not identical can be recognised by the derived expression.

An example will be used to show how EBG uses the above approach to derive rule based heuristics. The example is based on the one given in [Mitchell, 1997]. A problem definition is given in figure 3.15.

Given:

• Goal Concept: Pair of objects <x, y> such that SAFE-TO-STACK (x, y)

• Training Example:

ON (OBJ1, OBJ2)
TYPE(OBJ1, BOX)
TYPE(OBJ2, ENDTABLE)
COLOUR (OBJ1, RED)
COLOUR (OBJ2, BLUE)
VOLUME (OBJ1, 1)
DENSITY (OBJ1, 0.1)
....

• Domain Theory:

SAFETOSTACK(p1, p2) $\leftarrow$ $\neg$FRAGILE(p2)
SAFETOSTACK(p1, p2) $\leftarrow$ LIGHTER(p1, p2)
WEIGHT (p1, w1) $\leftarrow$ VOLUME (p1, v1) $\wedge$ DENSITY (p1, d1) $\wedge$ EQUAL(w1, TIMES(v1, d1))
LIGHTER (p1, p2) $\leftarrow$ WEIGHT (p1, w1) $\wedge$ WEIGHT (p2, w2) $\wedge$ LESSTHAN (w1, w2)
WEIGHT (p1, 5) (default) $\leftarrow$ TYPE (p1, ENDTABLE)
....

• Operationality Criterion: The concept definition must be expressed in terms of the predicates used to describe examples (e.g. VOLUME, COLOUR, DENSITY) or other selected, easily evaluated, predicates from the domain theory (e.g. LESS)

Determine:

• A generalisation of training example that is a sufficient concept definition for the goal concept that satisfies the operationality criterion.

Figure 3.15: Example of a problem definition for the EBG safe to stack problem.

The problem definition in figure 3.15 indicates that the goal concept is to derive an expression to identify examples of where one object x is safe to stack on top of another object y which is denoted as SAFE-TO-STACK(x, y). The operationality criterion specifies that the expression derived by the EBG approach must be expressed in terms of the predicates given in the training example such as COLOUR, VOLUME and DENSITY. This expression is derived using the domain theory, which as shown in figure 3.15 contains knowledge such as how to determine the weight of an object through its volume and density, and the process of generalisation. The first step of the EBG approach is to construct an explanation of how the training example satisfies the goal concept. Essentially the domain theory is used to distinguish from the training data what pieces of information are relevant to the goal concept SAFE-TO-STACK(x, y). The explanation for this problem definition is shown in figure 3.16.

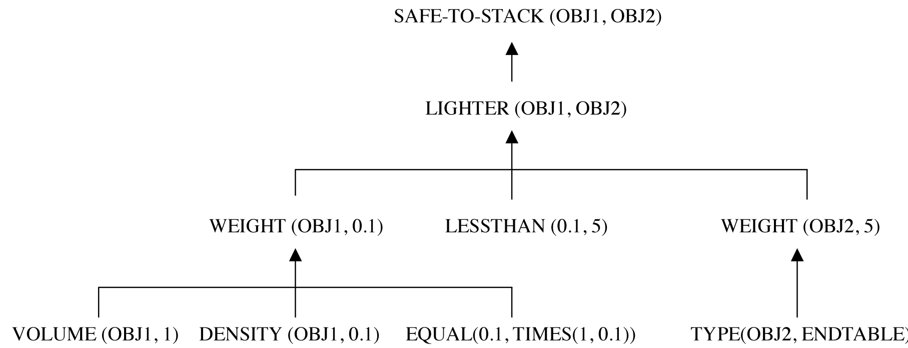


Figure 3.16: Explanation composed by the EBG approach to indicate how the current example fits the safe to stack criteria.

The explanation structure shows that OBJ1 and OBJ2 fulfil the goal concept of SAFE-TO-STACK(x, y). The weights of each object are determined from the Domain theory given in figure 3.15 through the use of the WEIGHT(p1, w1) expression for OBJ1 and the WEIGHT(p1, 5) (default) expression for OBJ2. From this it is inferred that OBJ1 is lighter than OBJ2 through the LIGHTER(p1, p2) expression given in the domain knowledge in figure 3.15. Therefore it is inferred that it is safe to stack OBJ1 onto OBJ2 using the expression given the domain theory in figure 3.15, that is, SAFETOSTACK(p1, p2) ← LIGHTER(p1, p2). Note that the explanation structure has been constructed so that each of its branches terminates in an expression that satisfies the operationality criterion given in figure 3.15.

While the first step in the EBG process isolates the relevant features in the training example it does not determine the generalised constraints within the explanation structure. Although the feature VOLUME (OBJ1, 1) given in figure 3.16 is relevant to explaining how the present training example fulfils the goal concept it does not contain general enough constraints which would encompass every training example of the goal concept. For example, consider a second training example which is identical to the one given in figure 3.15 except that VOLUME(OBJ1, 1) is replaced for VOLUME(OBJ1, 2). It should be apparent that, in this new example, it is still safe to stack OBJ1 on top of OBJ2 since OBJ1 is still lighter than OBJ2, that is 0.2 is less than 5. However the explanation developed by the EBG algorithm so far given in figure 3.16 would be unable to deduce this because it is only capable of recognising examples which contain the attributes VOLUME(OBJ1, 1), DENSITY(OBJ1, 0.1) and TYPE(OBJ2, ENDTABLE) as being safe to stack OBJ1 on top of OBJ2. The EBG approach would be able to prove that this second example is an instance of safe to stack by proving it again using the domain theory. This is undesirable since it is time consuming, especially for problems where the explanation is large. It is therefore desirable to generalise the explanation given in figure 3.16 so that similar examples of safe to stack, which are not exactly the same as the training example given in figure 3.15, can be classified.

This process of generalisation is performed by the second stage of the EBG process. This is achieved by using goal regression [Mitchell et al, 1986], which determines sufficient conditions under which the rule can infer the formula. The first stage of the regression process (R1) as shown in figure 3.17 starts with taking the goal concept which in our example is SAFETOSTACK(x, y). The regression approach then takes the expression from the domain knowledge which was used in the explanation given in figure 3.16. This expression is SAFETOSTACK(p1, p2) ← LIGHTER(p1, p2) as shown in figure 3.17. To translate SAFETOSTACK(p1, p2) into the form of our goal concept SAFETOSTACK(x, y) we need to make the substitutions x = p1 and y = p2. Therefore LIGHTER(p1, p2) is put into the form required by the goal concept LIGHTER(x, y) by making these substitutions as shown in figure 3.17.

Goal Concept:	*SAFE-TO-STACK (x,y)*	
R1:	SAFE-TO-STACK (p1, p2)	(x/p1, y/p2)
	↑	
	LIGHTER (p1, p2)	
	LIGHTER (x, y)	

Figure 3.17: Regression of SAFETOSTACK(x, y) expression from the derived explanation

The next stage in the regression process (R2) is to regress the LIGHTER(x, y) attribute as shown in figure 3.18. The expression used to develop the explanation in figure 3.16, LIGHTER (p1, p2) ← WEIGHT (p1, w1) ∧ WEIGHT (p2, w2) ∧ LESSTHAN (w1, w2), is applied. Again the substitutions x = p1 and y = p2 are again used to translate each attribute into the form required for the goal concept expression SAFETOSTACK(x, y) as shown in figure 3.18.

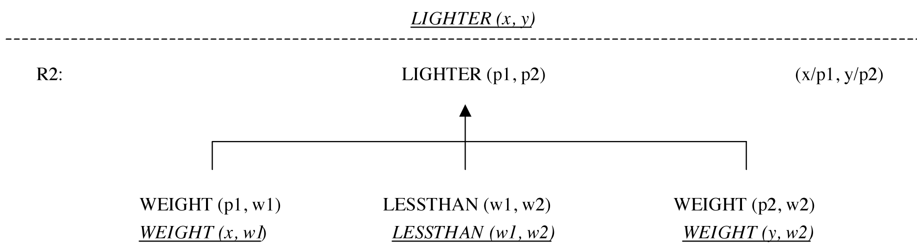


Figure 3.18: Regression of LIGHTER(x, y) expression from the derived explanation

The next stage in the regression process (R3) is to regress the WEIGHT(x, w1) expression as shown in figure 3.19. The expression used in generating the explanation in figure 3.16, WEIGHT (p1, w1) ← VOLUME (p1, v1) ∧ DENSITY (p1, d1) ∧ EQUAL(w1, TIMES(v1, d1)), is applied. The substitutions x = p1 and w1 = w1 have to be applied to each attribute so that they are in the form required for the goal concept SAFETOSTACK(x, y) as shown in figure 3.19.

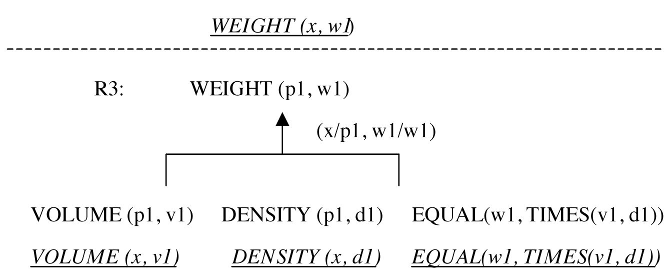


Figure 3.19: Regression of WEIGHT(x, w1) expression from the derived explanation

The next stage in the regression process (R4) is to regress the WEIGHT(y, w2) attribute as shown in figure 3.20. The expression used to generate the explanation in figure 3.16, WEIGHT (p1, 5) (default) ← TYPE (p1, ENDTABLE), is applied. The substitutions y = p1 and w2 = 5 have to be applied to the attributes so that they are in the form required for the goal concept SAFETOSTACK(x, y) as shown in figure 3.20.

WEIGHT (y, w2)

R4: WEIGHT (p1, 5)

$\uparrow$ (y/p1, w2/5)

TYPE(p1, ENDTABLE)

TYPE(y, ENDTABLE)

Figure 3.20: Regression of WEIGHT(y, w2) expression from the derived explanation

Finally the LESSTHAN(w1, w2) attribute has the w2 = 5 applied to it to derive LESSTHAN(w1, 5). The full regression process is summarised in figure 3.21. The fully regressed expression derived by the EBG approach for the goal concept SAFE-TO-STACK (x, y) is derived by conjugating all of the attributes at the bottom of the regressed explanation tree. Therefore the final expression is VOLUME(x, v1) $\wedge$ DENSITY(x, d1) $\wedge$ EQUAL(w1, TIMES(v1, d1)) $\wedge$ LESSTHAN(w1, 5) $\wedge$ TYPE(y, ENDTABLE). It should be apparent that a second example identical to that given in figure 3.15, except that VOLUME(OBJ1, 1) is replaced for VOLUME(OBJ1, 2), can now be classified as being safe to stack using the generalised expression derived by the EBG approach. There is no requirement to prove, using the domain theory, that this second example is an example of safe to stack, since a generalised expression now exists which is capable of classifying it and any other examples of safe to stack which contain the required attributes and fulfil the necessary criteria. The EBG approach has therefore learned a generalised expression using the single training example and the domain theory.

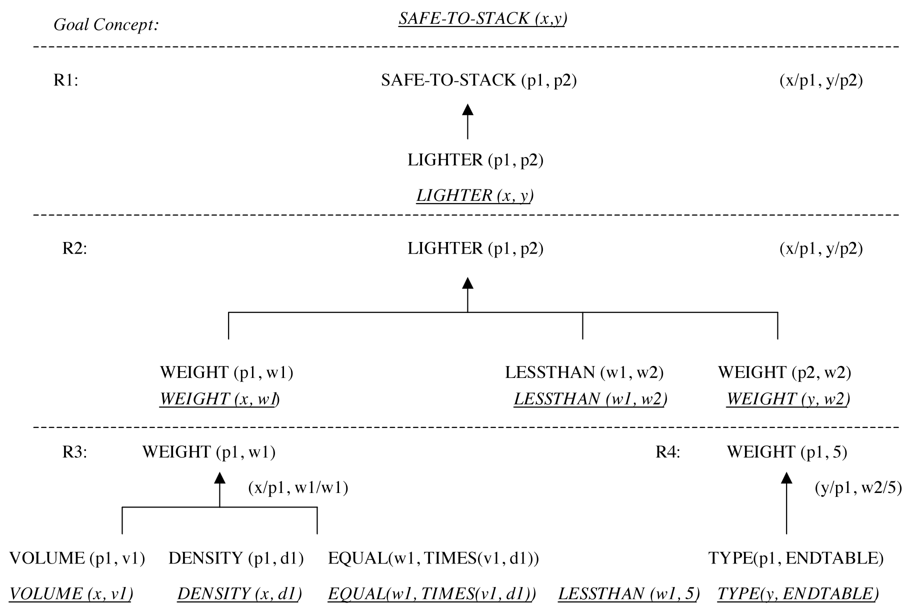


Figure 3.21: A reduced form of goal regression is used to generalise the constraints of the expression so that other examples of the concept can be classified.

An alternative approach to analytical symbolic learning is given in [DeJong & Mooney, 1986]. This approach is called EBL and is in many ways similar to the EBG approach previously outlined. The authors outline what they see as being weaknesses in the EBG approach. The main weakness identified is that the goal regression approach used in the EBG approach is not general enough. This is because only the variables within the predicates are generalised and not the predicates themselves. For example, in the safe to stack example outlined above, the authors argue that the derived expression would be unable to classify an example where the object's volume and density is not given, but it is stated that the object is an ashtray and that the ashtray weighs only 1. It should be apparent that the object still weighs less than the end table and it should therefore be safe to stack. Therefore the authors argue that an approach which is able to generalise the predicates to encompass this would be more powerful than that proposed in [Mitchell et al, 1986]. The authors also argue that the use of an operationality criterion is limiting since the criterion might not always be operational. This argument is true if it is the case that the information relating to the example is captured in a non-structured way, say through natural language. For example, two different people may describe the same scenario in two different ways. This would require two different operationality criterion if expressions had to be learned to recognise that scenario but from two different descriptions. To alleviate these problems the authors propose generalising the derived expression further using schemata, which allows sections of knowledge which are related to be grouped together and utilised within the expression. For example, if some derived expression contained the predicate GUN, a schemata might be available called WEAPON which contains the predicate GUN along with other types of weapon such as KNIFE etc. The predicate GUN could therefore be substituted for the more general predicate WEAPON, which would make the derived expression more general. To incorporate the use of schemata the authors provide a different approach to deriving the expression than that employed for the EBG approach in [Mitchell et al, 1986]. One of the differences is the use of goal regression has to be excluded in favour of an approach which employs schemata to generalise the predicates as well as the variables. The authors also argue that the operationality criteria could be eliminated through the use of schemata. This is because the generalised expression could always be used to classify an example which uses predicates which are more specific by referring to the schemata therefore eliminating the need to define an operationality criterion.

3.7.2 Applied Analytical Learning

One of the few examples where an analytical learning approach has found an application in fault diagnosis is in [Kobayashi & Nakamura, 1991]. Here the EBG approach is used to derive diagnostic heuristics for a cigarette making machine. Instead of using a formal first order logic knowledge representation, as traditionally used in EBG approaches, the authors used causal fault models to construct fault explanations. The constraint imposed on EBG approaches which adopt a first order logic is that the domain knowledge must be perfect i.e. the expressions within the domain knowledge must be logically correct. However the use of causal models eliminates this need for a perfect domain theory since the models are defining approximate relationships from the expert's previous experience. The authors also describe a knowledge refinement approach whereby the heuristic expression derived is compared against existing expressions associated with that concept and either generalised or made more specific using a technique called difference analysis. In addition to this the authors also used what are called negative literals in the explanation generation to add more detail to the derived heuristics. Negative literals are features which are not present in the training data but their absences can go some way to differentiating the current hypothesis with another. For example in turbine generator condition monitoring the absence of a particular vibration component could differentiate between one behaviour and another. The authors demonstrated the usefulness of their approach by running an experiment where 88 new training examples were applied to their causal model using the EBG approach. This experiment managed to generate 79 new heuristic rules which were not already contained within the existing rule base. One important consideration which was not included in this work which would have to be dealt with in turbine generator condition monitoring applications would be the inclusion of temporal constraints both into the causal model and ultimately in the heuristic expressions derived for the rule base. One other aspect of this work was the method in which the causal model was constructed. The model was constructed using nodes which could represent either causes, symptoms or faults and each of these were able to be causally linked. The model therefore seemed chaotic in structure and there was no discussion on whether the developed model could be easily transferred from one type of cigarette making machine to another which may exhibit subtly different behaviours.

Analytical approaches rely on the availability of a domain theory to distinguish between relevant and irrelevant knowledge in the training data. The turbine generator condition monitoring problem is one such application where there is access to such a domain theory through the use of knowledge elicitation in conjunction with the British Energy experts. The turbine generator condition monitoring application is also affected by the problem of having very few training examples for genuine faults on the equipment due to the rarity of such events. This constraint is less of a problem for analytical approaches since they are able to learn from single training examples through the application of the domain knowledge.

Access to a suitable domain theory is not always possible. Therefore alternative approaches to learning must be employed. The alternative to using knowledge of a particular domain is to utilise the statistical distributions within large training data sets by searching for common features within training examples which fall within a particular classification. One such approach is inductive based learning which is described in the following section.

3.7.3 Inductive Based Learning

There are several inductive approaches which are predominately symbolic based. One approach, called version space search [Mitchell, 1982], works by taking a positive instance of the training data to use as an expression for the concept being learned. This concept is specific to the single training example and therefore requires generalisation to be of any real benefit. The expression is generalised using additional positive training examples to generalise the features further. Negative training instances can also be used to improve expressions which are overly general.

Another inductive based approach makes use of decision trees. A decision tree consists of linked nodes where the nodes at the bottom of the tree are called leaf nodes and represent a particular classification. An example to be classified is input at the primary node at the top of the tree. The primary node will pose a question and the answer to this will determine the next node in the tree which should be progressed to. If the next node is a leaf node then the example will be defined as the classification relating to that leaf node, otherwise another question will be asked of the example. One of the earliest examples of an inductive decision tree approach is ID3 [Quinlan, 1986]. ID3 constructs a decision tree by partitioning the training data set into smaller subsets by selecting a property and then testing each training example against that property. The algorithm recursively partitions the data further using each property in turn until all of the training examples have been partitioned into disjoint sets. The order in which the tests are made is of importance in constructing the simplest decision tree, therefore ID3 relies heavily on its approach to selecting the test at each node. A worked example taken from [Luger & Stubblefield, 1998] now follows to explain how the ID3 approach derives a decision tree form a collection of training examples. Consider how the data in table 3.1 is used to construct a decision tree using the ID3 approach.

Table 3.1: Training examples for decision tree learning training example

No.	Risk	Credit History	Debt	Collateral	Income
1	High	Bad	High	None	$0 to $15
2	High	Unknown	High	None	$15 to $35
3	Moderate	Unknown	Low	None	$15 to $35
4	High	Unknown	Low	None	$0 to $15
5	Low	Unknown	Low	None	Over $35
6	Low	Unknown	Low	Adequate	Over $35
7	High	Bad	Low	None	$0 to $15
8	Moderate	Bad	Low	Adequate	Over $35
9	Low	Good	Low	None	Over $35
10	Low	Good	High	Adequate	Over $35
11	High	Good	High	None	$0 to $15
12	Moderate	Good	High	None	$15 to $35
13	Low	Good	High	None	Over $35
14	High	Bad	High	None	$15 to $35

The first task is to select what property from the training examples to use as the first node in the decision tree. The approach adopted by ID3 for selecting the property comes from the area of information theory [Shannon, 1948]. This approach is based on determining what property would supply the highest level of information if selected as the test for that particular node. To determine the test which supplies the highest level of information, that is the highest information gain, the measure of entropy from information theory is used. Entropy is defined as:

$$\text{Entropy}(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

3.5

Where S is a collection of examples and p_i is the proportion of S belonging to class i. Entropy(S) is defined as the uncertainty of event S, where an entropy of 0 indicates that there is no uncertainty since all examples in S belong to a single class i. The test which gives the highest information gain is one which gives the largest reduction in entropy when partitioning the examples according to that particular test. Therefore information gain is calculated using:

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} |S_v|/|S| \ \text{Entropy}(S_v)$$

3.6.

Where Gain(S, A) is the information gain of an attribute A, relative to a collection of examples S. Values(A) is the set of all possible values of attribute A, and Sv is the subset of S for which attribute A has value v. The first term in equation 3.6 is the entropy value of the original collection of examples S and the second term is the expected entropy after S is partitioned using attribute A.

If we select property income from the example given in table 3.1, the training set will be partitioned into $S_1 = \{1,4,7,11\}$, $S_2 = \{2,3,12,14\}$ and $S_3 = \{5,6,8,9,10,13\}$. The entropy value from the original set of data S is calculated as:

$$\begin{aligned}\text{Entropy}(S) &= -p_{high}\log_2 p_{high} - p_{mod}\log_2 p_{mod} - p_{low}\log_2 p_{low} \\ &= -(6/14)\log_2(6/14) - (3/14)\log_2(3/14) - (5/14)\log_2(5/14) \\ &= 0.524 + 0.476 + 0.531 \\ &= 1.531 \text{ bits}\end{aligned}$$

The entropy values for the remaining subsets S_1, S_2 and S_3 following the use of the property income to partition the data are:

$$\begin{aligned}Entropy(S_1) &= -p_{high}\log_2 p_{high} - p_{mod}\log_2 p_{mod} - p_{low}\log_2 p_{low}\\ &= -(4/4)\log_2(4/4) - (0)\log_2(0) - (0)\log_2(0)\\ &= 0 \text{ bits}\end{aligned}$$

$$\begin{aligned}Entropy(S_2) &= -p_{high}\log_2 p_{high} - p_{mod}\log_2 p_{mod} - p_{low}\log_2 p_{low}\\ &= -(2/4)\log_2(2/4) - (2/4)\log_2(2/4) - (0)\log_2(0)\\ &= 0.5 + 0.5\\ &= 1 \text{ bit}\end{aligned}$$

$$\begin{aligned}Entropy(S_2) &= -p_{high}\log_2 p_{high} - p_{mod}\log_2 p_{mod} - p_{low}\log_2 p_{low}\\ &= -(0)\log_2(0) - (1/6)\log_2(1/6) - (5/6)\log_2(5/6)\\ &= 0.431 + 0.219\\ &= 0.650 \text{ bits}\end{aligned}$$

Finally the information gain using the attribute income can be calculated as follows:

Gain(s, income) = 1.531 – (4/14*0) – (4/14*1) – (6/14*0.650) = 0.966 bits

Similarly it can be shown that:

Gain(S, credit history) = 0.266
Gain(S, debt) = 0.581
Gain(S, collateral) = 0.756

Because income provides the most information gain then ID3 will select it as the root of the tree as shown in figure 3.22.

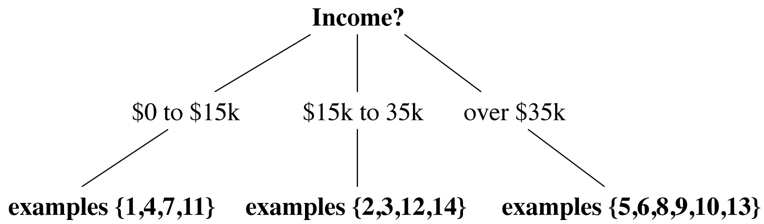


Figure 3.22: Income is selected as the root node of the decision tree using the ID3 approach

Since examples {1,4,7,11} can't be split any further this becomes a leaf node which denotes that any person with an income between $0 to $15k is a high risk. This analysis is continually applied to the two remaining training sub-sets until each set cannot be divided any further and the tree is completed as shown in figure 3.23.

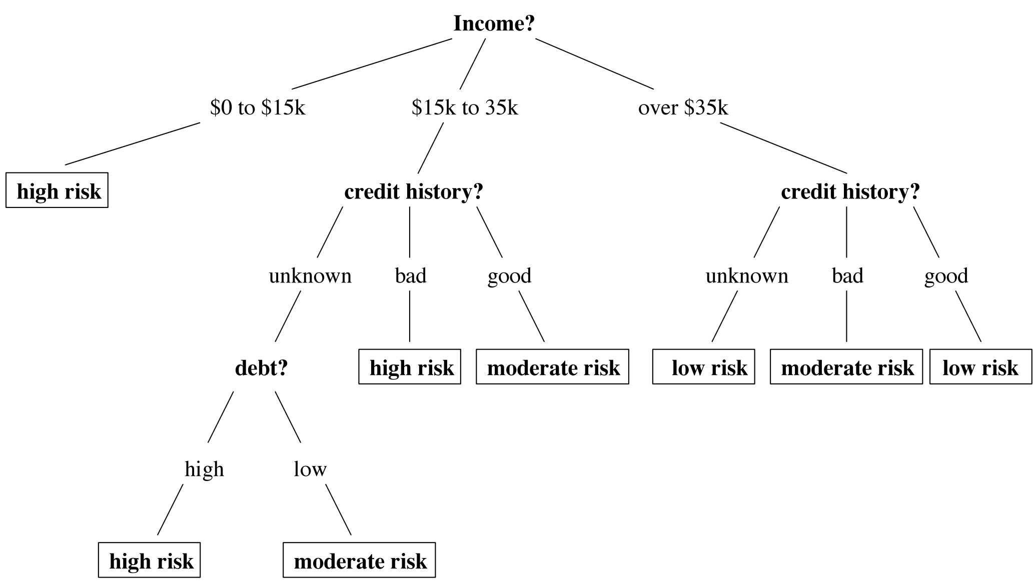


Figure 3.23: Complete decision tree derived from the data in table 3.1 using the ID3 approach

Some problems arise with this approach in instances where some of the data within the training set is erroneous or missing. Also the ID3 approach does not allow the use of continuous data. Further decision tree approaches have been developed to try and alleviate some of these problems. These approaches are C4.5 [Quinlan, 1993] and C5.0 [Strachan, 2005].

3.7.4 Applied Inductive Learning Approaches

Decision tree approaches have been shown to be effective at deriving rule-based knowledge in a wide range of applications. Some of these applications have been within the area of condition monitoring and fault diagnosis in power systems domains. C4.5 is applied to data taken from a power generation database in [Mejia-Lavelle & Rodriguez-Oritz, 1998]. The authors report on a decision tree and subsequent rules which are derived using the C4.5 algorithm. The rules are intended to determine if a power generating plant is deemed to be high performing based on information such as the capacity of the plant and information relating to staff etc. The C5.0 algorithm is applied in [McArthur et al, 2004] and [Strachan, 2005] to derive heuristic rules capable of detecting partial discharge faults from UHF transformer data. The authors demonstrate that symbolic rules can be derived from abstract numerical data. The authors also make an attempt at translating the derived rules so that they are more meaningful to the human expert user. It is reported that a significant amount of training data is used to derive the knowledge. The C.5 algorithm is also used in [Strachan, 2005] and [Strachan et al, 2007] to learn heuristic rules for a circuit breaker condition monitoring system from features extracted from trip coil data. In this application the K-Means clustering approach is used to derive distinct groups of circuit breaker behaviour from unclassified data. The C.5 algorithm is then used to derive rules to classify these clusters based on the feature vector derived from the current signal. This approach was used as a platform to assist domain experts in identifying key features for identifying unhealthy circuit breakers and therefore deriving explicit rules for a decision support system. One aspect of this application which made it suitable for the application of clustering and rule induction approaches was that there was a sufficient population of training data available.

Although decision tree rule induction has been shown to be effective in all of these applications, it does rely on the existence of a sufficient training data set as demonstrated in all of the applications reviewed. With respect to the turbine generator condition monitoring application, a sufficient training set was not available due to the rarity of genuine faults on such pieces of equipment.

3.8 Connectionist Networks

Another area of ML outlined in the hierarchy in figure 3.14 is Connectionist Networks which are more commonly known as ANNs. ANNs have already been discussed at length in section 3.3 where they were described as an AI approach to diagnostic problem solving. They were included in this section because ANNs have been widely adopted as an approach to diagnostic problem solving for applications over multiple domains, not least in power system diagnostic and condition monitoring problems. Any discussion of AI diagnosis would not be complete without such an overview. However, fundamentally ANNs are ML techniques in that they take training data and use this to incrementally improve their performance at carrying out a particular function using feedback on the performance of the previous attempt. The knowledge learned by ANNs is, however, inherently different from the type of knowledge learned by the symbolic based learning techniques outlined in the previous section. The type of knowledge derived by ANNs is implicit in the weighting factors assigned to each neuron within the network. These neurons do not relate in anyway to the semantics of the problem domain. That is, they do not represent any symbolic based knowledge which is commonly associated with the area of interest. This type of implicit knowledge can only be interpreted by the ANN approach itself and is in most cases, if not all, meaningless to humans. Therefore, the ANN approach is not suitable for applications, such as the turbine generator condition monitoring application, where the learned knowledge must be integrated with a system which utilises a symbolic based approach and where the rationale behind an assessment is of particular importance.

3.9 Instance Based Learning

Instance based learning (commonly referred to as CBR) [Kolodner, 1993] is an attempt to draw conclusions or make decisions on a particular problem based on previous examples and their associated conclusions. This process is closely matched to the intuitive approach used by humans when referencing previous instances of a given situation with a current problem to determine if any conclusions can be drawn and applied to the current situation.

A CBR system contains a knowledge base, which is composed of previous cases within a particular problem domain. These cases are composed of three primary component parts.

1. Problem/situation description: the state of the world at the time the case occurred.
2. Solution: the stated or derived solution to the problem specified in the problem description.
3. Outcome: the resulting state of the world when the solution was recorded.

The first stage in the CBR approach is to retrieve previous cases which are similar to the current case under investigation. The retrieved cases must then be analysed to determine what action should be taken in relation to the current case. A solution taken from a single case might be re-used if it bears a close enough resemblance to the current example or it may be the case that an aggregation of several case solutions is put forward as a revised solution. The final stage in the process is to determine if the case should be retained thereby enhancing the knowledge within the system. The three stage process can be summarised as follows:

- Retrieve similar cases.
- Derive conclusion through either re-use of a single conclusion or adaptation of several conclusions.
- Retain current example within case base.

Each of the three stages are explained further in the following sub sections.

3.9.1 Retrieve Similar Cases

This stage of the CBR approach compares the current case against all historical cases in the knowledge base in order to measure the similarity between each. This is achieved by comparing the relevant features of the current case with all of, or a specially selected population, of historical cases from the knowledge base. Searching only a specially selected population of the cases is of benefit where there may be a large number of cases to match against. The search can be performed more quickly if the case base is partitioned so that only cases which share a similarity with the current case are matched against. This approach can lead to improvements in efficiency, especially for large case bases. The features of a case may be represented either qualitatively as symbols, quantitatively or both. CBR approaches therefore require approaches for comparing both types of data

CBR makes no attempt to derive an explicit expression such as those generated in the aforementioned symbolic based approaches, nor do they derive a weighted network as in the ANN approach, which once derived can be used to interpret new cases directly without having to revert back to the training data. Instead each time a new instance needs processed the CBR approach relies purely on matching this against the case base examples to assess the new data. The CBR approach relies heavily on the availability of suitable cases being available to make the comparison between the new instance and the population of training data, in addition to the data being of a reliable accuracy.

The features used to describe a case are also used as the indices for matching one case against another. CBR approaches do not require an exact match between all of the indices for the cases to be matched as is the case with database searches. Instead cases can be partially matched in that not all indices are the same or the same index may not match exactly. Therefore, it should be clear that an important performance factor in CBR techniques is the approach adopted for retrieving similar cases. One such approach is that of the k nearest neighbour [Strachan, 2005].

Once a population of similar cases has been selected by the similarity approach it is required to rank each in order to determine which should be brought forward to the next stage for consideration of re-use.

3.9.2 Derive Conclusion

Each case has associated with it a solution to the problem defined within the case. In some situations the retrieved case may be deemed a close enough match to the current case for the conclusion to simply be re-used without having to make any changes. In other instances it may be necessary to adapt the conclusion either through adapting the existing conclusion to suit the current situation or by aggregating the solution to multiple solutions to construct a single solution. The process by which the current conclusion is made to fit more closely with the current situation is known as adaptation. This involves determining which elements of the existing solution should be changed and what actual changes should be made to these elements.

In domains which generally contain clearly defined conclusions, the process of adaptation may never be required. Alternatively, systems which act more as an advisory tool to expert users may leave any adaptation to be undertaken manually. Automated adaptation relies on determining the differences between the case under analysis and the current solution case and applying heuristics to these differences to determine a suitable new conclusion. Automated adaptation may involve a new component being introduced to an old solution (insertion), a component being removed from an existing solution (deletion), a component being replaced (substitution) or an old solution may require extensive changes (transformation). It should be noted that the extensive use of adaptation introduces a knowledge based element in addition to the basic inductive approach which underpins CBR.

3.9.3 Retain Case

In most CBR applications it is impractical and too costly to retain every case processed by the system. Therefore a decision must be taken following assessment of each case on whether to retain it in the case base or not. The most important consideration when determining the most suitable case is the closeness of the match between the current case and those in the case base. When deciding if a case should be retained or not, the most important consideration is whether it is unique enough to add value to the already existing cases. Although the case will have displayed a degree of similarity with others already present in the case base, it may not have been a complete match and therefore may contain new information which can be utilised by the CBR system.

3.9.4 Applied Case-Based Reasoning

Although there are few examples of CBR being applied to diagnosis and condition monitoring tasks within the area of power systems, there have been some notable attempts in other domains. The CBR system described in [Derere, 2000] diagnoses faults on diesel-electric trains through the use of a case base constructed specifically for the project. The system matches information taken from the train control modules to determine which subsystem is causing the fault. The system then locates the particular cause of the fault by asking the technician a series of questions to add more detail to the case information and therefore provide a more accurate assessment. This application of CBR demonstrates that the approach can be useful for constructing a whole system but it demonstrates that there would be little benefit in using it to derive knowledge for an Expert System since no explicit knowledge is derived. Another domain where CBR has been used for the purpose of diagnosis is medicine. PROTOS [Porter, 1986] was a CBR system which was designed to diagnose hearing disorders using patient records which contained symptoms, history and test results. This application again demonstrated that CBR was well suited to being the main approach in which to build an intelligent system i.e. an alternative to using an approach such as an Expert System. CBR however is not well suited for deriving knowledge to be utilised by alternative approaches. This is primarily because the CBR approach does not derive any explicit knowledge. Instead the learning and the derivation of knowledge is derived by the CBR technique used to measure similarity. Therefore CBR had to be discounted as an approach for deriving knowledge to be used within the Expert System developed for the turbine generator condition monitoring application.

3.10 Statistical Based Approaches

The heading of statistical based approaches has been used in the taxonomy given in figure 3.14 to encompass a multitude of techniques whose roots derive firmly from the probabilistic/statistical area of mathematics. This thesis has already touched upon some techniques which derive from a statistical background such as ANNs or even as demonstrated by Mitchell in version space search or inductive decision tree approaches such as ID3 [Mitchell, 1997]. These approaches, however, have been separated from statistical based approaches either due to the fact that they have been so widely adopted in the area of automated diagnosis they merit being in a field of their own, or because they primarily deal with a specific data type as is the case for the symbolic based techniques. To give an insight into some other forms of statistical based approaches this section gives an overview of two techniques which have found much application in recent years. The first discussed is the area of Bayesian based approaches which encompasses the Bayesian classifier, the naïve Bayesian classifier and Bayesian Belief networks. The second area discussed is Hidden Markov Models (HMMs).

3.10.1 Bayesian Learning Approaches

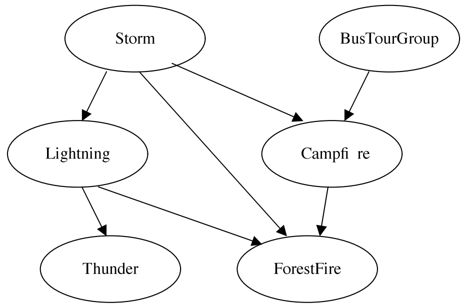
Bayes theorem provides an inference to determine the probability of some event given a set of observations and prior knowledge. The prior knowledge required by Bayes theorem relates to the likelihood of the set of observations given that the event has occurred and both the probability of the event given no observations and the probability of the set of observations. Bayes theorem is defined as:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} \tag{3.7}$$

Where P(h) is the probability of the event given no observations. Similarly P(D) denotes the probability of the set of observations. P(D|h) denotes the likelihood of the set of observations given that the event has occurred. Finally, P(h|D) is the probability of the event given a set of observations. Bayes theorem can be extended to perform diagnosis problems using either the Bayes optimal classifier or the naïve Bayes classifier. A full discussion of how Bayes theorem can be used to derive both of these is given in [Mitchell, 1997]. Both classifiers extend Bayes theorem to determine the most likely event which has occurred given the set of observations. The optimal Bayes classifier obtains the best performance that can be achieved from the observations and knowledge given but can be very costly to compute whereas the naïve Bayes classifier makes some simplifying assumptions in order to reduce this processing cost. The naïve Bayes classifier introduces the assumption of conditional independence in that the states observed are not dependent on one another. For example the conditional independence assumption would mean that the relationship between an increase in steam input to a turbine and the subsequent increase in electrical watts produced by the generator are not related i.e. there is no conditional probability held between both observations.

An example of where a Bayes classifier has been applied to an industrial related problem is given in [Haji & Toliyat, 2001]. The authors used a simplified experimental set-up to test the use of a Bayes classifier to diagnose a rotor bar fault on a lab based induction motor. The experimental set-up captured information relating to the speed of the motor and from features extracted from the measured torque. The experiment was run for both a healthy motor and one which had a rotor bar fault introduced. The access to such a lab based set-up allowed the authors to produce multiple training examples which could be used to derive the probabilistic knowledge for the Bayes classifier. The authors report good results for the Bayes classifier at diagnosing a single example of a rotor bar fault, however the degree at which the experiment had been simplified (the classifier only had to choose between healthy and rotor bar fault) did not prove the approaches applicability to a complex real world application. In addition the experimental set-up provided the authors with multiple examples to train the classifier which may not be the case in certain real world industrial applications such as the turbine generator condition monitoring application.

Bayesian belief networks provide an intermediate approach between the naïve Bayes classifier assumption of conditional independence and the optimal Bayes classifier which ignores conditional independence completely. It does this by constructing a network which indicates the conditional probabilities held between variables. The example of a Baysian belief network given in figure 3.24(a) from [Mitchell, 1997] represents the joint probability distribution over variables ‘Storm’, ‘Lightning’, ‘Thunder’, ‘ForestFire’, ‘Campfire’ and ‘BusTourGroup’.



	S,B	S, ¬B	¬S,B	¬S, ¬B
C	0.4	0.1	0.8	0.2
¬C	0.6	0.9	0.2	0.8

(a) **(b)**

Figure 3.24: Baysian belief network.

The conditional probability table given for the variable 'Campfire' in figure 3.24(b) describes the probability distribution for that variable given the values of its immediate predecessors 'Storm' and 'BusTourGroup'. 'BusTourGroup', 'Storm' and 'Campfire' have been abbreviated to 'B', 'S' and 'C' respectively in the conditional probability table. The conditional probability table in figure 3.24(b) indicates that given there is a 'Storm' and a 'BusTourGroup' there is a probability of 0.4 that a 'Campfire' will be lit but if there is a 'Storm' and no 'BusTourGroup' there is still a probability of 0.1 that the 'Campfire' will be lit. This is accounting for the fact that there may be someone not in the 'BusTourGroup' present to light the 'Campfire'. In addition, the Baysian belief network asserts the conditional independence of the variables within the network. If two variables are not linked by a directed path then they are said to be conditionally independent. For example, it is shown in figure 3.24(a) that the 'Campfire' node is conditionally independent of all other variables apart from its parents 'Storm' and 'BusTourGroup'. Bayesian belief networks provide a convenient means of combining causal and probabilistic knowledge with Baysian inference approaches to infer new knowledge. Approaches have been developed to learn the conditional probabilities of Bayesian belief networks. These approaches are similar to those used to derive the weights in ANNs as described in section 3.3.1 and include the gradient descent approach used in [Yongli et al, 2006] & [Yan & Lanqin, 2006]. Some other applications of Bayesian networks rely on the experience of a domain expert to assign the conditional probabilities such as in [Chien et al, 2002]. All researched industrial applications where Bayesian networks have been applied have relied on experts to develop the network structure manually through their experience although there is some discussion in [Mitchell, 1997] of instances where the network structure has been developed using learning approaches.

3.10.2 Applied Bayesian Networks

A Bayesian network is used in [Chien et al, 2002] for fault diagnosis on a network distribution feeder. The network's causal dependencies along with the conditional distributions were derived through consultation with condition monitoring experts, not through learning approaches. The causal network related typical observations or information which could be captured by the distribution network owner such as the reporting of fires, customer complaints etc, with hypothesis of what has happened and the resulting impact this situation would have on the network. The authors show that the Bayesian network developed performed well at predicting the state of the network following a fault when compared against historical data. However, from a learning viewpoint, the authors relied completely on the experts to provide the relevant probabilistic knowledge for the network as opposed to learning this automatically. As acknowledged by the authors there is a limit to the level of precision obtained by capturing the probabilistic knowledge in this way which could in turn reduce the accuracy of such a system. Additionally, since the knowledge is not being captured automatically, an additional burden is placed on the development of the system in the form of additional knowledge elicitation.

Another application of Bayesian Networks is reported in [Yongli et al, 2006] & [Yan & Lanqin, 2006]. Both of these papers describe an approach to diagnosing the location of faults on a power systems network. Logical fault models for the failure of transmission lines, transformers and busbars are transformed into Bayesian Networks. These fault models can then be applied to any power systems network topology to derive the location of any transmission line, busbar or transformer fault on the network. The conditional probabilities within the Bayesian Networks are learned using an approach analogous to the back-propagation approach used for learning the neuron weights in ANN's described in section 3.3.1. The conditional probabilities are initialised and training data is then applied to the Bayesian Networks. The output of the Bayesian Networks can then be used to update the conditional probabilities. This process is then repeated until a termination criteria is met. The benefits of using this approach, in such an application is reported as the Bayesian Networks ability to make an accurate diagnosis in instances where the data is incomplete or missing. This problem is prevalent in this application since the failure of a circuit breaker to open or even the operation of a breaker being lost over the data transmission system is common. The authors also highlight that the 3 basic logical models are easily applied to any power system network to derive the required Bayesian Network. The application does however benefit from the availability of training data to derive the conditional probabilities required for the Bayesian Networks. Bayesian approaches seem to provide little benefit with respect to the derivation of explicit knowledge for the turbine generator condition monitoring Expert System. This is because they do not derive explicit knowledge and because the application does not have access to a large set of training data.

3.10.3 Hidden Markov Models

Hidden Markov Models (HMMs) are an attempt at modelling a particular system probabilistically. HMMs are an extension of the Markov Process which describes a system in terms of states, a state transition matrix and a set of initial conditions. To illustrate lets take an example from [Sonka et al, 1999] which models a simplified weather system as shown in figure 3.25.

Initial Conditions (π)**:**
- Sunny - 1.0
- Cloudy - 0.0
- Rainy - 0.0

States:
- Sunny
- Cloudy
- Rainy

State Transition Matrix (A):

Today

Yesterday

	Sunny	Cloudy	Rainy
Sunny	0.5	0.1	0.4
Cloudy	0.4	0.3	0.3
Rainy	0.1	0.6	0.3

Figure 3.25: A Markov process which models a simplified weather system.

The weather system is represented as a Markov Process as the states *Sunny*, *Cloudy* and *Rainy*, the state transition matrix (A) which defines the probability of moving from one state to the other plus the initial conditions (π) which indicates the probability of the system initially being in that particular state. The weather system can then be represented as a state transition diagram as shown in figure 3.26.

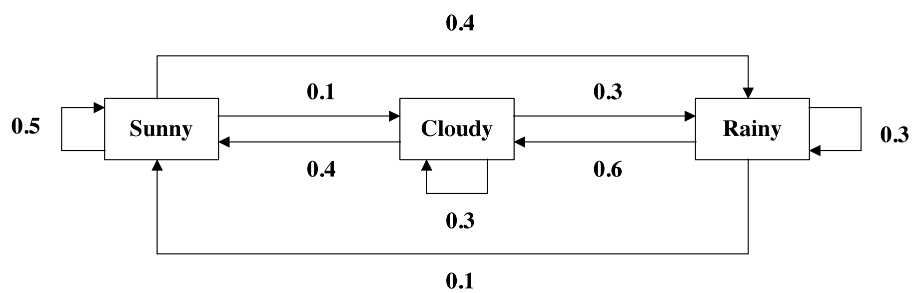


Figure 3.26: State transition diagram for the weather system Markov Process.

In some instances we may not have access to the observable being modelled but we may have access to a related observable as in figure 3.27.

Confusion Matrix (B):

Seaweed

Weather

	Dry	Moist	Damp	Soggy
Sunny	0.5	0.1	0.2	0.2
Cloudy	0.3	0.4	0.1	0.2
Rainy	0.1	0.6	0.2	0.1

Figure 3.27: A confusion matrix probabilistically relates the moisture content of seaweed to the task of predicting the weather.

The matrix depicted in figure 3.27 lists the probability of a given weather condition given the moisture content of seaweed. The name given to this type of matrix is the confusion matrix (B). The additional information provided by the confusion matrix can be added to the state transition diagram as shown in figure 3.28.

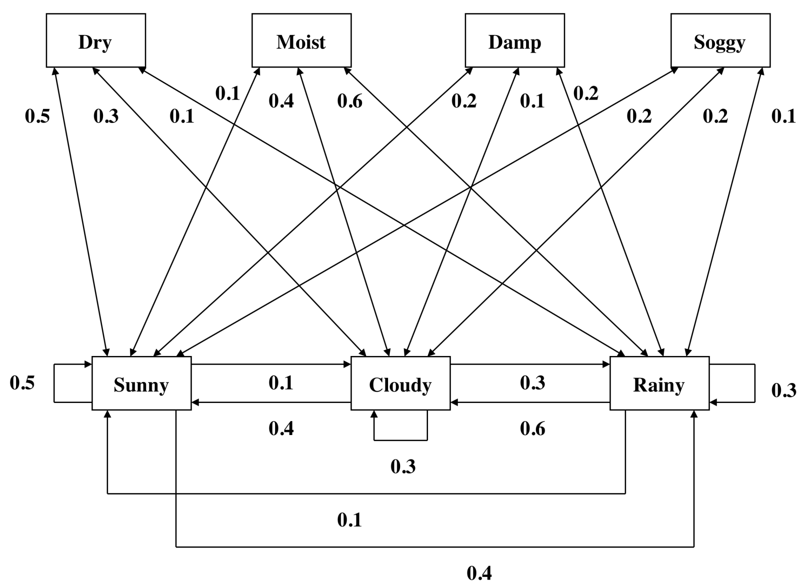


Figure 3.28: State transition diagram with additional observable data added for the weather system Markov Process.

HMMs use a confusion matrix (B) combined with the initial conditions (π) and a state transition matrix (A), here the state transition matrix is said to be hidden, to perform one of three problems.

1. The first is known as the evaluation problem, which given a model (λ) and a set of observations (O), how do we compute the probability that the observed sequence was produced by the model i.e. $P(O|\lambda)$.
2. The second problem is to uncover the hidden part of the model i.e. to uncover the states which are most likely to have produced observations (O) given model (λ).
3. The last problem is where we attempt to optimise the model parameters (λ) that best describe how a set of observations (O) are produced.

All HMM problems are defined as follows:

1. The number of states (N) in the model. Although these states are hidden the number normally corresponds to some physical attribute of the system.
2. The number of distinct observation symbols per state (M). The observation symbols correspond to the output of the system being modelled, therefore in the weather examples this would be sunny, cloudy and rainy.
3. The state transition probability distribution (A).
4. The confusion matrix (B).
5. The initial conditions (π).

From a learning perspective, HMMs offer a lot of potential to diagnostic problems where there is a lack of understanding of the domain or where the area is very complex. The third type of problem outlined above suggests an approach to learning a model of a particular domain without the need for knowledge elicitation. This approach is similar to that undertaken by ANNs but HMMs differ in two key areas. The first is that HMMs are temporally dependent in that the derivation of the output state is dependent on the previous n states. This is advantageous in the fault diagnosis and condition monitoring domain since two behaviours may appear similar when a snapshot of some data set is taken, but when put into a temporal context and analysed against previous states, they can exhibit distinct behaviours. A second difference is in the representation of the learned knowledge. ANNs are traditionally difficult for humans to interpret. However, the states within the HMM model commonly have some physical significance to that system [Rabiner, 1989] which offers the potential of the users being able to interpret the knowledge learned. Once these statistical models have been developed HMMs can be used much in the same way as ANNs for tasks such as anomaly detection and fault diagnosis. Anomaly detection can be achieved by performing the second type of problem outlined above. A model of good behaviour of a system can be developed and then compared against the observed behaviour of a system. When the real system is behaving as expected the HMM process will predict from the observed behaviour that hidden state transitions, which are more likely to occur according to the model, have produced the observations. In contrast the HMM process will predict from a set of observations, for a system deviating from normal behaviour, hidden state transitions which have less likelihood of occurring according to the state transition models learned for normal behaviour. HMMs can also be used for fault diagnosis by undertaking the first type of HMM problem outlined above. Models of faulty behaviour can be developed from training data in the same way that a model of healthy behaviour can be. These models can then be compared against the observed behaviour to determine which maximises $P(O|\lambda)$.

3.10.4 Applied Hidden Markov Models

There have been some examples of HMMs being applied to industrial based problems. One of these is reported in [Brown et al, 2007]. Here the authors report on the development of an anomaly detection agent which was integrated with the existing agent based transformer condition monitoring system COMMAS. The anomaly detection agent was developed by firstly developing a model of good transformer behaviour. The model was developed to utilise features extracted from Ultra High Frequency (UHF) partial discharge data. The hidden states of the HMM were determined using clustering approaches and the state transition probabilities were learned using the Expectation

Maximisation (EM) [Baum et al, 1970] learning approach. The model of healthy behaviour could then be used to detect anomalies in transformer behaviour using the Viterbi algorithm [Forney, 1973] which predicts the most likely state based on the observations and a given model. Instances where the algorithm predicted an unlikely state would imply that the transformer in question was deviating from normal behaviour. The anomaly detection approach described in this paper provides an avenue of research in relation to the transformer condition monitoring application. Such an approach would provide an alternative to the limit checking approach employed by the Beran system described in section 2.2. The data stored currently by the Beran system would provide such an approach with adequate data to train a model for turbine generator normal behaviour. Therefore this field of development should be explored as a further avenue of research within this area.

Another application of HMMs is reported in [Nakamura et al, 2007]. Here the authors have trained multiple HMMs for varying degrees of winding turn-to-turn faults on induction motors using an experimental set-up. Observed features derived from the current waveform of an induction motor can then be compared against these fault models to determine which produces the highest likelihood $P(O|\lambda)$. The fault model which produces the highest likelihood $P(O|\lambda)$ would indicate the fault type affecting the induction motor. The tests performed highlighted that the models were not always able to differentiate between a healthy motor and a faulty one. Another approach for diagnosing faults in induction motors using HMMs is proposed in [Ocak & Loparo, 2001]. This approach attempts to diagnose mechanical faults on the motor using vibration data. Models were developed for each fault type using vibration data captured from an experimental set-up with various faults introduced. The observed vibration features were then matched against the HMM fault models and the one with the highest likelihood $P(O|\lambda)$ determined the fault which the motor was experiencing. Both of these applications demonstrated how HMMs could be used for fault diagnosis applications; however both examples used experimental set-ups to train and test their systems. This allowed the authors to create suitable training data to train the models. Such an experimental set-up is not available for the turbine generator condition monitoring application. Another limitation of all of the HMM applications described is the form in which the knowledge learned is represented. The learning module for the turbine generator application requires that the knowledge learned is compatible with the Expert System developed, which the HMM probabilistic models would not be.

3.11 Learning Technique Selection

Section 3.5 explained that the diagnostic AI approach selected for the turbine generator condition monitoring application was the Expert System approach. The objective was therefore to select an approach suitable for the application which would be able to assist in the derivation of knowledge for the diagnostic system. The approach therefore had to be capable of deriving knowledge which is compatible with the Expert System approach; that is symbolic heuristic knowledge. This requirement therefore eliminated all of the ML approaches with the exception of the two symbolic ones. Another constraint placed on the selection of the ML approach was that training data for genuine turbine generator faults was not available in large sets. Therefore the ML approach selected would have to be capable of deriving the knowledge from single training examples. This constraint therefore omitted rule induction approaches such as ID3, C4.5 and C5.0. The only remaining suitable area of ML was that of symbolic analytical approaches. This area of ML was well suited to the turbine generator condition monitoring problem since there was access to domain knowledge from the British Energy experts and at least some examples of the faults to be learned were available from historical raw data on the Beran system. Specifically the EBG approach was selected for this particular application since its problem definition matched well with the learning problem encountered for this project as explained in detail in section 5.3.

The following chapter will now describe in detail the Expert System developed for British Energy for the turbine generator condition monitoring application.

Chapter 4

4 Alarm Assessment Expert System

4.1 Introduction

The previous chapter gave a detailed overview of the main Artificial Intelligence (AI) and Machine Learning (ML) techniques which have been adopted in condition monitoring and diagnosis applications. This chapter will now outline in detail the Expert System which has been developed to assist the turbine generator condition monitoring task outlined in section 2.3. Section 4.2 outlines the rationale behind selecting the Expert System approach. Section 4.3 outlines the knowledge engineering approach used to derive the design of the Expert System. Section 4.4 then demonstrates how the results of the knowledge engineering approach were used to develop the architecture of the system. Section 4.5 describes how the system design was realised as a prototype system by outlining how the key features were extracted from the data, how the interface was designed to maximise the information extracted by the Expert System and how this assisted the experts at the verification stage. A review of how the system performed on real data taken from the Beran condition monitoring system is then detailed in section 4.6 by contrasting the Expert System results with British Energy's historical records. Section 4.7 then discusses the overall performance of the Expert System and any additional features along with improvements which could be implemented to integrate it with British Energy's on-line system.

4.2 Technique Selection

British Energy's approach to turbine generator condition monitoring was outlined in section 2.3. It was shown that all of the experts throughout British Energy's UK locations use a standard approach to analyse the alarms generated by the Beran condition monitoring system described in section 2.2. The final outcome of the analysis is a fully completed checksheet for each alarm, which notes important parameters in the analysis process and ultimately the cause of the alarm. The results of these analyses can be monitored to determine if any faults are developing on the equipment and to subsequently plan any actions which may have to be taken based on these results. Much emphasis is placed on the monitoring and reporting of these behavioural changes as a result of the strategic importance of plant items such as turbine generators. In addition, regulatory bodies undertake quarterly inspections of all alarms triggered by the Beran system in each station to ensure that all events are being thoroughly and accurately assessed.

Experience has shown that many alarms are commonly caused by faulty signals, signal drift or changes in operational parameters which cause the vibration signals to temporarily move outside their limits. Generally these signals do not provide the experts with information on the health or state of the equipment and so have no further operational consequence. However, each alarm must be inspected by the three full time employees and five part time contractors within British Energy who are qualified to do so. This effectively intensifies the already substantial daily workload on this small team.

This problem prompted British Energy to commission a project which aimed to develop a system capable of automatically completing the checksheet and diagnosing the cause of each alarm triggered by the Beran system. The system had to enable an expert to select an assessed alarm, review the system analysis and then sign off the alarm. This would allow the expert to quickly confirm alarms of no further operational consequence and focus their time and expertise on diagnosing incipient faults which may impact on the health and operation of the turbine generator. In addition the system was to assist in standardising the analysis performed across all British Energy locations so that the results are accurate and consistent for the external quarterly audit.

The Expert System approach was the AI approach chosen to realise an automated system for the aforementioned problem. The reasons for choosing this approach can be summarised as:

- The development team had access to expert diagnostic knowledge.
- The rationale provided by the explicit knowledge provides an explanation to each assessment.
- Assessment explanations would enable experts to quickly verify the alarm cause
- The assessment explanation increases the level of user confidence.
- Expert knowledge is captured and recorded for the company archive and used to standardise the analysis performed across all UK locations.

One of the primary reasons for selecting the Expert System approach was the availability of knowledge. 660MW turbine generator sets are complex and dynamic machines meaning that modelling the domain of interest was prohibitive. An MBD approach, as discussed in section 3.4, requires that functional models of the domain are available. Such models deemed accurate enough were simply not available for the British Energy turbine generator sets, nor was it deemed a feasible task to develop such models. It was however more practical to acquire heuristic based diagnostic knowledge from the company experts who were involved in the project. The use of a knowledge engineering approach also presented the opportunity of developing a knowledge base of the experience which had been accumulated for the company archive. This would also assist in standardising the analysis performed across all of British Energy's UK locations.

Another important reason for adopting the expert system approach was due to the representation of the knowledge, which in particular, had an impact how the system explanations would be presented to the end user. ANNs, as discussed in section 3.3, store knowledge in an implicit form which humans are unable to easily interpret. ANN results are normally presented to the user as a single classification with little or no explanation of the rationale used to derive the outcome. This lack of explanation means that user confidence is lost, or at the very least reduced, whenever a misclassification occurs. This inherent lack of transparency is a problem not shared by Expert Systems. On the contrary, an Expert System's explicit knowledge base provides a rationale and subsequently an explanation to support each assessment. This level of transparency is important for verifying correct diagnoses and even more so in cases of misdiagnosis, which will undoubtedly occur with any automated system, since it allows users and system maintenance engineers to locate and correct areas of vulnerability within the system.

4.3 Knowledge Engineering

A formal knowledge engineering approach was adopted to realise the Expert System design. The knowledge engineering methodology used was that of CommonKADS [Schreiber et al, 2000]. Knowledge engineering is concerned with capturing knowledge from experts through knowledge elicitation interviews and then documenting this using knowledge transcripts and models as discussed in section 3.2.1. The formal knowledge engineering approach used to capture British Energy expert knowledge for turbine generator condition monitoring is summarised in figure 4.1.

The first stage is knowledge elicitation where experts are interviewed in order to determine their approach and the knowledge used to fulfil the task in question. The form of knowledge elicitation interview adopted is dependent on the type of information being captured and the stage of the elicitation process. For instance, the first interviews conducted are normally less structured than later interviews to allow the expert to give general descriptions of the problem area normally through the means of worked examples. Interviews later in the elicitation process will become more structured by the knowledge engineers who having a better understanding of the problem will want to acquire more detail on specific areas. The knowledge captured during these interviews is recorded into what is known as a knowledge transcript. The knowledge transcripts document all of the relevant information such as reasoning strategy, knowledge and tasks in a structured format. The transcript is continually updated and refined through a validation process following each knowledge elicitation interview. The validation process is an iterative process between the knowledge engineers, experts and their peers. Once the transcripts are completed and verified the knowledge contained within them is transformed into knowledge models. The purpose of the knowledge models is to extract the key information from the transcripts such as the approach and knowledge specific to the task. The CommonKADS knowledge modelling methodology segregates the knowledge into three areas:

- Task Knowledge
- Inference Knowledge
- Domain Knowledge

The overall task and how it is subdivided into smaller more manageable goals is represented by the task knowledge. Task models normally embody a tree structure where the primary goal is represented by the root node. The primary goal can then be dissected into sub-goals which in turn can be further divided until an adequate level of task description has been met. Inference knowledge is used to model the reasoning approach adopted to achieve the goals set out in the task knowledge. This typically involves stipulating the order in which tasks are implemented, the flow of data between tasks and the knowledge required to fulfil these tasks. The domain knowledge is particular to the area under investigation and is required to interpret the information to achieve the tasks outlined in the task model. The domain knowledge can represent different types of knowledge such as concepts, theories or simple heuristics which map symptoms to conclusions. CommonKADS suggests representing the domain knowledge using models which are compatible with the Unified Modelling Language (UML) which is a de facto standard for representing data-structures in object-oriented software platforms such as Java (http//www.uml.org).

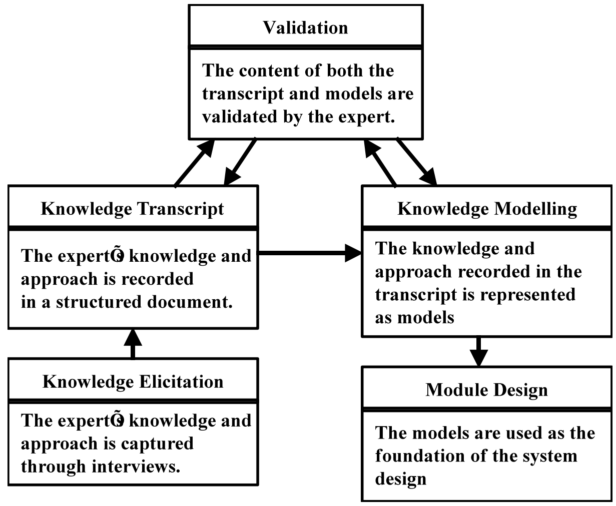


Figure 4.1: Overview the formal engineering approach used to capture expert knowledge for the turbine generator condition monitoring Expert System.

The following sections detail the results obtained from the knowledge engineering exercises undertaken with the 3 British Energy condition monitoring experts. The knowledge models within these sections therefore outline the best practice human expert approach to the assessment of turbine generator condition monitoring alarms. This should not be confused with the automated system design which is ultimately derived from these models. The system design is covered in detail in section 4.4.

4.3.1 Task Knowledge

The tasks undertaken by British Energy condition monitoring experts when assessing the condition of their turbine generators are summarised in the task models in figure 4.2. It can be seen from the figure that the five main tasks undertaken when analysing the alarms triggered by the Beran condition monitoring system are:

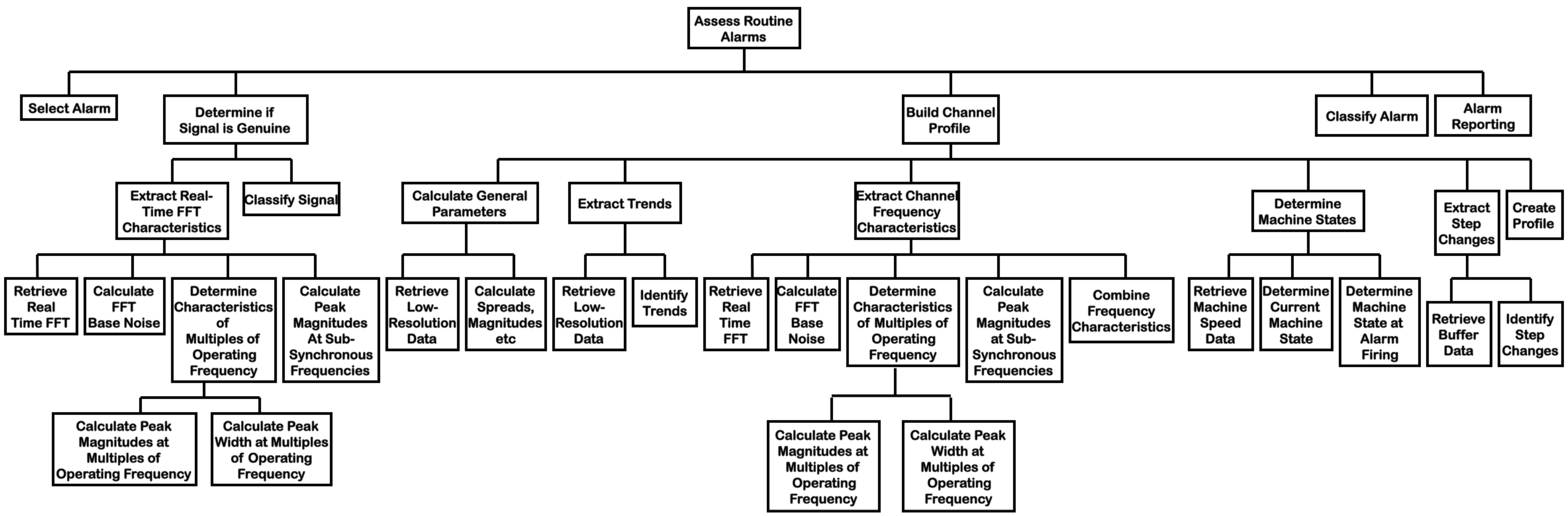


Figure 4.2: Task model for turbine generator condition monitoring alarm assessment

- Select Alarm
- Determine if Signal is Genuine
- Build Channel Profile
- Classify Alarm
- Alarm Reporting

The expert first of all selects an alarm for assessment by consulting the Beran system. When there is an alarm it is selected by the expert who then determines what data is required for the assessment. This data is then downloaded from the Beran system.

The first piece of analysis undertaken is to determine if the signal is genuine i.e. is the data being measured accurately or is there a problem with the measuring equipment. The FFT data is analysed to determine if the channel's frequency distribution exhibits that of a genuine signal. The expert inspects the real-time FFT data to determine if there are features which would indicate some error in the captured data. These features include the level of base noise, which can indicate noise in the channel, the nature of the spikes at multiples of the operating frequency, which can indicate loose sensors or verify genuine signals, and the spikes at sub-synchronous frequencies which can also be indicative of various faults as shown in figure 4.3.

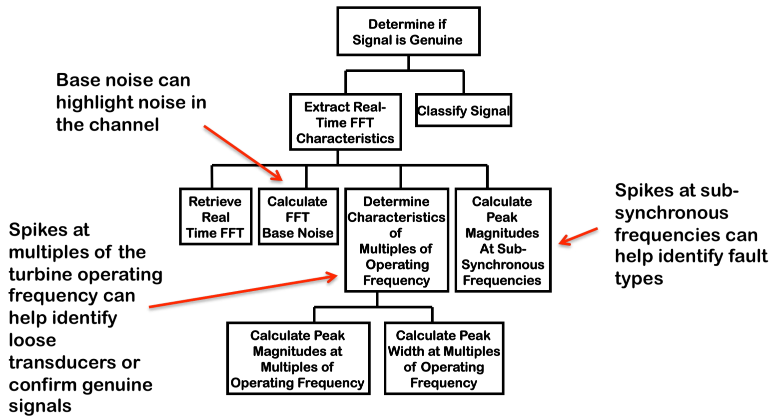


Figure 4.3: Determine if signal is genuine task from the turbine generator condition monitoring task model.

The build channel profile task requires the expert to analyse various bodies of data to determine the channel behaviour at and around the time of alarm firing. General parameters of interest such as the signal spread, average etc are approximated as shown in figure 4.4. These values can help the expert build up a general picture of how the channel is behaving. Also examined are the gradual changes which have occurred in the signals. Specifically correlations between changes and whether they could be possible contributing factors to the alarm as shown in figure 4.4.

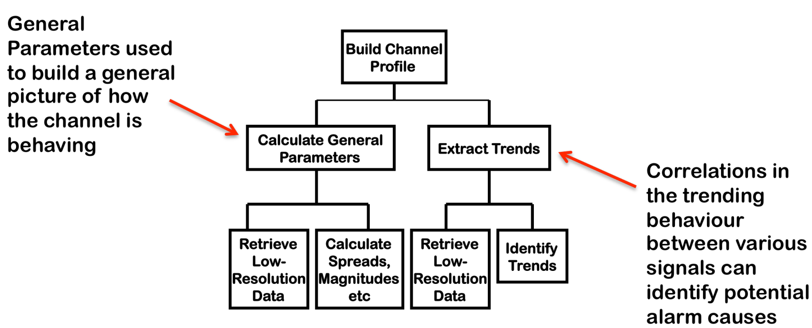


Figure 4.4: Calculate General Parameters and Extract Trends tasks, which fall under the Build Channel Profile task, from the turbine generator condition monitoring task model.

The FFT data captured at the time of alarm firing is then analysed in a similar fashion to how the real-time FFT data is analysed when determining if the signal is genuine. The features taken from the FFT can help determine possible causes of the alarm especially certain fault types as shown in figure 4.3. The current speed and the speed of the machine when the alarm fired are checked to determine the state of the machine as shown in figure 4.5. The buffer data as described in section 2.2 is also analysed to determine if sudden changes occurred which could constitute certain fault types as shown in figure 4.5. Finally, all the information is combined to compile a complete overview of the channel behaviour as shown in figure 4.5.

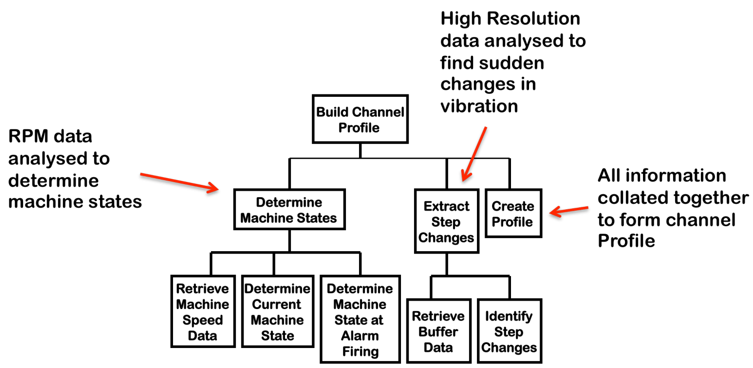


Figure 4.5: Determine Machine States, Extract Step Changes and Create Profile tasks, which fall under the Build Channel Profile task, from the turbine generator condition monitoring task model.

The next task is to determine the cause of the alarm. This is achieved by analysing the channel profile derived from the sensor data and determining if any of the information or features correspond with known alarm causes. In effect, the experts look for symptoms which relate to particular alarm causes thereby classifying the alarm. The causes of Beran alarms identified by the experts are summarised in figure 4.6.

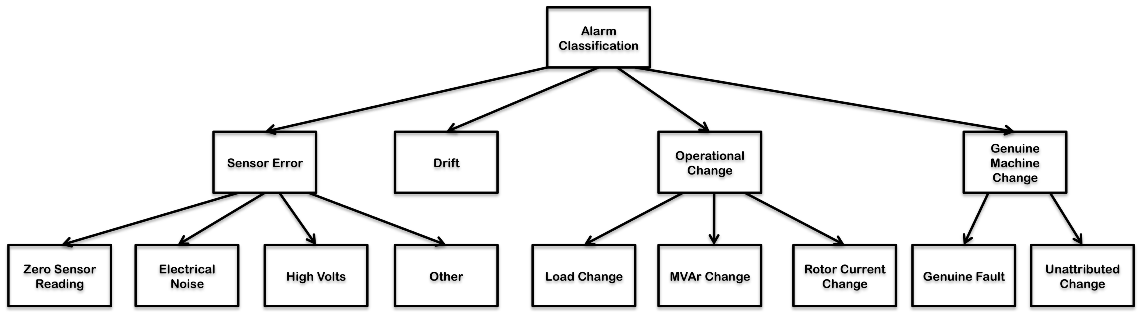


Figure 4.6: Potential causes of alarm on the Beran turbine generator condition monitoring system

Alarms triggered due to a Sensor Error may exhibit the characteristics of noise or abnormal high or low reading levels. Drift alarms are the result of small temporary movements in either the first or second order vectors which fall outside of the alarm limits. Drift movements do not have any corresponding change in the operational parameters which could explain the temporary change in the vector. Operational Change alarms are caused by a change in the operational data such as the load, MVAr or rotor current which cause a temporary change in the vibration data. Genuine Machine Change alarms indicate that some problem or fault has occurred on the turbine which requires further investigation. The change could either be Unattributed i.e. there is no corresponding change in any of the operational parameters which seem to have caused the change or a Genuine Fault which would have a distinguishable cause in the data such as a change in load etc. All of the above alarm causes are explained in greater detail in section 4.5.5.

The final task requires that the assessment results are recorded. The results along with important information taken from the alarm channel profile are entered into a Beran alarm checksheet, an example of which is given in figure 2.5.

A brief description of each of the entries now follows:

- OA Amp – The OA Amp entry requires that an approximation of the overall amplitude vibration of the affected channel is recorded. This is not always simply an average but must account for fluctuations in vibration etc.
- Zone – The zone gives some indication of the severity of the alarm and is calculated in relation to the overall amplitude level. The zones range from 1 to 4 with 1 being the least severe (lowest level of vibration) 4 being the most (highest level of vibration).
- Non-synch. Amp/Freq – Here a note is taken of the non-synchronous amplitude and frequency. A note is taken of the amplitude, but the frequency is only noted in cases where the amplitude is large, in which case the FFT is consulted to locate the frequency location of the high amplitude.
- 1x Amp – This entry requires an approximation of the first order amplitude level. As for the OA Amp entry the value is not always an average but must account for fluctuations.
- 1x Phase (*lag) – This is the same as 1x Amp entry except the signal under analysis is the first order phase as opposed to the magnitude.
- 2x Amp - This is the same as 1x Amp entry except the second order magnitude signal is analysed.
- 2x Phase (*lag) - This is the same as 1x Amp entry except the second order phase signal is analysed.
- OA Genuine? – This entry records if the overall amplitude level observed seems genuine. Methods for detecting non-genuine signals are the occurrence of signal faults and the overall amplitude not approximately equalling the addition of the first and second order magnitudes.
- OA ~ $\sum(1x+2x)$? – This entry indicates whether the first and second order magnitude levels approximately equal the overall level.
- Step $\Delta 1x$? – This entry indicates if a step change occurs in the first order magnitude or phase.
- Signif. $\Delta 2x$? – This entry indicates if a significant change occurs in the second order magnitude or phase.

- Operational Change? – This entry indicates if there has been a change in any of the operational parameters such as load, rotor current, steam temperatures etc.
- Commentary – The primary purpose of this entry is to indicate the cause of the alarm and any remedial action which must be taken

The experts identified the completion of this checksheet as the primary objective of the automated system being developed.

4.3.2 Inference Knowledge

The knowledge engineering activity also resulted in inference models which outlined the reasoning approach used by the experts. The high level inference model derived is given in figure 4.7.

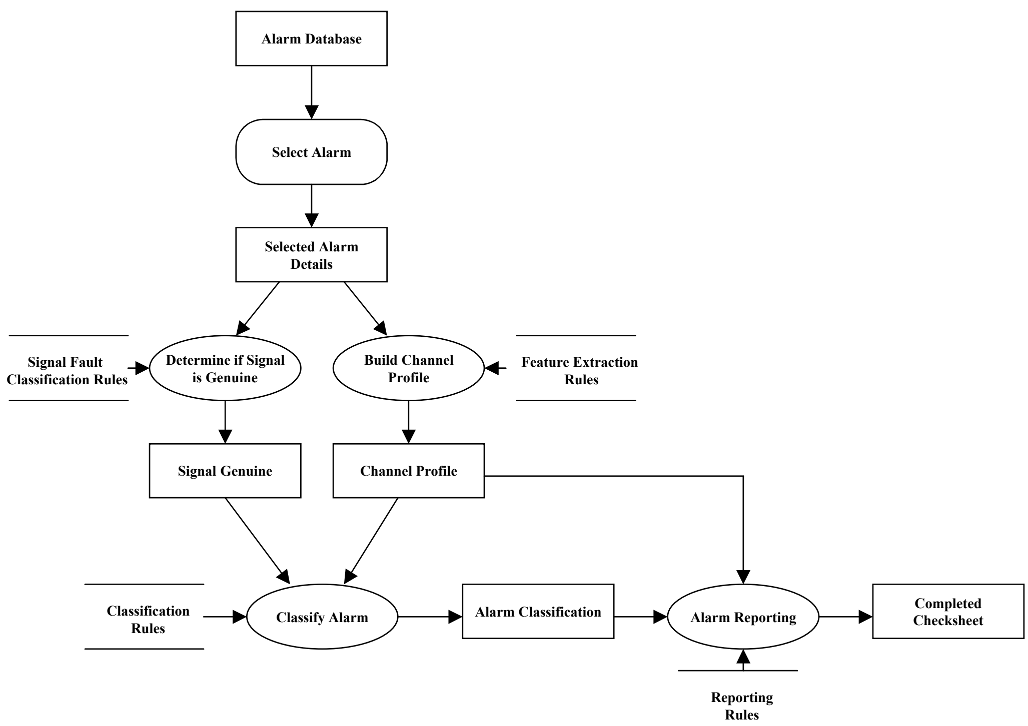


Figure 4.7: High level inference model for the turbine generator alarm assessment task

The inference model outlines the reasoning approach which must be undertaken to complete the tasks outlined in the task model. The high level tasks in the task model become inferences in the inference model. Therefore the ‘Determine if Signal is Genuine’ task in figure 4.2(a) becomes an inference in the inference model. The inference model outlines the flow of data between each inference, the ordering in which the inferences are undertaken and the knowledge required to complete the inferences.

The inference models can be constructed using four main component types as defined in [Schreiber et al, 2000] and shown in figure 4.8. The first is an inference which effectively denotes a reasoning step. Therefore, as already described above, the model will contain inferences (sometimes referred to as inference steps) which relate back to the tasks outlined in the task model such as ‘Determine if Signal is Genuine.’ Inferences are denoted in the model as ovals as shown in figure 4.8. The next component type is a dynamic knowledge role. This is a type of input or output knowledge to or from an inference who’s value can vary depending on the instance. For example, a dynamic knowledge role input to the ‘Determine if Signal is Genuine’ inference may be to give details of the triggered alarm under investigation, and the output would be if the signal is genuine or not. The information both input to and output from the inference will vary depending upon the alarm. Dynamic knowledge roles are denoted by rectangles in the inference model as shown in figure 4.8. Another component type is a static knowledge role which is an inference input that remains constant regardless of the instance. For example the ‘Determine if Signal is Genuine’ inference step may require knowledge on how to determine if a channel is exhibiting faulty characteristics. This knowledge will remain constant regardless of the alarm being examined. Static knowledge roles are represented by two parallel lines in the inference model as shown in figure 4.8. The final component which can be used in the inference model is known as a transfer function. A transfer function is where the reasoning process must interface with the external world to capture some information. Transfer functions are denoted by rounded rectangular boxes as shown in figure 4.8. In figure 4.7 the ‘Select Alarm’ task from the task model in figure 4.2(a) is represented as a transfer function since the expert captures the alarm information from the condition monitoring system for analysis.

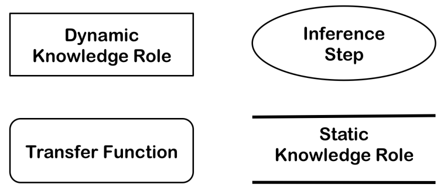


Figure 4.8: Component types which are defined by the CommonKADS methodology for inference models.

The first task undertaken is to select an alarm for analysis. This is achieved by inspecting the alarm database on the Beran system and from this selecting the alarm and recording its details such as time of triggering, turbine, etc. The alarm details are then used to select the data required to complete the 'Determine if Signal is Genuine' and 'Build Channel Profile tasks'. Both of these tasks require the input of expert knowledge in order to complete their objective. Knowledge on how to analyse the real-time FFT signals, such as how to interpret spikes and at what frequencies, are applied by the 'Determine if Signal is Genuine' reasoning step. Knowledge on how to extract features and key parameters are used by the 'Build Channel Profile' reasoning step. An indicator of whether the signal is genuine, or not, in addition to the channel profile are used as inputs to the 'Classify Alarm' reasoning step. Diagnostic knowledge is then applied to the information to determine a classification for the alarm. This classification is then passed to the alarm reporting inference step along with the channel profile to compile the alarm report. Knowledge on alarm reporting is used to distinguish the relevant information required to complete the check sheet given in figure 2.5. Further inference models were also developed to depict the reasoning process for all of the tasks defined within the assess routine alarms inference model. These models dissect each high level task to add more detail to how they are achieved by highlighting the flow of data, the steps involved and the knowledge required. All of the low level inference models developed for the alarm assessment task are given in appendix A.

4.3.3 Domain Knowledge

In addition to the task and inference knowledge derived by the knowledge engineering approach, domain knowledge which details information which is specific to the task being undertaken was also captured and recorded into the knowledge transcripts. This knowledge was not transformed into models since it could be represented and better understood as production rules. Each production rule developed related to one of the alarm classifications which resulted from the elicitation process. These alarm classifications are summarised in figure 4.6. A complete review of the production rules derived is given in section 4.5.4 and 4.5.5.

4.4 System Design

Section 4.3 reviewed in detail the results obtained from the knowledge engineering exercise undertaken with the three British Energy condition monitoring experts. These knowledge models are constructed with the human expert acting as the reasoning agent. The final stage of the design process was to adapt these models so the condition monitoring alarm assessment could be undertaken by an automated reasoning system.

There were some issues associated with the knowledge models developed for the human reasoning approach which led to changes in the approach developed for the automated system. The first issue was the availability of FFT data. The Beran system was able to save the time series data to excel files which allowed interpretation programs to be easily developed to read that type of data. The FFT files saved by the Beran system were however encrypted and therefore no program could be developed to interpret this type of data. British Energy were reluctant to involve Beran at such an early stage to determine how the FFT files could be decoded which meant that FFT data was not available for the automated system to analyse. The result was that all automated feature extraction on the

FFT data was not included in the Expert System prototype. Therefore, the ‘Extract Real-Time FFT Characteristics’ and ‘Extract Channel Frequency Characteristics’ tasks in figure 4.2 were unable to be implemented. Instead, the prototype developed asks the user to enter some of the key FFT features through a manual inspection of the FFT data and was represented as ‘Manually Enter FFT Data’ in the task model given in figure 4.9. This allowed the diagnostic knowledge which interprets the FFT features to be included in the automated system. The manually entered FFT data was solely used in the automated system to determine if the signal was genuine or not. Therefore ‘Determine if Signal is Genuine’ is the only remaining task in the automated system task model in figure 4.9 which utilises FFT information.

The ‘Retrieve Low Resolution Data’ and ‘Retrieve Buffer Data’ were brought under the ‘Select Data’ task with the ‘Select Alarm’ task. Also, both of the data collection tasks are only referenced once since the automated system collects the data at the same time. This is in contrast to the expert who may look at various data sets at different stages in the analysis process. The machine states are derived from the low resolution data in the automated system as opposed to the machine state log used in the manual process. This allowed the ‘Determine Machine States’ task to be moved directly under the ‘Build Channel Profile’ task. The revised task model for the automated system is given in figure 4.9.

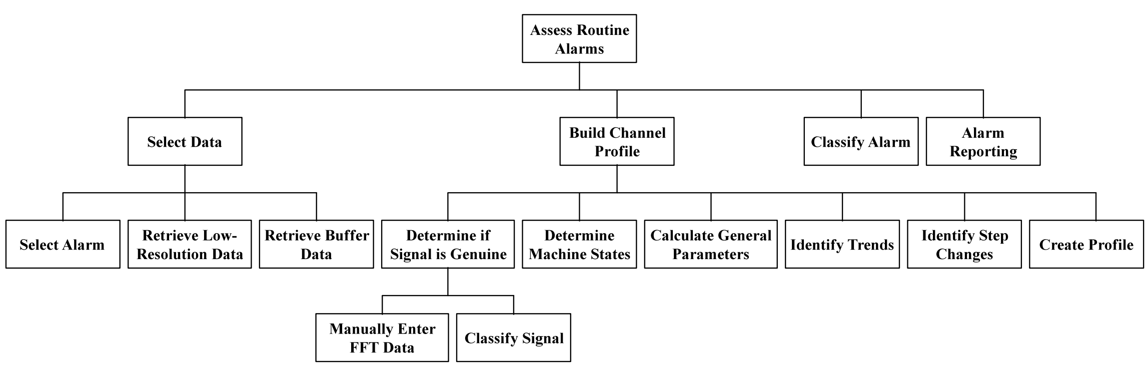


Figure 4.9: Alarm assessment module high level task model.

From this a high level inference model was developed which outlined the reasoning steps undertaken by the Expert System. The high level inference model is given in figure 4.10. The alarm database acts as the input data to the Expert System. The 'Select Data' inference, along with rules which determine what information is required to perform the assessment, are used to capture the required alarm data. This is then passed to the 'Build Channel Profile' inference, which uses rules to derive the channel profile. This profile is then used to determine the alarm classification by the 'Classify Alarm' inference. Both the alarm classification and the channel profile are the input to the 'Alarm Reporting' inference to construct the alarm report.

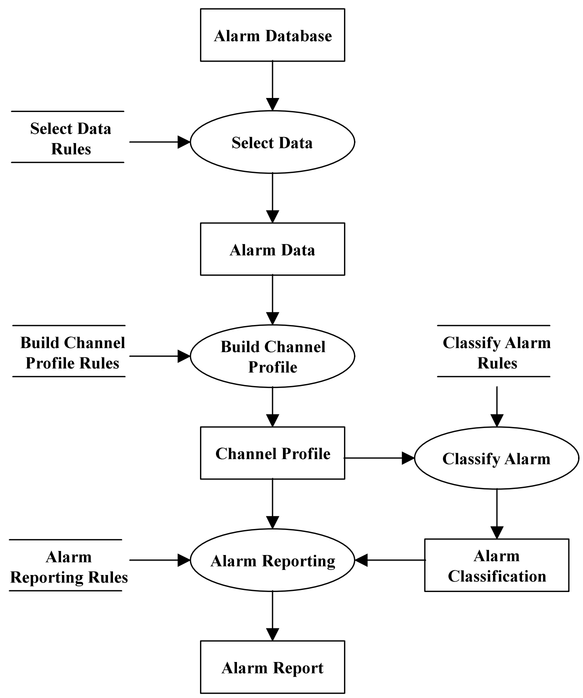


Figure 4.10: Alarm assessment module high level inference model.

The Expert System approach given in the inference model in figure 4.10 can be summarised by the system flow diagram given in figure 4.11. The Expert System can be divided into the three modules 'Select Data', 'Extract Channel Profile' and 'Classify Alarm'. Each module has an inference, which performs the required reasoning, and a knowledge base which contains the necessary knowledge, in a rule based format, required to perform the analysis.

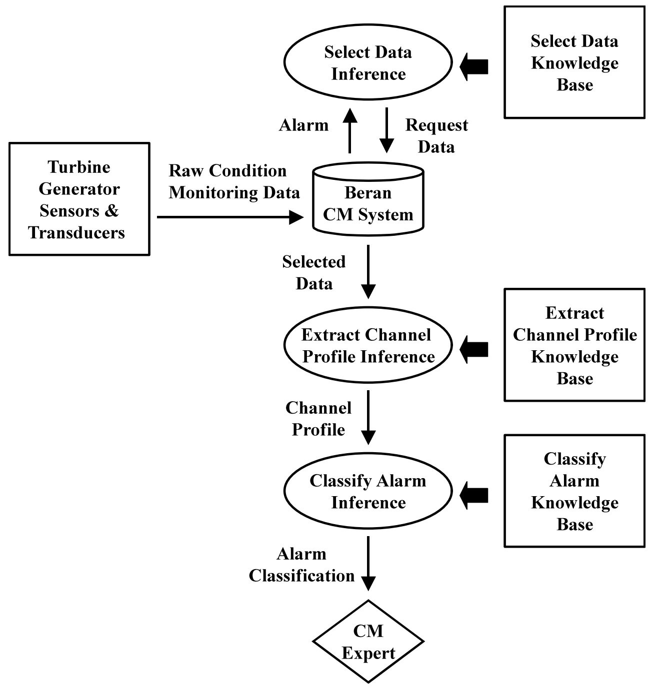


Figure 4.11: Overview of alarm assessment Expert System architecture.

The 'Select Data' module interfaces with the Beran system to retrieve details of triggered alarms. Appropriate signal types for further analysis are determined using the following alarm information:

- Type of alarm triggered (e.g. overall or 1st order vibration alarm)
- Position of the Beran channel (e.g. front low pressure rotor)

Experience gained by the condition monitoring experts had revealed that certain alarm types require specific condition monitoring data. For example a sub-synchronous high alarm will require an analysis of sub-synchronous magnitude and phase whereas operational parameters, such as generator load, take precedence for vector and zone type alarms. Equally, the Beran channel position can influence the data selection process. For instance, steam parameters, such as governor valve positions, are selected ahead of further operational data, e.g. rotor current, when the alarm triggers at the high pressure end of the turbine. The module must also determine the various data representations required for analysis. In general, all data representations are used for further analysis except in cases where buffer data is not available. This situation can arise when buffer data is not saved for certain alarm types.

The 'Extract Channel Profile' module locates relevant features from the condition monitoring data and represents them qualitatively as symbolic data objects. These features are then used to instantiate rules within the 'Classify Alarm' module. Each feature was defined mathematically through consultation with the condition monitoring experts by developing a standard model for each feature and then defining each of the key descriptors associated with that model. Specific instances of the features could then be defined by varying the limits of the descriptors in the model. This approach to feature extraction is covered in greater detail in section 4.5.3. This method of data extraction was selected since the rationale behind each feature representation is transparent to the user and limits used to define features are easily amendable. To gain user confidence in such a strategically important application requires that the rationale behind any decision is traceable. This is equally true for tasks such as feature extraction. The user should be able to track through the analysis and easily locate where the system has incorrectly interpreted the data in the event of any misdiagnoses. The knowledge can be updated accordingly once the cause of such a misinterpretation is located. Turbine generators are highly complex dynamic systems which result in a multitude of various operating regions throughout the system. Therefore, flexibility in defining the same feature over multiple channels must be facilitated. This is achieved by varying the limits used to define each feature accordingly. The 'Extract Channel Profile' module consolidates all of the features to form the channel profile once all the raw data analysis is complete. This is then passed to the 'Classify Alarm' module for further analysis.

The 'Classify Alarm' module uses the feature profile to instantiate the diagnostic rules within the ‘Classify Alarm’ knowledge base. The heuristic rules are arranged hierarchically into different categories of alarm cause. The four main categories are given in figure 4.6 and are summarized as:

- Signal Fault – Commonly caused by electrical noise, zero or abnormally high sensor readings.
- Operational Change – A change in an operational parameter such as load, rotor current or MVArs which has caused a temporary change in the vibration characteristics.
- Drift – A slight temporary change in vibration characteristics due to an unknown cause.
- Genuine Change – A permanent change in vibration characteristics due to an unknown cause or a genuine fault.

The assessment conclusion along with a report, which includes the rules instantiated during the assessment and all of the completed checksheet entries are output to the expert for verification. The expert can view any of the condition monitoring signals which were used in the diagnosis. All of the features located by the ‘Extract Channel Profile’ module are highlighted to augment the experts understanding of the alarm analysis. Restrictions on access to the FFT data on the Beran system meant that information on the frequency characteristics of the data had to be entered manually therefore a method in which the user inputs this to the system had to be developed.

4.5 Software Implementation

The software implementation of the Expert System had to be realised in many different stages. The first stage dealt with how the necessary data would be captured from the Beran System. This would fulfil the ‘Select Data’ task outlined in the system task model in figure 4.9. Once the data is captured by the Expert System the information of interest must be derived from the raw data i.e. the system would have to perform the analysis required to fulfil the 'Build Channel Profile' task in figure 4.9. This task is achieved by the system feature extraction where information such as the trend profile of the signal and step changes are derived. The transformed data then has to be passed to the working memory to be processed by the rules in the knowledge base to fulfil the 'Classify Alarm' task in figure 4.9. The knowledge base determines modes of behaviour by analysing the extracted features and then uses this information to determine the conclusions in which the expert user is interested in. Some of the modes of behaviour which are of particular interest are the stability of the signal and how the signal varies over particular time periods. These modes of behaviour then dictate how the results are calculated and presented to the expert to complete the 'Alarm Reporting' task given in figure 4.9. Central to the design concept of the Expert System is the user interface which attempts to relay all of the important information back to the user. This is achieved by highlighting the features extracted by the system at the signal to symbol transformation stage as well as outputting the decision tree which determined the alarm cause. These tools allow the user to visually verify that the system is picking up on relevant features and then using these features to arrive at valid conclusions. Each of these stages required of the automated implementation of the prototype Expert System are now reviewed in the remainder of this section.

4.5.1 Data Capture

The first stage of the automated alarm assessment process is to retrieve the necessary condition monitoring data from the Beran system. This relates to the 'Select Data' task given in figure 4.9. In practice this task would be achieved by the Expert System completing the 'Select Alarm' task in figure 4.9 by interfacing with the Beran system to detect any triggered alarms which have not been analysed by the system. Once an alarm has been selected the relevant condition monitoring data can be retrieved from the Beran system raw data archive. The two data types captured are the low resolution and buffer data as described in section 2.2. The capturing of both data types would complete the 'Retrieve Low Resolution Data' and 'Retrieve Buffer Data' task in figure 4.9. The selection of data retrieved is dependent on the type of alarm and the channel position on the turbine. Due to restrictions on how the Expert System was allowed to interface with British Energy's on-line system, the prototype developed was unable to automatically search for and retrieve data from the Beran system. Instead, historical case studies were selected by the experts and the relevant condition monitoring data was manually captured and then uploaded to the Expert System.

4.5.2 Data Entry

The tasks which involve analysing the frequency data have been omitted from the prototype due to the unavailability of the FFT data at the time of writing this thesis. It was therefore only possible to include a task whereby the information on the channel frequency distribution is entered manually. The information required of the user was as follows:

- Are there spikes in the FFT at multiples of the operating frequency? – This feature can help identity signal faults and looseness of components.
- The shape of the frequency spikes, if there are any, are they broad or narrow? – This type of information can help differentiate between certain faults which exhibit similar characteristics.
- Whether there is base noise in the FFT? – This can help diagnose a noise fault in the transducers.

Once the condition monitoring data has been uploaded to memory, the user is asked to answer yes or no to the above questions to complete the 'Manually Enter FFT Data' task given in figure 4.9. These answers are transformed into qualitative symbols and uploaded to the system's working memory to determine if the signal is genuine or not by applying rules AC1 and AC2 given in appendix B. The application of this knowledge fulfils the 'Classify Alarm' and hence the 'Determine if Signal is Genuine' tasks in figure 4.9.

4.5.3 Build Channel Profile

The next task undertaken once the data has been extracted from the Beran system is that of 'Build Channel Profile'. This task is essentially the signal to symbol transformation of the system. The data is analysed by the system in an attempt to draw out features which can help identify the cause of the alarm, in the same way that the human experts will analyse the data plots on the Beran interface to draw out features which are common to certain behaviour types. Algorithms were developed to extract trends and step changes from the times series data and complete the 'Identify Step Changes' and 'Identify Trends' tasks outlined in figure 4.9. These features are extracted using standard models of step changes and trends defined by the experts. Associated with each model are parameters which make up a complete description of the feature. These parameters can have their limits varied to allow the location of features over various channels which exhibit different operating points from one another. These processes are described in detail in this section.

Each turbine generator set has a channel for every bearing on the unit. Therefore the number of channels per turbine generator ranges from 10 to 12 depending on the configuration of the unit, i.e. 2 bearings per HP, MP and LP turbine, and generator. Each channel will therefore have its own overall amplitude vibration signal and the associated first and second order magnitude vibration data, FFT, etc. The algorithms developed are the same for each channel on the turbine generator set but the limits associated with the features for each channel are stored in their own individual rule base. Therefore, if an alarm from channel 2 on turbine 1 was being analysed then the limit rule base for that particular channel would be run to upload all the relevant limit values. The system must also interpret the machine state to complete the ‘Determine Machine State’ task given in figure 4.9. The derivation of the machine states is described in this section. The two remaining tasks under ‘Build Channel Profile’ in figure 4.9 are ‘Calculate General Parameters’, which is the derivation of the signal spread and mean in addition to some other parameters of general interest, and finally ‘Create Profile’ which compiles all of the aforementioned features into one profile.

4.5.3.1 Trend Profile

One feature of the data which is of interest in the alarm assessment analysis process is the gradual gradient profile of the signals. Of interest are gradual increases or deceases in the data or periods of stability. The term commonly used to describe this type of feature is trend. The trends present in each of the times series data signals must be extracted as indicated by the ‘Identify Trends’ task in figure 4.9. The trend parameter is focussed more on describing general fluctuations which take place over extended time periods as opposed to sharp changes. These features are of interest since many alarms can be explained by correlating increases in vibration signals against causal operational events. It is important to clarify as clearly as possible the type of descriptions compiled by the experts during their analysis before explaining how this signal to symbol transformation is achieved by the Expert System. The overall amplitude signal given in figure 4.12 is typical of the type of data analysed by a condition monitoring expert.

A typical description given by an expert of such a signal during the knowledge elicitation interviews was:

“The first half of the signal is relatively stable with maybe a slight downward trend. Then there is a period of noisy data but it still maintains a relatively stable average. This is followed by a short period of stability which soon encounters another period of noise.

This second period of noise is combined with a gradual upwards trend which continues on throughout a period relatively free of noise before stabilising."

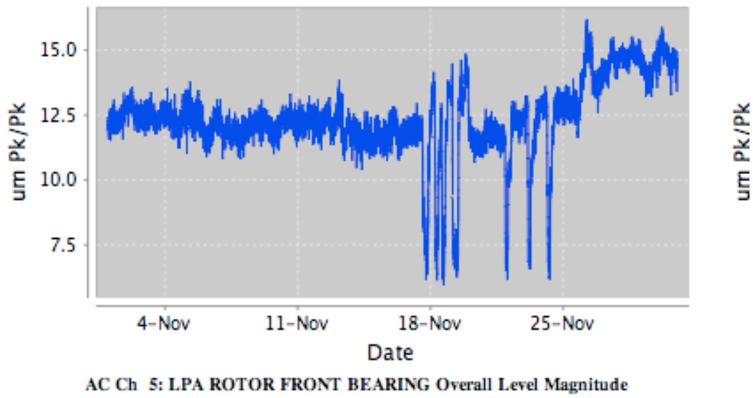
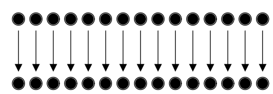


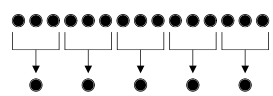
Figure 4.12: Overall amplitude signal taken from the Beran system.

This type of interpretation is what comes naturally to humans, so it wouldn't take an expert in condition monitoring to describe the above signal in those terms. It bears a strong similarity to how most people would describe such a signal. The description is very approximate and makes quite significant abstractions in order to provide a succinct and easy to understand description. This form of approximate human interpretation is no menial task for an automated computer system to perform. One of the primary difficulties is the level of noise contained within the signal which makes it difficult for any algorithm to follow a general trend. Another difficulty is the lack of definition available to describe these gradual changes. Estimations, by virtue, lack any form of definition. For these reasons any trend profile description derived by an automated system will not correlate exactly with that derived by the human expert but it should be possible to acquire an initial approximation. There now follows a description of the approach adopted by the Expert System to extracting an overall description of the trend profile.

The basic idea behind the trend extraction algorithm is to break the signal into periods and from this determine the type of behaviour exhibited in each. Once this is complete the results can be combined in order to construct a more complete description of the trend profile. The first task is to break the signal down into the specified number of distinct periods. This number is determined by the trend period size. Using this value and the file size, the number of periods to be set is derived. For example, the signal in figure 4.13(a) has been broken down into 5 periods of size 3 in figure 4.13(b). This period was set to 1 day in the prototype Expert System developed. Various periods were experimented with and output to the user interface for the expert to review. The expert felt that the period of 1 day produced a trend profile description which correlated closely with a manual analysis.



(a)



(b)

Figure 4.13: Samples of the continuous signal are broken down into periods of a pre-defined size

Each period then has its average, minimum, maximum, spread, end average and start average calculated. The end average is the average magnitude over a pre-defined number of samples at the end of the period. Similarly, the start average is the average magnitude over a pre-defined number of samples at the start of the period.

These derived parameters are then used to build up a description of the period. The trend description consists of two primary components. These are the level in which the period resides and the trending profile of the period i.e. is it increasing, decreasing etc. The description of the level in which the period resides is simply calculated by comparing its average against the predefined trend level limits assigned in the channel's limit rule base. If the average falls below the lower limit then the period level is labelled as low, if the average is above an assigned limit then it is labelled as high otherwise it is stable. This is highlighted in figure 4.14.

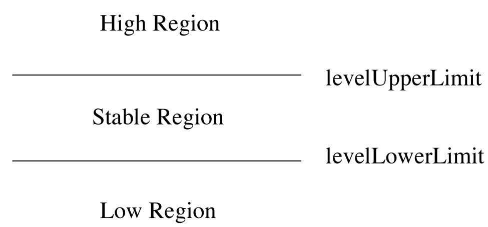


Figure 4.14: Assignment of level value in trend description.

The period's trend profile descriptor is calculated using the period start average and end average along with some tolerance values set within the channel's limit rule base for each signal type. The rules then determine if the signal has trended upwards, downwards or remained stable during the period. A tolerance value is set in the assign limits rule base which assists in determining whether the period has increased or decreased in value or has remained stable. If the end average is less than or equal to the start average plus the tolerance value and greater than or equal to the start average minus the tolerance value then it is labelled as stable as shown in figure 4.15(a). If the end average is greater than the start average plus the tolerance then it is defined as increasing as shown in figure 4.15(b). If the end average is less than the start average minus the tolerance then it is defined as decreasing as shown in figure 4.15(c). The overall signal is broken down into periods, which have trend profiles of either increasing, decreasing or stable and have average magnitude levels which are either high, low or stable once this interpretation has been completed.

On completion of this task the trend profile descriptors are combined to give general descriptions of the signal. For example any extended periods of an increasing signal should be identified by consecutive periods with a trend profile descriptor of increasing value. It should also be possible to identify areas where the magnitude of the signal is particularly high or low.

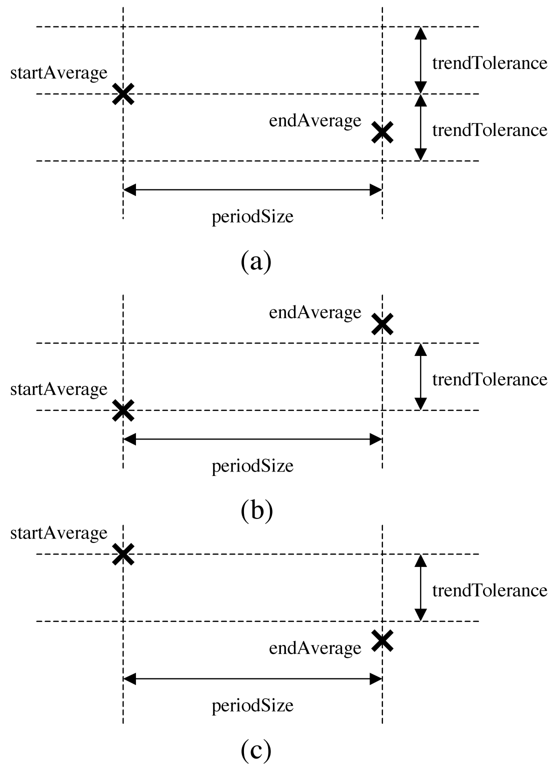


Figure 4.15: Derivation of the trend profile behaviour. Figure 4.15 (a) shows a trend period, which would have its trend profile descriptor set to stable. Figure 4.15 (b) shows a trend period, which would have its trend profile descriptor set to increasing. Figure 4.15 (c) shows a trend period, which would have its trend profile descriptor set to decreasing.

4.5.3.2 Machine State

The next stage in the 'Build Channel Profile' task model is to determine the machine state as indicated by the 'Determine Machine State' task in figure 4.9. The machine can be in one of three states. These are online, run up/down or offline. The machine is online when it is rotating at or close to 3000rpm. The machine is offline if it is rotating less than a pre-defined level of 50rpm. If the machine is rotating at a speed within these limits then the state is set to run up/down. These limits are set within the channel's assign limit rule base. The rpm values which are compared against the limits are averages taken over pre-defined periods as performed for the derivation of the trend descriptors. Therefore a period size relating to the machine state is assigned within the limits rules base. The state of each period is determined by taking the rpm average over each period and then comparing this value to the limits. This approach to determining the machine state is depicted in figure 4.16.

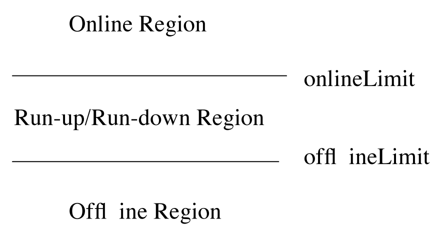


Figure 4.16: Derivation of machine state.

4.5.3.3 Step Changes

The next stage in the 'Build Channel Profile' task is to extract any step changes which may exist in the file. This analysis is required to complete the 'Identify Step Changes' task given in figure 4.9. Unlike the calculation of the general parameters, extraction of trends and the determining of the machine state, the steps are extracted through the use of an algorithm which is coded into the automated system as opposed to the use of a rule base. The reason for this is the number of rule bases which would be required to extract the features used for the step change analysis. Another reason for hard coding this task into the automated system is due to the approach being more algorithmic as opposed to rule based. Before describing the features of the algorithm used to extract step changes, it is necessary to provide a definition. Figure 4.17 depicts some of the important features associated with both an upwards and downwards step change.

Two of the most obvious parameters define the steepness of the change which must occur in order to be a step change. These two parameters are the step period and the step magnitude and they are set within the assign limits rule base. Any changes which exceed the step magnitude within the step period are defined as potential step changes. In addition to exhibiting a sudden change in magnitude a step must also show a degree of stability in both the period before and after the change occurs. The period before the change occurs is called the lead and the period following the change is called the tail. Both of these are set within the assign limits rule base. If the change is downwards then the magnitude of the data contained within the step's lead period should be above a certain limit and the data within the tail period should be below a certain limit as shown in figure 4.17(a). The opposite of this applies if the change is upwards as shown in figure 4.17(b). These limits are calculated using the appropriate tolerance values which are set using the assign limits stage. These limits account for the fact that the sensor signals oscillate somewhat due to the natural noise levels in the channel. The number of samples which fall within these limits in both the lead and tail are then compared against a value which denotes the minimum number of samples required in each to constitute a step change. These limits are calculated using further tolerance values which are set at the assign limits stage. The purpose of this limit is to account for noise spikes which may occur in the data even when the data is relatively stable.

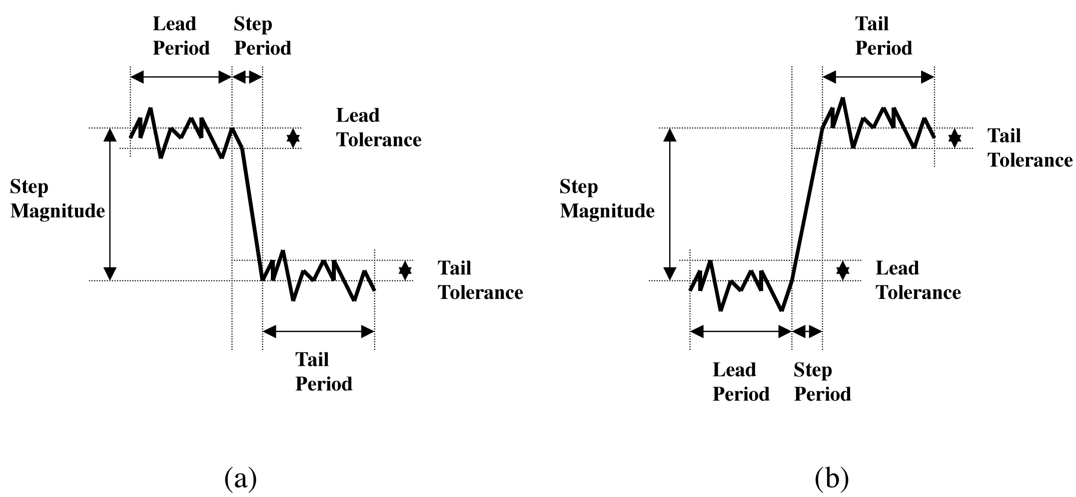


Figure 4.17: Definition of decreasing and rising step changes within the Expert System.

4.5.3.4 Compile Profile

The final task of 'Build Channel Profile' in the task model in figure 4.9 is 'Compile Profile'. The purpose behind this task is to collate all of the information derived from the previous tasks into one profile which describes the channel. Therefore this task is achieved by collecting together all of the features and information derived during the previous tasks and passing this information onto the 'Classify Alarm' task.

4.5.4 Compile Channel Description

The next stage is to compile a description of the behaviour of the channel. In practice, this task requires a two pronged approach which entails completing all of the check sheet entries currently completed manually by the experts and using this along with some additional information to deduce what caused the alarm to trigger. An example of the check sheet required to be completed by the experts is given in figure 2.5.

All of the information is derived using expert diagnostic knowledge captured at the knowledge elicitation stage and recorded in the knowledge transcripts. The knowledge contained within these transcripts for the most part is expressed in general terms used by the experts. In order to transfer this knowledge onto an automated system it is necessary to add a significant level of detail to the knowledge definitions mostly through adding knowledge which the experts may have overlooked due to its procedural nature. For example, some of the knowledge contained within both transcripts describes operational changes occurring close to the time of alarm firing. The word close must be defined in more detail so as an automated system can recognise whether the change occurs closely. Another example is when compiling a description of the general signal behaviour like completely stable, generally stable, noisy etc. These terms required a more exact definition in order to be deducible from the condition monitoring data.

4.5.4.1 Vibration File Stability

A large percentage of the rules contained within the rule base are concerned with determining events which occur in certain files. Six vibration files are assessed: overall amplitude; first order magnitude; first order phase; second order magnitude; second order phase; and sub-synchronous amplitude. They are assessed to determine the overall stability of the signal, whether any changes have occurred within various periods of interest and what the level of vibration was in particular regions. The overall stability of each vibration parameter is defined in terms of one of the following categories:

- Completely Stable – 100% of trend periods have a change state equal to stable.
- Relatively stable – more than or equal to 70% and less than 100% of trend periods have a change state equal to stable.
- Unstable – less than 70% of trend periods have a change state equal to stable

The noise status of each vibration parameter is defined as either noisy or not noisy. A noisy signal has over 30% of its trend periods in the change state equal to noisy. A not noisy signal has less than or equal to 30% of its trend periods in a change state equal to noisy.

Each of the six vibration files have been assessed with respect to stability using rules *file*13 to *file*17 in appendix B.

4.5.4.2 Vibration Events

There are also rules in place which determine if any events such as trends and step changes occurred within a certain period as well as what the state of the level descriptor was within these predefined periods. The vibration data is analysed to determine if there is a high or low value in the phase or magnitude level following the alarm as given in rules *file*1 to *file*3 in appendix B. Another piece of information recorded is whether the signal was within the stable level region before or after the alarm was fired as given in rules *file*4 to *file*7 in appendix B. There are also rules which determine if a change in any of the periods occurs within the period after the alarm or if the change descriptors were of a purely stable nature as given in rules *file*8 to *file*12 in appendix B.

4.5.4.3 Vibration Check Sheet Entries

The next step is to calculate the values of the vibration parameters ‘OA Amp’, ‘1x Amp’, ‘1x Phase’, ‘2x Amp’, ‘2x Phase’ and ‘Non-synchronous Amp/Freq’ entries, which are placed in the checksheet. The rules used to calculate the correct value for each of these entries are the same but the values and events utilised within the rules are derived from the appropriate file. The rules use the information on the file stability and the vibration events to determine the checksheet entries. There are 4 variables associated with the checksheet entry of each file. A value is calculated for one or more of these variables depending on the behaviour of the signal. The first 2 of these variables denote the file spread by storing the minimum and maximum recorded values from the data. These values are only recorded for instances where the file has been deemed noisy. This is implemented by rule *file*25 in appendix B. If it has been determined that a low or high level has been read after the alarm has triggered, and the signal is within the stable region before the alarm triggered, then it is necessary to record both the higher and lower level and store these as two overall amplitude values for the checksheeet. This is implemented by rules *file*19 & *file*21 given in appendix B. In all other situations a single value is calculated in rules *file*18, *file*20, *file*22, *file*23 & *file*24 given in appendix B.

The overall amplitude value derived is used to determine what is entered into the ‘Zone’ entry. The overall amplitude level calculated is checked against the zone limits to determine if the channel falls within zone 1, 2, 3 or 4. Rules CH7 to CH14 in appendix B are used to determine this value.

Another entry calculated using the vibration information is the 'OA ~ ∑(1x+2x)?' entry. The values calculated for first order magnitude and second order magnitude are added together to determine if they are approximately equal to the overall amplitude value. The overall amplitude variable is multiplied by a tolerance factor to determine a range of values for the approximation. Rules CH15 to CH17 in appendix B are used to determine the result for this entry.

4.5.4.4 Vibration Step Events

Also completed are the entries which indicate whether a step change has occurred in both the first and second order signals. When undertaken manually, the experts analyse the magnitude and phase to determine if a sudden change occurs. Currently the automated system only analyses the magnitude files for a step change because the nature of the phase readings can result in false step change readings. For example, if the phase is situated close to the 180° point then a movement of only a couple of degrees can give a reading of approximately -180°. Although the change would be of a small magnitude, this could be mistaken for a large magnitude change, if the difference between the points is calculated by simply subtracting one angle from the other. Therefore the 'Step Δ1x?' and 'Signif. Δ2x?' entries are completed by checking if any steps were extracted at the 'Extract Step Changes' stage within the first and second order magnitude files respectively. Rule *file*27 determines this for both the 1st and 2nd order magnitude files.

Since the development of the system, a potential solution to identifying whether large changes in both the 1st and 2nd order phases has been considered. The proposed solution would be to determine the difference in magnitude between the vibration points on the 2-dimensional magnitude and phase plot. This could be achieved through the use of trigonometry and rules which determine when to add and subtract the distances calculated on the x and y magnitude axes calculated between each point. If it is determined that there is a large difference in the 2 dimensional magnitude and phase plot then the system can firstly look towards the difference in vibration magnitude to determine if this is the contributing factor. If this is not the contributing factor then it can be deduced that a large difference in the vibration phase is the primary contributing factor. This additional functionality will have to be researched, tested and added to any future implementations of the Expert System.

4.5.4.5 Operational Events

The next stage is to determine what events occurred in the operational signals. The only operational parameters which have been processed by the automated system to date are the Generator Load, Generator Rotor Current and the Generator MVArs since these were the only files identified as being necessary for the analysis. The key events of interest here are whether any changes have occurred (steps or trends) within a specified period before the alarm was triggered. The main purpose of this analysis is to determine if any changes in the operational parameters could explain corresponding changes in the vibration parameters which lead to the alarm triggering.

The first feature searched for is whether a trend change has occurred within the specified period or none at all. This change can be of either an increasing or decreasing nature. Rules *op*1 to *op*3 in appendix B are used to determine if any of these changes occurred. A similar process is undertaken for determining if any steps have occurred within the specified period using rules *op*4 to *op*6 given in appendix B. The 'Operational Change?' entry can be completed once this information has been derived. Each of the three operational files Generator Load, Generator Rotor Current and Generator MVArs are checked to determine if any steps or trends occurred within the specified period. If so then the operational change variable is set to true. This checksheet entry is completed by rules CH1 to CH6 in appendix B.

4.5.5 Classify Alarm

The next stage is to classify the alarm according to one of the alarm classifications defined in figure 4.6.

4.5.5.1 Signal Faults

Rules are in place to determine if any signal faults have occurred on the channel. The four sensor fault types identified in the knowledge transcripts were 'Electrical Noise', 'Zero Sensor Reading', 'Excessive Sensor Level Reading' and 'Zero Rpm Reading' as given in figure 4.6.

The electrical noise fault can be identified in two ways. The first method is to determine if there are spikes on the channel's FFT at multiples of the operating frequency. The second feature which can identify an electrical noise fault is a high level of base noise in the FFT data. The rules which diagnose these faults must rely on the information entered manually at the data entry level as described in section 4.5.2 since the FFT is currently unavailable in a suitable format for analysis. The two rules which interpret the data entered manually are AC1 and AC2 in appendix B.

The overall amplitude data is analysed to determine if either a zero sensor or excessive sensor level error has occurred on the channel. The associated zero sensor reading rule AC3 in appendix B checks the raw data for any data samples within a specified period which fall below a set limit and that the system state was not offline at the time of alarm firing. The excessive sensor level rule AC4 in appendix B analyses the data in a similar fashion except the samples are checked against a limit which denotes an excessive level. Finally the rpm data is analysed using the same approach as used for determining if the overall signal read zero by AC5 in appendix B.

The remainder of the rules are concerned with checking for alarm causes which are in some way related to changes in the vibration signals. The three main categories for these causes are 'Operational Change', 'Drift' or 'Genuine Change'.

4.5.5.2 Operational Changes

An operational change is where some vibration movement has been caused by a change in one of the three operational parameters already discussed. A key characteristic of the operational change is that the level of change is not large. Therefore the variables of interest are:

- Whether a change in one of the operational parameters has occurred?
- Has there been a corresponding change in the relevant vibration parameter?
- Has this change remained within its stable level limits following the alarm?

The vibration signal analysed is dependent on the alarm cause. Therefore if the alarm was of type '1X Vector' then the 1^{st} order magnitude and phase is analysed, if '2X Vector' then 2^{nd} order magnitude and phase, if 'Sub-Synch High' then sub-synchronous amplitude and the overall amplitude is used for any 'Zone' type alarms. Rules AC6 and AC7 in appendix B are concerned with determining this type of event.

4.5.5.3 Genuine Large Vibration Change

Another possible type of operational change would be for the relevant vibration signal to move outside of its stable level region after the alarm has triggered and return back within this region. Therefore the data is analysed to determine if a trend change or step change occurred in the operational parameter prior to the time of alarm firing. The relevant vibration signal which is again dependent on the alarm type is checked to find if a high level occurs after the alarm and if a stable level also occurs after the alarm. The analysis performed to find this type of behaviour is derived by rules AC8 & AC9 in appendix B.

4.5.5.4 Genuine Change No Recovery

A genuine change with no recovery is caused by a change in an operational parameter in the form of a step or trend prior to the alarm. This results in the vibration signal increasing to a high level and not recovering back to within its stable level region. This alarm cause is very similar to the operational change which causes the vibration to move to a high level, as described previously, only a genuine change is characterised by no recovery. The analysis performed to find this type of behaviour is performed by rules AC10 & AC11 in appendix B.

4.5.5.5 Unattributed Change

Another possible alarm cause is an unattributed genuine change. This type of alarm is characterised by a high level of vibration after the alarm which does not recover. There is however no corresponding change in the operational parameters to account for the change in vibration. This particular type of behaviour is derived by rule AC12 in appendix B.

4.5.5.6 Drift

Finally the rules analyse the data to determine if the alarm was caused due to drift. Drift is characterised by a slight change in the vibration, which moves it from outside of the alarm limits, and no apparent corresponding operational change which has caused it. Ideally a slight change in the relevant vibration should be observed by the system but in some cases the movement is so slight that it is difficult to detect due to the approximate feature extraction. Therefore the rules which detect drift only check that the vibration falls within the stable level region and that no operational change has occurred. This particular type of behaviour is derived by rule AC13 in appendix B.

4.5.6 User Interface

So far, how the Expert System captures the data and then interprets this to arrive at conclusions on the alarm cause, and particular behavioural properties of the channel has been detailed. The automated processing of the data is however only a portion of the functionality provided by the Expert System design which assists the expert in processing the condition monitoring alarms. One of the primary reasons for selecting Expert System technology was due to the transparency in the assessment rationale provided by such a technique. This transparency is traditionally provided by the tree explanation structure generated by the rules which arrived at the final conclusion. This method of transparency can be a very powerful tool if used in an appropriate fashion and is included within the Expert System design as one of two main explanation functions provided. The second explanation feature was developed when researching how novel methods of explanation could be incorporated into Expert System design. This second explanation feature aims to visually highlight to the user key features derived from the raw data which were used by the Expert System in arriving at its conclusions. It does this primarily by highlighting some of the features of key interest to the experts when building a picture of the channel behaviour. The features which are highlighted by the current Expert System prototype were selected by determining which ones would provide the most information to the user in explaining how the conclusion was deduced. Throughout the knowledge elicitation process the experts highlighted the importance of vibration and operational parameter features such as the general profile and step changes and how this information was critical to a large percentage of assessments. Therefore the interface developed provides functionality which attempts to relay to the user the key features derived and subsequently used in the alarm cause analysis process. The approach in which the developed prototype system achieves this goal is outlined in this section as is some of the other functionality provided by the interface design.

4.5.6.1 Alarm Selection

The functionality built into the Expert System interface will allow users to access the current or historical alarms from any of the Beran monitored turbines throughout all of the British Energy facilities. An example of one of the selection screens currently used within the developed prototype system in given in figure 4.18.

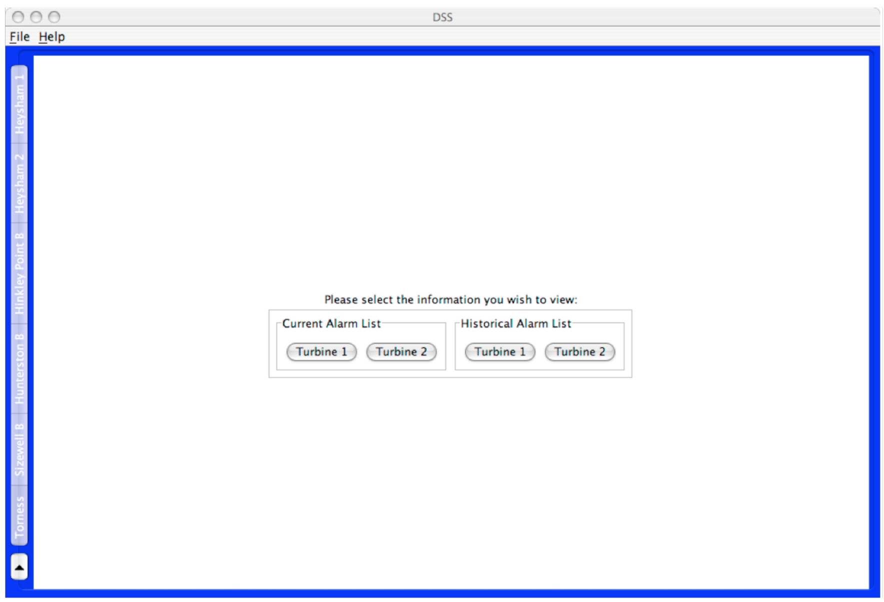


Figure 4.18: Historical or current alarm selection screen.

Here the user is prompted to select from the database of current alarms which have been assessed by the system but not yet verified or historical alarms which have been both assessed by the system and verified by the expert. The current alarm list would be the most common selection but the historical alarm list would be useful in situations where audits of previously assessed alarms would have to be carried out as is currently performed on a quarterly basis at all British Energy locations.

An example of the screen which allows the expert user to select a current alarm for verification is given in figure 4.19.

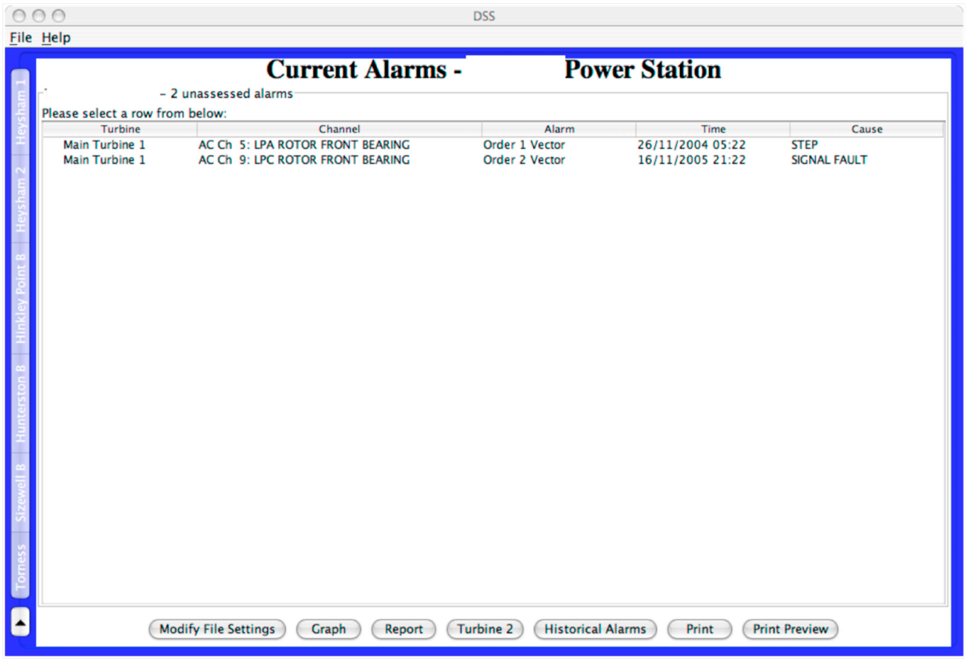


Figure 4.19: Current alarm selection screen.

Here the user is provided with the details of the triggered alarm such as the turbine it was triggered on, the channel, the date and time, the alarm type and the cause of the alarm as assessed by the system. The alarm cause determined by the system provides a natural ordering of priority. For example, alarms which have been assessed as a genuine change would be of a higher priority than those assessed as a signal fault. Therefore the pre-processing undertaken enables the experts to prioritise which alarms are assessed first and increases the chances of locating genuine problems at an earlier stage than what would be possible with no pre-processing. This is especially true in situations, where vast amounts of alarms have been triggered, and only a small percentage, allude to a genuine problem on the machine. One additional piece of functionality which can add benefit to this type of interface screen would be some form of traffic light labelling which would visually enhance the priority levels of each alarm. For example high priority alarms may be highlighted in red whereas alarms of a lower priority may be labelled green.

4.5.6.2 Assessment Results

When the expert selects the alarm in which he/she is to verify, the alarm analysis results screen, an example of which is given in figure 4.20, is displayed to the user. It is here that the system attempts to relay back to the user in a novel fashion some of the key features extracted at the analysis stage. The two primary features which the current prototype system aims to highlight to the user are the gradual trend profile of the system and any step changes. In addition to these features the checksheet entry results are displayed along with any header information.

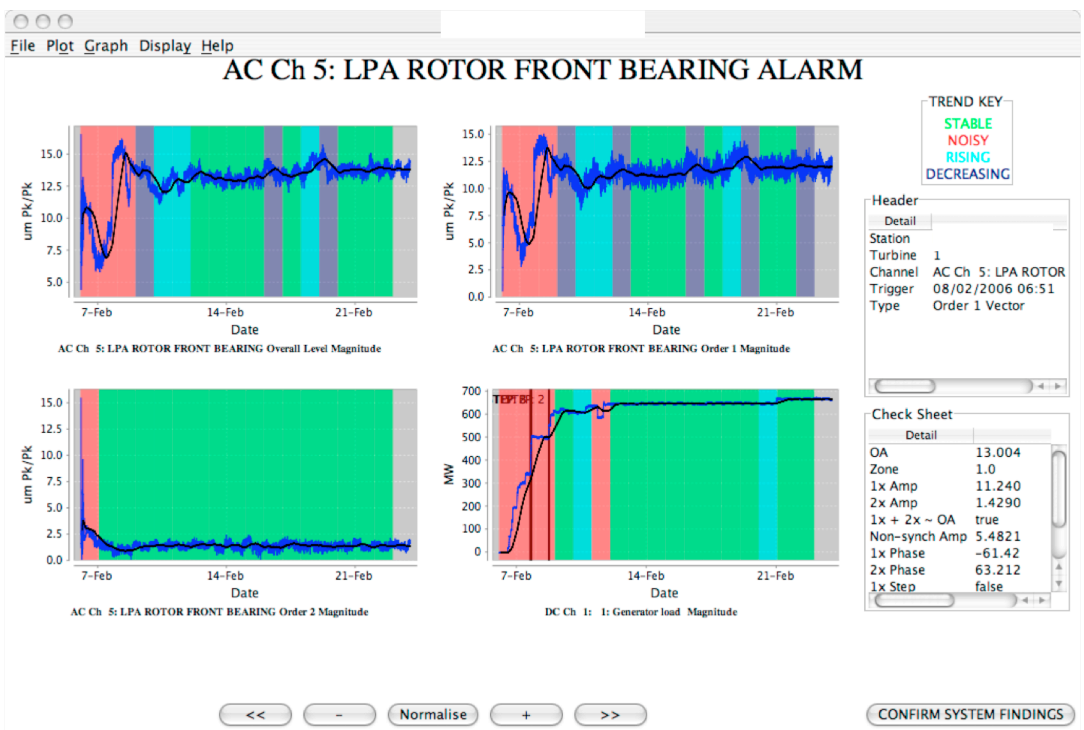


Figure 4.20: Analysis results screen.

4.5.6.3 Feature Extraction

The trend profile derived by the system is conveyed back to the user by highlighting the background of the signal with the relevant colour to indicate if the signal was increasing, decreasing, stable or noisy. The generator load signal depicted in figure 4.21 highlights how the trend profile of the signal is fed back to the user. Stable periods are highlighted with a green background, decreasing regions with a grey background, increasing areas with a light blue and noisy regions are in red. This approach to visually highlighting the trend features allows the user to instantly determine how accurate the system's analysis compares with his/hers. Therefore if the alarm cause flagged up by the system related to a gradual operational change in some parameter, the expert could quickly analyse that parameter, locate the trend change and determine if it is valid or not.

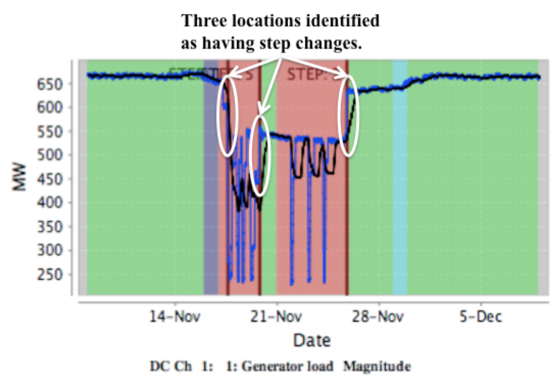


Figure 4.21: Feature extraction fed back to the user.

Another explanation facility provided by the expert system which is also shown in figure 4.21 is the highlighting of step changes which occur in the signal. The step changes are highlighted by a black vertical line running through the graph at the positions where step changes are located. It can be seen that there are three areas on the graph where step changes have been located in this particular signal. This again allows the user to instantly locate where any step changes have been found in the data by the signal to symbol transformation so that any results relating to these features in the checksheet can be quickly verified.

4.5.6.4 Assessment Rationale

Another explanation facility put into the Expert System design was the reporting of the system rationale by displaying what rules were triggered to arrive at the assessment conclusion. This tool was not fully implemented into the final prototype due to development time constraints. The current prototype outputs a list of the rules which generated the system conclusion. Each rule contains its own identifier and the identifier of the rule which fired it meaning that the tree structure of the explanation can be easily deduced manually using the existing information. The next version of the system should contain functionality which outputs the explanation in a tree structure automatically via the user interface. However some work was undertaken to enhance the information conveyed by the explanations. In order for a rule based explanation tree to be of any assistance to the expert user in verifying the system rationale, the triggered rules which appear in the explanation would have to mean something. Therefore, all of the rules placed in the knowledge base were given a description which explained the rationale behind the firing of each rule. This allows the rationale of each rule to be displayed in the explanation tree, enabling the user to verify the rationale. An example of one of the explanation trees generated by the system is given in figure 4.22.

This explanation tree indicates that this particular alarm has been triggered due to a rotor current operational change. This has been derived due to the triggering of rules which indicate that a rising change has occurred in the rotor current, a rising change has occurred in the first order magnitude and this change has not been of a high level. It should be apparent that this method of explanation allows the user to quickly determine the rationale which has lead to the conclusion. The users would firstly have to become familiar with the meaning of the rule descriptions but for the most part they are self explanatory. The system can be improved upon by generating these explanations using the tree structure format shown in figure 4.22 automatically within the user interface and then linking these explanations directly with the features which triggered these rules. For example, rule RC 1, which indicates that a rising change occurred in the rotor current would have been triggered by a feature in the rotor current data. It would add value to the system if the user were able to click on this rule and then be taken directly to the features which triggered it.

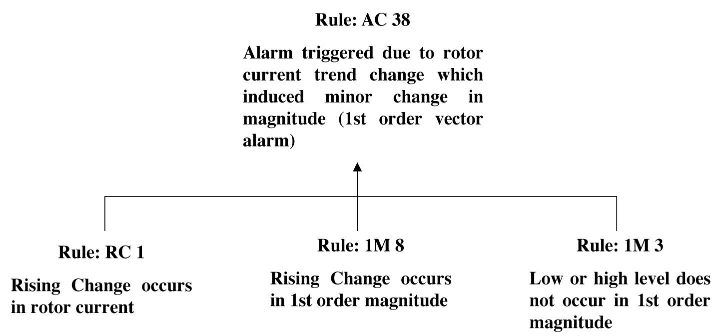


Figure 4.22: Explanation tree generated by Expert System to explain the rationale behind the alarm being diagnosed as a rotor current operational change.

4.6 System Testing

Testing has been carried out on the Expert System module. Of most interest was the analysis performed by the module on the alarm data. Therefore test data in the form of previously assessed alarms were used to assess how well the module performed. The module performance was measured against how close the checksheet results derived automatically by the prototype compared with the historical records which were completed manually. Also of interest was how well the module's feature extraction rules and algorithms performed in extracting a description of the signal behaviour. To assess this function the module's description of the signals is compared with that given by a turbine generator condition monitoring expert from British Energy.

Only data taken from turbines at one of British Energy's locations was used to test the system because the system developers were able to gain access to this more easily. Alarms from channels 5, 6, 9 and 11 were tested from turbine 1 and an alarm from channel 9 was tested from turbine 2.

4.6.1 Turbine 1

This section will review the Expert System performance at analysing each of the test case studies from turbine 1.

4.6.1.1 Channel 5

There were two alarms analysed on channel 5 of turbine 1. The first of these alarms was triggered as a 1X Vector alarm on the 26/11/04 which on inspection of the data was caused due to a change in load. The FFT characteristics appeared to be normal from the manual inspection of the plots so when prompted the user answered that there were no spikes or base noise on the FFT plots. The results derived by the Expert System and completed manually by the expert are summarised in table 4.1.

Table 4.1: Check sheet results for alarm turbine 1, channel 5, 26/11/04

Check Sheet Entry	*Manual*	*Expert System*
OA	13	12.343
Zone	1	1
1X Amp	11	10.445
2X Amp	2	1.5617
1X + 2X ~ OA	Y	Y
Non-Synch Amp	7	5.7021
1X Phase	-37	-31.39
2X Phase	78	74.851
1X Step	N	N
2X Step	N	N
Op Change	Y	Y
Cause	Load Change	Load/Rotor Current/Mvar Change

The module indicated three potential causes of the alarm. These were Load, Rotor Current or MVAr change. The manual inspection had determined that the alarm had been triggered due to a load change. The rationale provided by the system for these 3 diagnoses are given in figure 4.23. Table 4.2 can be used to relate the rule numbers given in figure 4.23 to the rules listed in appendix B.

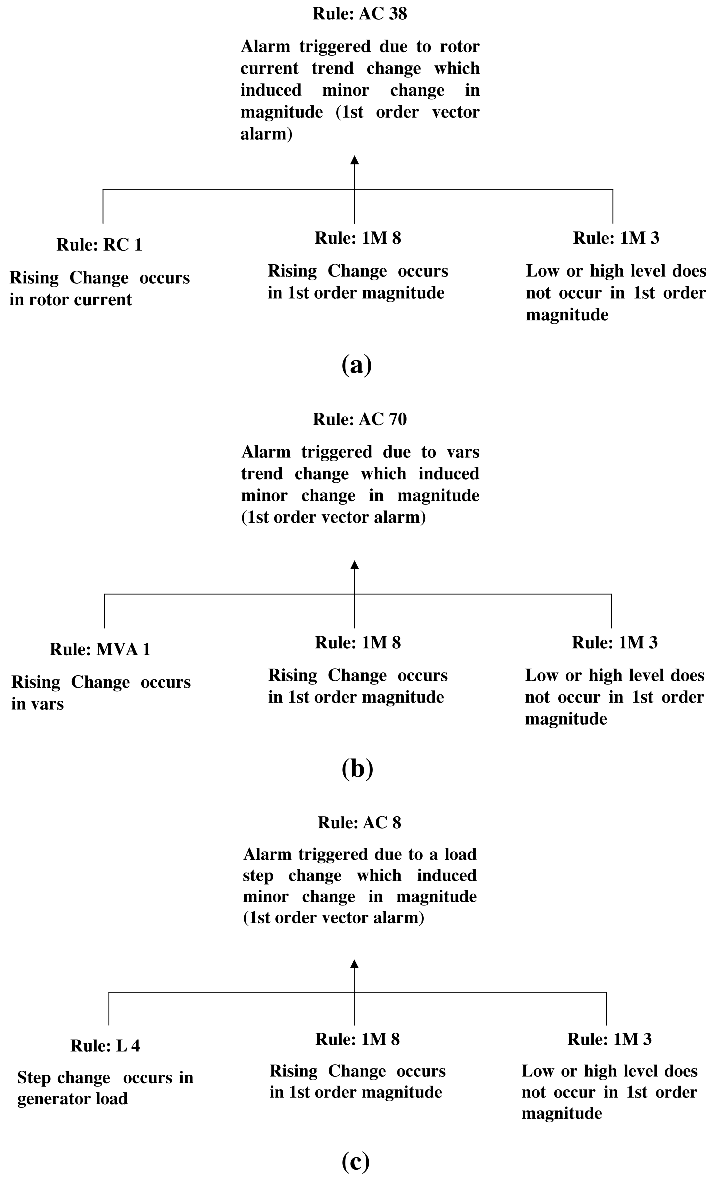


Figure 4.23: Rationale produced by the Expert System for assessment of alarm on turbine 1, channel 5, 26/11/04

It should be clear that all 3 assessments have deduced that a rising change in the 1st order magnitude has triggered the alarm and that this change never fell within the high or low level set out by the channel limits. Each assessment then provides a different explanation as to the cause of this change. Figure 4.23(a) indicates that the change was caused by a rising change in rotor current through the triggering of RC 1. Figure 4.23(b) indicates that the change was caused by a rising change in the MVArs through the triggering of

MVA 1. Finally, figure 4.23(c) indicates that the change was caused by a step change in the load through the triggering of L 4.

Table 4.2: Conversion table to relate the rules used in the rationale in figure 4.23 to those listed in appendix B

Expert System Rule Number	*Appendix Rule Number*
AC 38	AC 6
AC 70	AC 6
AC 8	AC 7
RC 1	RC 1
MVA 1	MVA 1
L 4	L 4
1M 8	1M 8
1M 3	1M 3

The user can then plot the graphs to determine visually what features extracted by the system led to the above assessment being deduced. Figure 4.24 highlights the feature extraction results produced by the system for this alarm assessment. The expert user can easily deduce the features which triggered each alarm in the assessment rationale therefore speeding up the verification process and building user confidence in the system. The trend change in both the MVAr and rotor current signals which triggered rules MVA 1 and RC 1 respectively in the assessment rationale are clearly visible as is the step change in the generator load which resulted in rule L 4 triggering. In practice the user can use this explanation function to determine what the actual cause was, based on the suggestions output by the system. In this example the expert deduced that the original assessment of a load change produced in the original manual assessment and provided as one of the three conclusions by the Expert System was the main cause of the alarm. This is mainly because the change in load in this case initiated the changes in rotor current and MVAr.

This example demonstrates that the Expert System uses the explanation facilities provided and the expert verification to deal with resolving conflict in the system assessments. Since the system assessments are verified by the user as opposed to being used as a definitive conclusion, the need for conflict resolution was not identified as being of primary importance to the system design and could be implemented as part of the evolution of the system. An approach which could be used to differentiate between multiple conclusions would be to add additional knowledge to identify the most likely conclusion. For example, the conflict here could be resolved by additional knowledge which identifies that a change in load would initiate both the change in rotor current and MVArs therefore making load change the primary cause. This approach in addition to alternative approaches for dealing more effectively with the issue of conflict resolution will have to be investigated further for future project developments.

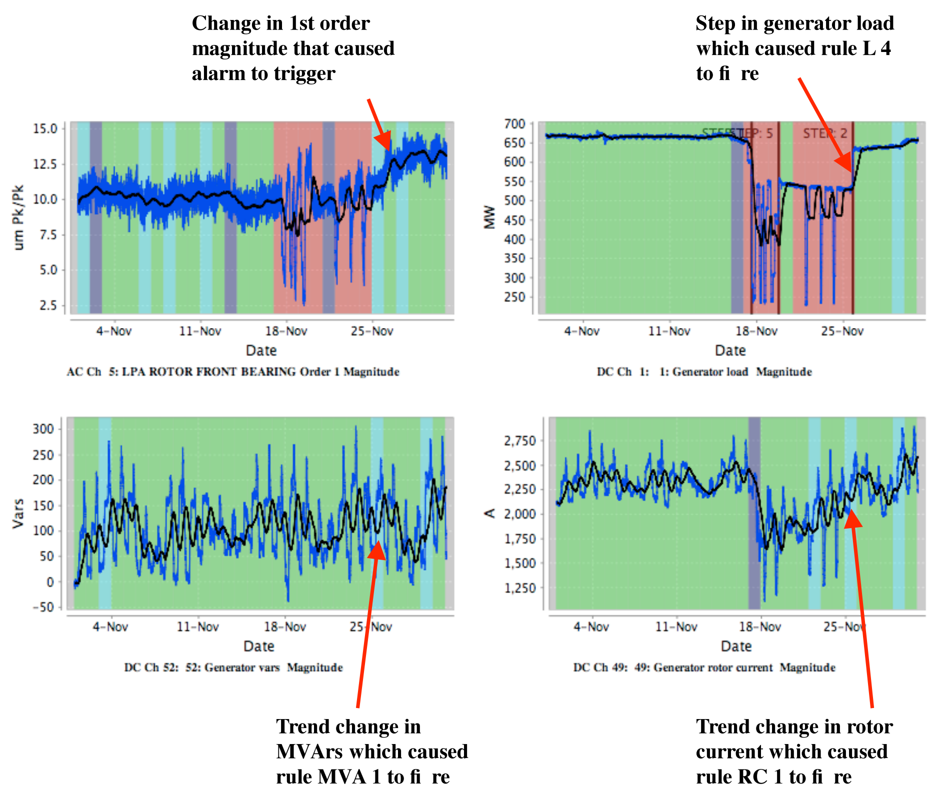


Figure 4.24: Feature extraction results highlight the features which produced the assessment rationale for the Beran alarm triggered on turbine 1, channel 5 on 26/11/04

It should also be apparent from the results given in table 4.1 that the numerical values given by the system do not correlate exactly with those given in the manual assessment. The results were reviewed by the company experts and they felt that the system results correlated well with those in the manual assessment. Therefore, the expert verified that the results provided by the system would be accepted to be put forward for the checksheet report. However, the difference in numerical results was discussed and it was felt that the discrepancy could be explained by the approximate nature of the assessment performed by the expert. In practice the expert will be performing an assessment on several alarms with a limited amount of time. Therefore, the accuracy of the numerical values entered in the checksheet is not seen to be important as long as they are within a certain tolerance. This tolerance varies depending on the magnitude of the signal. For example the results given in table 4.1 show that the 2^{nd} order magnitude was approximated in the manual analysis as being 2μm peak to peak (p-p) whereas the Expert System which is using the verified knowledge calculates it to be 1.5617μm p-p. In percentage terms this is a significant difference but since the vibration magnitude would have to be greater than 63μm p-p to fall outside of zone 1 the difference of 0.5μm p-p appears insignificant. Alternatively if the manual analysis estimated a vibration magnitude as being at 80μm p-p and the system calculated it to be 60μm p-p then this would be a significant difference since the first value is in 'Zone 2' and the second is in 'Zone 1'. The expert also highlighted that there is error in the expert analysis when visually interpreting time series plots on a screen since the resolution of the plot may not permit accurate readings to be easily obtained. It was concluded that the analysis performed by the system was producing more accurate numerical readings compared with the manual analysis due to the aforementioned reasons. Therefore, the Expert System showed its ability to increase the accuracy and the consistency of the numerical checksheet analysis by eliminating the human error introduced in the manual analysis.

The next alarm tested on channel 5 on turbine 1 was again triggered as a '1X Vector' alarm on the 08/02/06. Again the FFT data showed no abnormal signs of behaviour therefore it was manually input to the system that no spike or base noise was present in the FFT plots. The Expert System results and manual results are given in table 4.3.

The non-synchronous amplitude is higher for the module results than that recorded in the manual analysis and there is no frequency given since the module doesn't have access to the FFT data to determine the particular frequency that this magnitude dominates. In addition there are discrepancies in the numerical values given for the 1st and 2nd order phases. On inspection of the results with company experts it was concluded that the system was producing the more accurate results due to the approximate nature of the manual analysis as explained for the previous results. It is however important that methods of automatically calculating the dominant sub-synchronous frequency are researched once the FFT data becomes available so that this information can be input to the final checksheet report.

Table 4.3: Check sheet results for alarm turbine 1, channel 5, 08/02/06

Check Sheet Entry	*Manual*	*Expert System*
OA	15.6	13.004
Zone	1	1
1X Amp	14.6	11.240
2X Amp	1.3	1.4290
1X + 2X ~ OA	Y	Y
Non-Synch Amp	4.1/16.4	5.4821
1X Phase	-69	-61.24
2X Phase	72	63.212
1X Step	N	N
2X Step	N	N
Op Change	Y	Y
Cause	Load Increase	Load/Rotor Current Change

There are further discrepancies in the overall amplitude and first order magnitude values. An inspection of the results found that the manual analysis had outperformed the system in this instance. It can be seen from figure 4.25 that the manual analysis has noted the peak value close to the alarm firing in both the overall amplitude and 1st order magnitude plots. The system on the other hand has recorded an average for both signals. The knowledge within the system has been designed to record maximum values close to the time of alarm firing only in instances where the magnitude exceeds a maximum level defined for that channel. In this instance the magnitude did not exceed this value, therefore the average was only taken. The expert who was verifying the results did express that in instances where there has been upwards trending close to the firing of the alarm, it is best practice to record the peak magnitude within that vicinity.

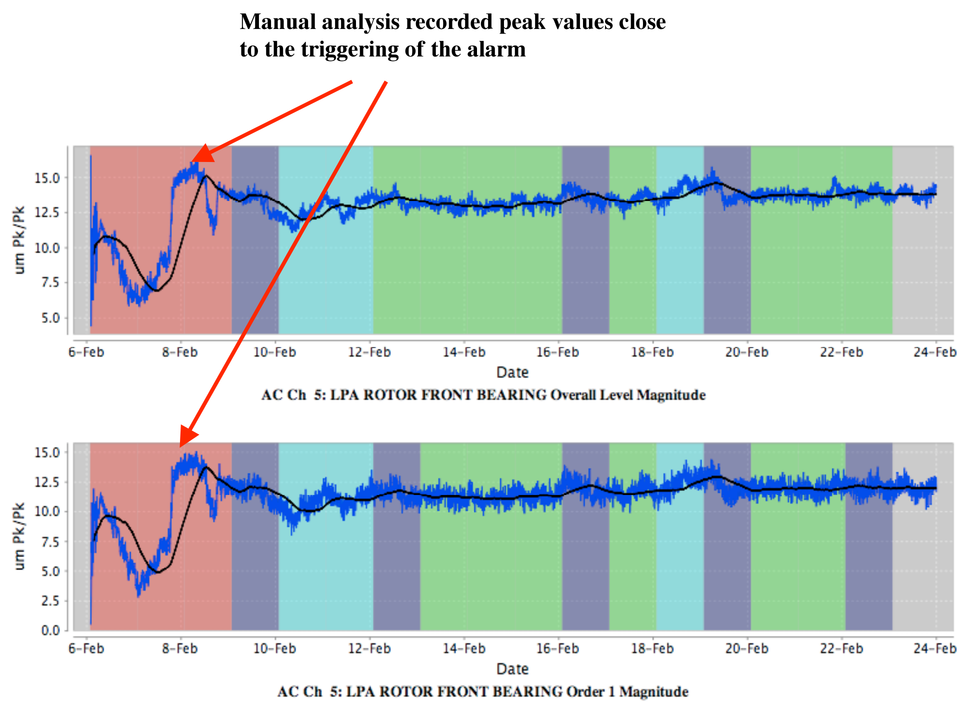


Figure 4.25: Overall level magnitude and 1st order magnitude plots highlight where the expert has read the peak magnitude values and input both values to the checksheet.

It is important that this feedback at the system result verification stage is used to update the existing knowledge base so that such functionality is included in later versions of the Expert System. This highlights an important advantage of the Expert System approach, where due to the transparency of the rationale provided by the system results, the system developer and experts are able to interrogate the results and determine how they can be improved upon. This has the effect of both improving the analysis performed by the system and at the same time building user confidence in the system. This was one of the primary reasons for selecting the Expert System approach as outlined in section 4.2.

The module does not specify that an increase in the load triggers the alarm but does indicate that either a load or rotor current change was the cause. The rationale produced by the system for both of these assessments is given in figure 4.26.

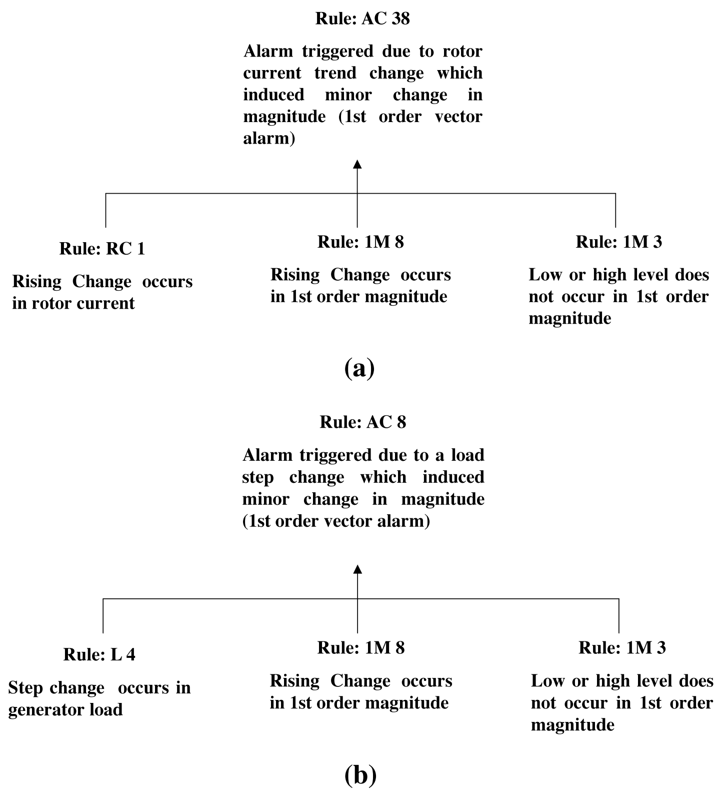


Figure 4.26: Rationale produced by the Expert System for assessment of alarm on turbine 1, channel 5, 08/02/06

The rules given in figure 4.26 can be matched to those described in appendix B using table 4.2. Both rationales indicate that the alarm triggered on an increase in the 1^{st} order magnitude which fell within the normal magnitude limits stipulated by the channel limits. Figure 4.26(a) indicates that a trend change in the rotor current was the cause for the change through the triggering of rule RC 1. Figure 4.26(b) indicates that a step change in the generator load was the cause through the triggering of rule L 4. The expert user can verify the presence of these features in both the rotor current and generator load through an analysis of the feature extraction results as shown in figure 4.27.

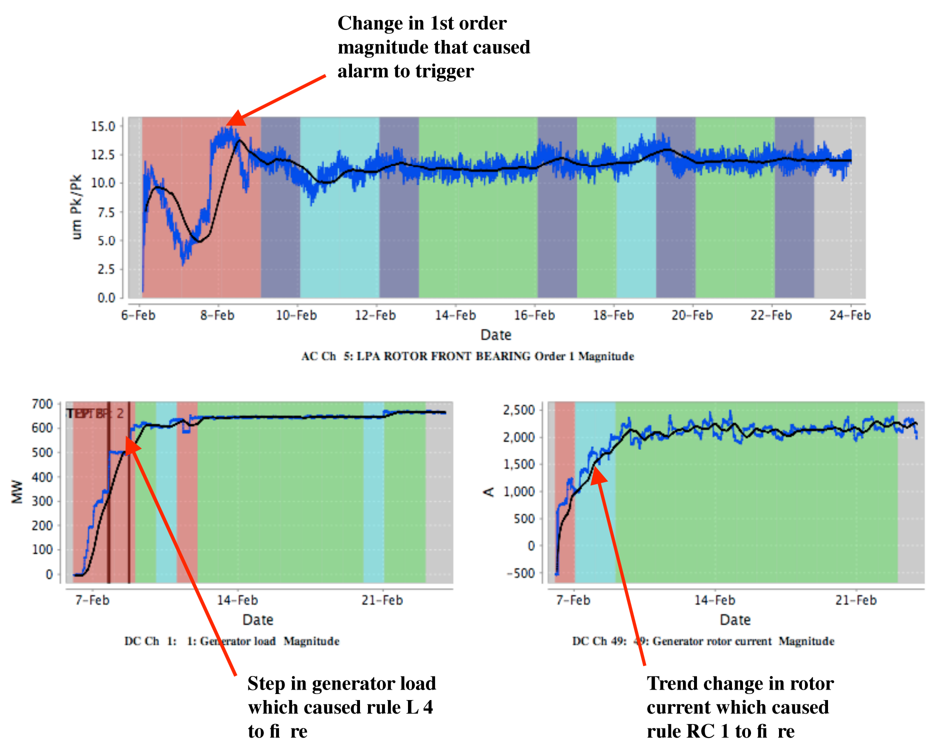


Figure 4.27: Feature extraction results highlight the features which produced the assessment rationale for the Beran alarm triggered on turbine 1, channel 5 on 08/02/06

The trend change in the rotor current signal which triggered rule RC 1 in the assessment rationale is clearly visible as is the step change in the generator load which resulted in rule L 4 triggering. The expert was able to quickly verify that the alarm was triggered due to a load increase through the use of the rationale and the visually highlighted feature extraction results. The reason that the change in load was deemed to be he primary cause was for the same reason as the previous example, that is the change in load would have been the cause of the change in rotor current. This example again shows how the Expert System does not implement automated conflict resolution but uses the user and the system explanation to resolve the conflict at the assessment confirmation stage. Again a future development of the system could be to investigate an implement approaches to conflict resolution to provide the expert with less assessments to verify.

This example demonstrates that the Expert System was able to identify the cause of the alarm and effectively feed this information back to the user through the system's explanation facilities. It also demonstrated the strength of the Expert System approach in being able to identify areas where the results were not as expected and use this to update the knowledge to improve the system performance for further alarms. This was demonstrated for the overall amplitude and 1st order magnitude cases where the system did not identify the correct location in the graph to derive the associated checksheet entries, the transparent nature of the knowledge allowed the expert and system developer to determine why there was a discrepancy and feed this information back so that later versions could be updated with improved knowledge.

4.6.1.2 Channel 6

The next alarm tested on the module was taken from channel 6 on turbine 1 and was triggered on 31/10/04. The alarm was triggered as a 1X Vector alarm and there were no signs of abnormal behaviour in the FFT. Therefore it was entered manually into the system that no spikes or base noise appeared within the FFT plot. The results for this test are given in table 4.4.

Table 4.4: Check sheet results for alarm turbine 1, channel 6, 31/10/04

Check Sheet Entry	*Manual*	*Expert System*
OA	12	11.306
Zone	1	1
1X Amp	9	8.1370
2X Amp	6	5.2712
1X + 2X ~ OA	Y	Y
Non-Synch Amp	6	5.4195
1X Phase	80	77.104
2X Phase	~180	-5.348
1X Step	N	N
2X Step	N	N
Op Change	N	N
Cause	Drift	Drift

The results for the module analysis and the manual analysis compare well with the exception of the second order phase. The reason for the discrepancy is that the phase value is situated on the 180° mark which means that the signal is alternating between approximately the 180° mark and the -180° mark. An average of the signal is being taken since the signal has been described as stable by the 'Extract Channel Profile' module and so this equates to approximately 0°. This problem is discussed in section 4.5.4.4 as is a solution proposed which if developed further could correctly calculate the differences in the phase signals. The review of the differences between the manual and system derived checksheet entries concluded that it was again due to the approximations made by the expert in the manual analysis. Therefore the Expert System was again producing more accurate results than the manual analysis. The alarm cause was assessed to be 'Drift' in both the manual and automated analysis. The rationale produced by the system for the 'Drift' assessment is given in figure 4.28.

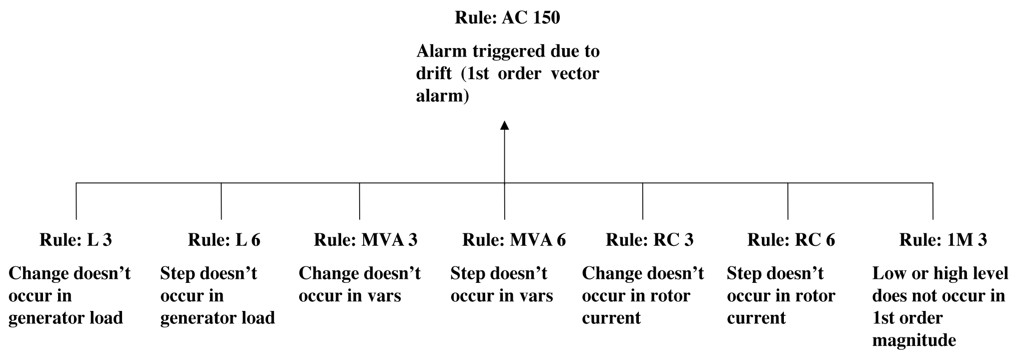


Figure 4.28: Rationale produced by the Expert System for assessment of alarm on turbine 1, channel 6, 31/10/04

Table 4.5 can be used to match the rules given in the explanation in figure 4.28 to those given in appendix B.

It can be seen from the rationale produced by the system that the alarm was assessed as drift because no change was located in any of the operational parameters. This is verified by the triggering of rules L 3, L6, MVA 3, MVA 6, RC 3 and RC 6. In addition the level of the 1^{st} order magnitude remained within normal limits, which is verified by the firing of rule 1M 3. This indicated that the change was 'Drift' as opposed to a 'Genuine Unattributed Change'. These results can be quickly verified through analysis of the feature extraction results given figure 4.29.

Table 4.5: Conversion table to relate the rules used in the rationale in figure 4.28 to those listed in appendix B

Expert System Rule Number	*Appendix Rule Number*
AC 150	AC 13
L 3	L 3
L 6	L 6
RC 3	RC 3
RC 6	RC 6
MVA 3	MVA 3
MVA 6	MVA 6
1M 3	1M 3

It can be seen from figure 4.29 that the load signal remains stable throughout the period of alarm firing. There is some movement in the MVAr and rotor current signals but this occurs after the alarm has triggered therefore implying that neither of these changes were the cause. This example again demonstrates the accuracy of the Expert System assessment and how the user is able to easily validate the assessment using the explanation functions provided by the tool.

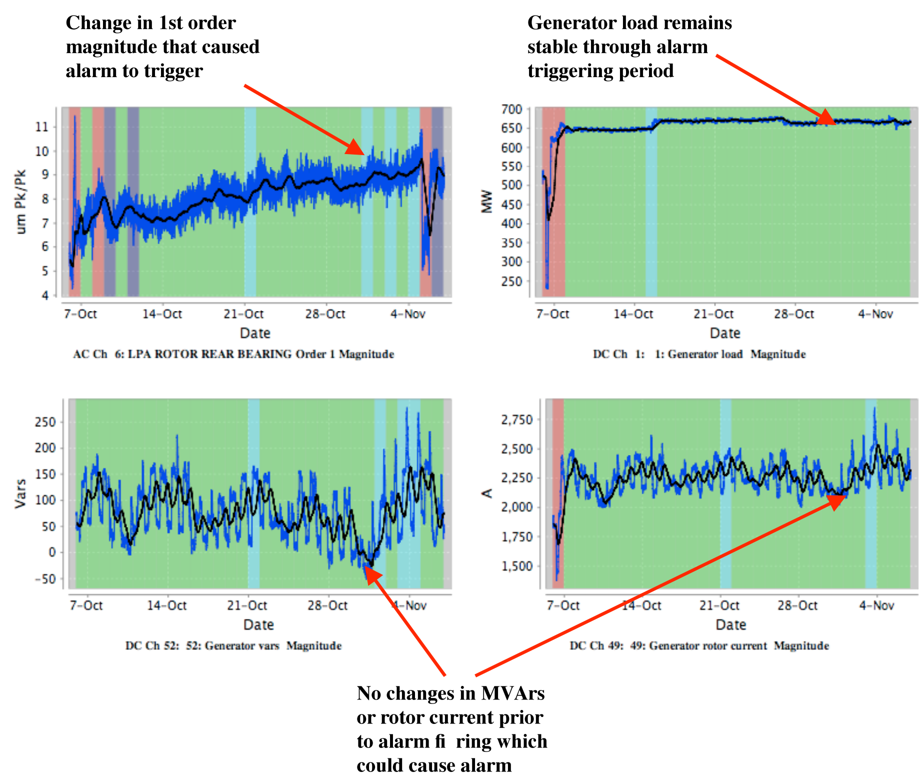


Figure 4.29: Feature extraction results highlight the features which produced the assessment rationale for the Beran alarm triggered on turbine 1, channel 6 on 31/10/04

4.6.1.3 Channel 9

The next alarm tested on the module was from channel 9 on turbine 1. This alarm occurred on 03/12/04 and was triggered by a ‘1X Vector’ alarm. A manual inspection of the FFT data showed that there were no abnormal characteristics in base noise or frequency spikes. The test results are given in table 4.6.

Many of the entries calculated by the Expert System compare almost exactly with the values derived manually by the expert. There are however some discrepancies between both sets of results. The numerical entries match very closely between both assessments although it was concluded that all the differences between both assessments were due to the approximations incurred during the manual assessment, with the Expert System producing more accurate results. A good example which demonstrates this is the non-synchronous amplitude entry. The time series data can be seen in figure 4.30 where the average of the signal has been plotted on top of the raw signal. It can be clearly seen that the average falls closer to 4.6μm p-p as opposed to the 6 estimated in the manual assessment.

Table 4.6: Check sheet results for alarm turbine 1, channel 9, 03/12/04

Check Sheet Entry	*Manual*	*Expert System*
OA	40	39.320
Zone	1	1
1X Amp	25	24.746
2X Amp	31	29.919
1X + 2X ~ OA	N	Y
Non-Synch Amp	6	4.6248
1X Phase	20	23.611
2X Phase	-135	-135.0
1X Step	N	N
2X Step	N	N
Op Change	N	Y
Cause	Low Rotor Current (Potential Mvar Change)	MVAR Change

Note: This entry is answered Y if 1X + 2X = ≤1.5xOA and ≥0.8xOA

The expert who completed the checksheet determined that the addition of the first and second order magnitudes did not approximately equate to the value of the overall amplitude whereas the Expert System's diagnostic knowledge calculated that they were approximately equal. The expert who compared both sets of results felt that the system conclusion was correct. The best practice agreed upon at the knowledge elicitation stage for this particular chaecksheet entry was that if the addition of the 1st and second order magnitudes fell within +50% and –20% of the overall amplitude value then both values were deemed to be approximately equal to each other. The results from this analysis fulfil this condition therefore the Expert System correctly determined that both values were approximately equal to each other. The expert was unable to conclusively state why the manual analysis was completed incorrectly and could only speculate that the best practice had not been adequately cascaded to the rest of the team. Therefore the Expert System results combined with the best practice derived during the knowledge engineering process provided the expert with a learning point which could be fed back to the Rotating Plant and Dynamics Team.

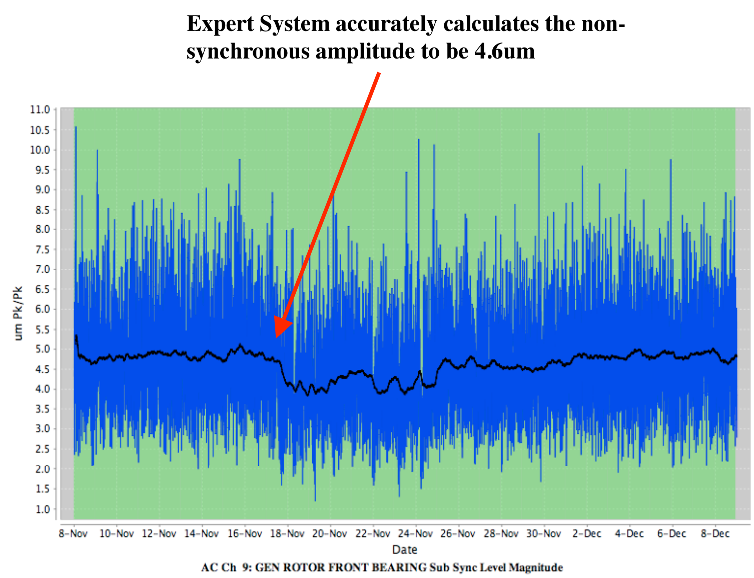


Figure 4.30: The average value plotted on the sub-synchronous magnitude plots highlights that the Expert System correctly approximated the average to approximately 4.6 as opposed to 6 which was entered in the manual analysis.

Another discrepancy was that the Expert System picked up a change in the MVAr operational parameter sufficiently close to the time of alarm firing to record that an operational change occurred. The manual results show that the expert performing this particular assessment concluded that no change had occurred in any of the operational parameters. Additionally the manual assessment concluded that the alarm had been caused by either a low rotor current or a potential change in MVArs which contradicts the assessment that there was no operational change. Again the experts who compared both sets of results felt that the Expert System had performed more accurately in picking up the change in the MVAR signal. The rationale produced for the MVAr change alarm assessment is given in figure 4.31.

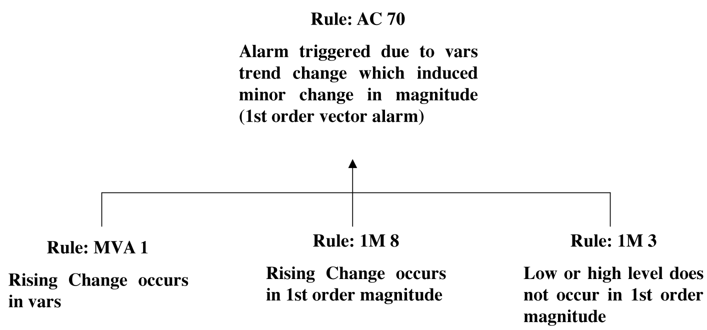


Figure 4.31: Rationale produced by the Expert System for assessment of alarm on turbine 1, channel 9, 03/12/04

The rules given in the explanation in figure 4.31 can be related back to the rules listed in appendix B using table 4.7.

Table 4.7: Conversion table to relate the rules used in the rationale in figure 4.31 to those listed in appendix B

Expert System Rule Number	*Appendix Rule Number*
AC 70	AC 6
1M 3	1M 3
1M 8	1M 8
MVA 1	MVA 1

The rationale indicates that a trend change in the MVAr signal highlighted by the triggering of rule MVA 1 caused the change in the first order magnitude as indicated by the triggering of rule AC 70. This rationale can be verified by analysing the feature extraction given in figure 4.32. The graphs show that there is a change in the MVArs which occurs on the 1st December, approximately 2 days before the alarm. This falls within the time window, elicited at the knowledge engineering stage, in which a change in an operational parameter is able to effect a change in a vibration parameter. The system concludes that it is this change which causes the change in the 1st order magnitude. This assessment was verified by the expert. The rotor current signal is also plotted in figure 4.32 to highlight how the change in this parameter preceding the time of alarm triggering occurred on the 29th November, approximately 4 days before the alarm. This falls outside the period set at the knowledge elicitation stage in which an operational change can induce a change in vibration parameter. Therefore the Expert System correctly assessed that this was not a contributing factor to the change in the 1st order vibration. In addition it can also be deduced from the plot that there is no abnormal drop in rotor current therefore contradicting the manual analysis listed in table 4.6. The expert verifying the results was unable to conclusively determine why the original manual analysis contained these inconsistencies.

This case study demonstrates that the assessment produced by the Expert System has located features which were not found in the manual analysis and highlighted incorrect assessments and contradictions in the original manual analysis. It also demonstrated how the assessment rationale and the visual aids provided by the explanation functions allowed the expert to review and confirm the Expert System assessment and discount elements of the manual analysis. This is a powerful tool in ensuring that all possibilities of alarm cause are identified therefore ensuring that the most accurate assessments are produced.

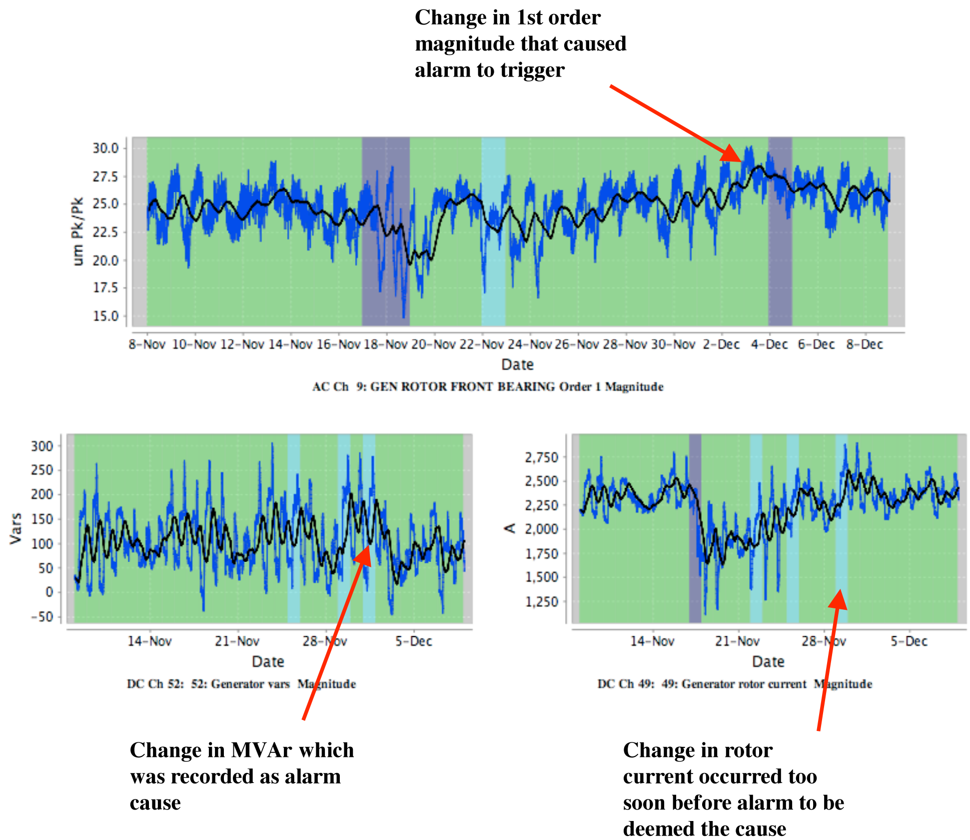


Figure 4.32: Feature extraction results highlight the features which produced the assessment rationale for the Beran alarm triggered on turbine 1, channel 9 on 03/12/04

4.6.1.4 Channel 11

The next alarm used to test the module was taken from channel 11 on turbine 1 and occurred on 19/12/04. This alarm was triggered as a signal low alarm. No abnormal behaviour was observed in the manual inspection of the FFT. The test results are given in table 4.8. Note that there are bracketed values in table 4.8. This denotes that a maximum and minimum value for that signal was recorded. This is implemented when either a large or low value is recorded in the signal following the alarm, but only in instances where the signal has been relatively stable before the alarm. For example, in table 4.8 the expert system has recorded a maximum overall amplitude of 14.278μm p-p and a minimum of 1.0438μm p-p since the signal was recorded as being relatively stable and then recording a low value following the alarm.

The review of the results indicated that the Expert System had recorded all of the numerical entries more accurately than the manual analysis with the exception of the 2^{nd} order magnitude. The increase in accuracy for most of the numerical entries could be explained by the approximate nature of the manual analysis as discussed for the previous alarms. The expert system only recorded a single value for the 2^{nd} order phase entry because it did not record an abnormally high or low level following the alarm. The expert explained that it was good practice to record a maximum and minimum in all of the vibration signals if one is recorded for the overall amplitude, 1^{st} order magnitude and second order magnitude signals. This knowledge had not been recorded at the knowledge elicitation stage because the scenario had never arisen in the worked examples in the earlier knowledge elicitation sessions, neither had the possibility been explored at the later more structured meetings as discussed in section 4.3. This knowledge will be used to update the knowledge base on the next version of the system.

Table 4.8: Check sheet results for alarm turbine 1, channel 11, 19/12/04

Check Sheet Entry	*Manual*	*Expert System*
OA	13-1	14.278-(1.0438)
Zone	1	1
1X Amp	11-0.8	12.4-(0.4259)
2X Amp	3.4-0.6	3.9295-(0.6306)
1X + 2X ~ OA	Y	Y
Non-Synch Amp	5-1	5.7467-(1.0147)
1X Phase	153-(-120)	150.65-(-98.30)
2X Phase	38-29	26.827
1X Step	Y	Y
2X Step	Y	Y
Op Change	N	N
Cause	Signal Fault	Zero Sensor Reading

The Expert System outperformed the manual analysis by giving more detail as to the cause of the alarm. The system determined the alarm to be a ‘Zero Sensor Error’ which is a type of signal fault. This was diagnosed through the triggering of rule AC 3: Alarm triggered due to zero sensor reading which relates to rule AC 3 in appendix B. This rule effectively analyses the overall signal to determine if a certain number of samples fall below a threshold point within a specified period. Figure 4.33 highlights where these low value readings were read by the system. This example again demonstrates that the Expert System is able to provide accurate results which are able to be quickly and easily verified by the expert user. It also highlighted how in instances where the system output does not exactly match that expected by the user, the cause of the discrepancy can be located and this learning fed back into later versions to improve the performance further.

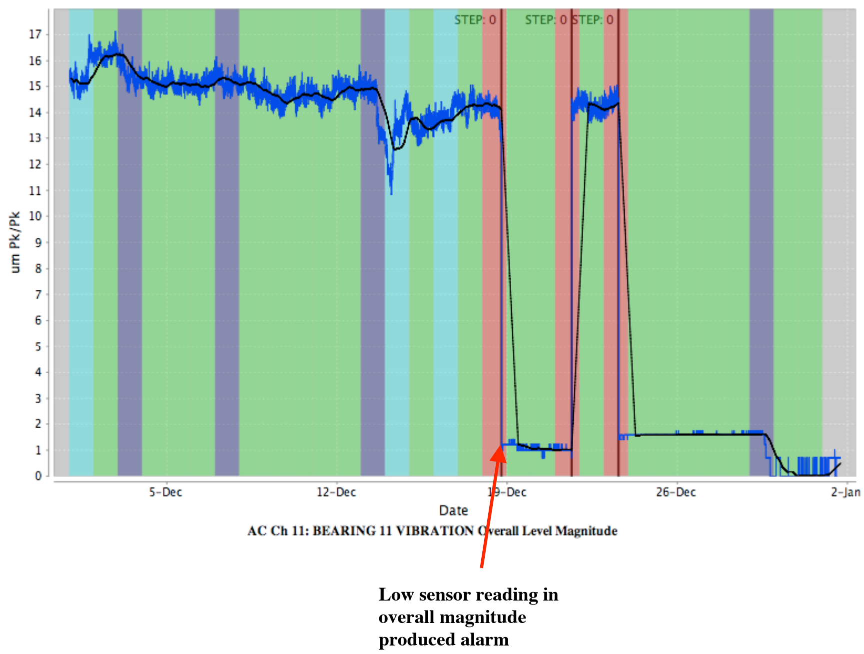


Figure 4.33: Low level reading which produced the Beran alarm on turbine 1, channel 11 on 19/12/04

The next alarm tested was also taken from channel 11 on turbine 1 and occurred on 08/07/06. This alarm was triggered by a signal low alarm and there was no abnormal behaviour observed in the FFT plot at the manual inspection stage. The results of this test are given in Table 4.9.

On review of the numerical entries it was confirmed that the Expert System produced more accurate results than the manual analysis due to the error introduced by the expert inspection of the signals as explained for the previous alarms. There is a large discrepancy in the calculation of the 1st order phase. The verification stage discovered that the user had incorrectly noted the minimum value where there was a severe noise spike. The best practice taken from the knowledge elicitation exercises had determined that instances of severe noise should not be used for the derivation of the checksheet values. Therefore the Expert System had correctly excluded the noise spike to determine the 1st order phase entry.

Table 4.9: Check sheet results for alarm turbine 1, channel 11, 08/07/06

Check Sheet Entry	*Manual*	*Expert System*
OA	0.71	1.1997
Zone	1	1
1X Amp	0.56	0.9598
2X Amp	0.4	0.3621
1X + 2X ~ OA	Y	Y
Non-Synch Amp	0.9	0.8722
1X Phase	-180	-65.15
2X Phase	-15	-16.63
1X Step	N	Y
2X Step	N	Y
Op Change	N	Y
Cause	Low Signal	Zero Sensor Reading

Other differences between the manual and system assessment were in detecting steps in the first and second order magnitude and in also detecting an operational change. Steps do occur in the first and second order phase signals as well as changes in the operational signals as can be seen in figure 4.34. One explanation for these discrepancies would be that the expert never defined these as changes in the checksheet because the steps in the vibration signals were caused by the signal regaining the reading after having a signal error and because the operational changes never contributed directly to the alarm. However the best practice derived for the Expert System indicated that the changes in both the operational and vibration signals should always be noted regardless of the cause. The expert reviewing the results also agreed with this reasoning and confirmed that the Expert System analysis was correct. The alarm cause was diagnosed again by the AC 3 rule as was the case for the previous alarm on channel 11 reviewed in this section.

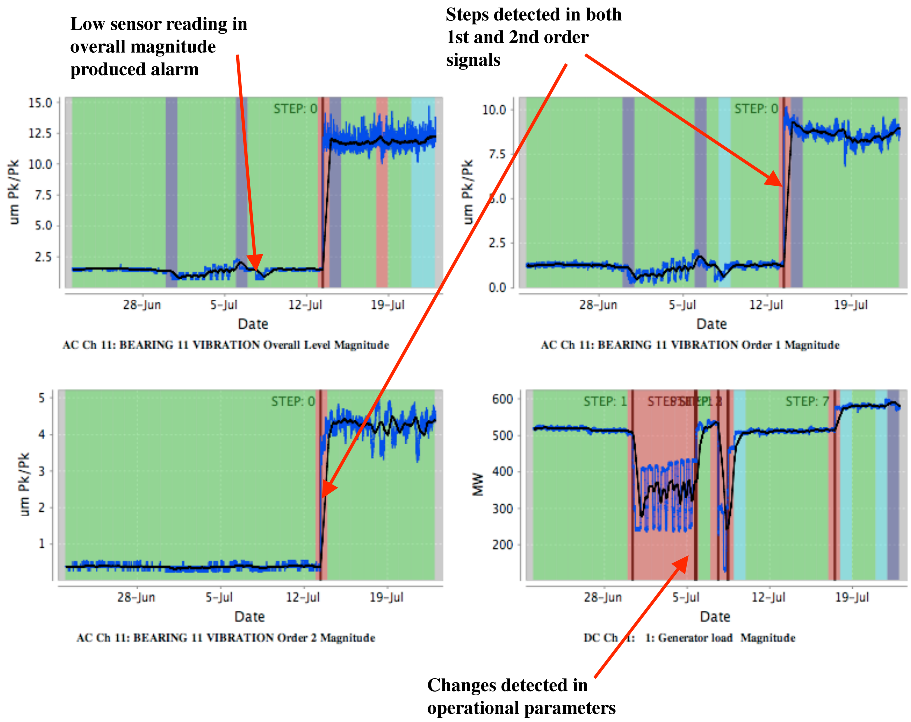


Figure 4.34: Feature extraction results highlight the features which produced the assessment rationale for the Beran alarm triggered on turbine 1, channel 11 on 08/07/06

The results for this alarm again demonstrate the benefits of the Expert System in its ability to accurately and consistently assess the data. This is highlighted in the Expert System providing more accurate readings and highlighting areas where the manual analysis had noted incorrect conclusions when compared with the best practice captured at the knowledge engineering stage.

4.6.2 Turbine 2

The system was also tested on a single alarm from turbine 2. This alarm was triggered on channel nine and is reviewed in the following section.

4.6.2.1 Channel 9

The final alarm used for testing the module was from channel 9 on turbine 2 on 08/01/06. This alarm was triggered as a '1X Vector' alarm and the manual inspection of the FFT data showed no abnormal behaviour which had to be entered into the system. The results for this test are given in table 4.10.

Table 4.10: Check sheet results for alarm turbine 2, channel 9, 08/01/06

Check Sheet Entry	*Manual*	*Expert System*
OA	62	50.435
Zone	1	1
1X Amp	57.1	51.924
2X Amp	26.3	25.026
1X + 2X ~ OA	N	N
Non-Synch Amp	6.2/18	5.3931
1X Phase	145	145.91
2X Phase	29.7	36.292
1X Step	N	Y
2X Step	N	Y
Op Change	Y	Y
Cause	Load Change	Load/Rotor Current Change

It can be seen that there is a discrepancy in the overall amplitude and 1st order magnitude value calculated. The expert has noted the peak value which occurred before the alarm was fired, whereas the average value after the alarm has been noted by the module as shown in figure 4.35. The expert reviewing the results agreed with the values noted in the manual analysis. The reason for this discrepancy is the same for that explained for the alarm which triggered on channel 5 on turbine 1 on 08/02/06 and reviewed in section 4.6.1.1. The reason being that it is best practice to record the maximum value where there has been oscillation close to the alarm firing. This had not been brought out at the knowledge engineering process and will therefore be fed back to update the knowledge base for later implementations of the system. The expert agreed that the system had recorded all of the other numerical entries more accurately than the manual analysis. No dominant frequency value has been noted for the non-synchronous entry since the module has no access to the FFT data. The module uncovered step changes in the first and second order magnitude as shown in figure 4.35, whereas the expert had not. The expert agreed that steps changes had occurred in the signals and should have been noted in the checksheet. There was no clear explanation as to why these features had been omitted in the original manual analysis.

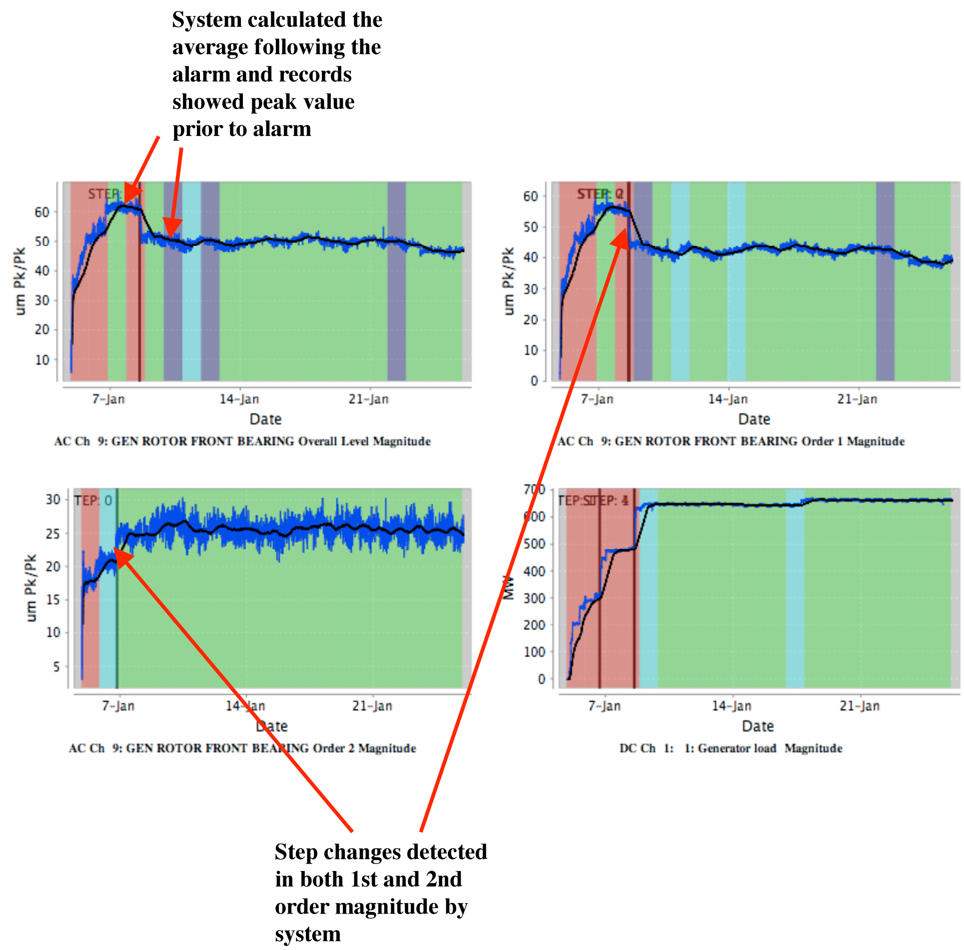


Figure 4.35: Discrepancies between manual and system results in values calculated and events detected for Beran alarm on turbine 2, channel 9 on 08/01/06

The rationale for the assessment results are given in figure 4.36. The system detected that the 1st order magnitude changed but remained within normal operating limits. A step change in the load or a trend change in the rotor current were judged to have been the causes. The rules given in the rationale can be matched back to those given in appendix B using table 4.2. The features extracted by the system which produced the rationale in figure 4.36 are shown in figure 4.37.

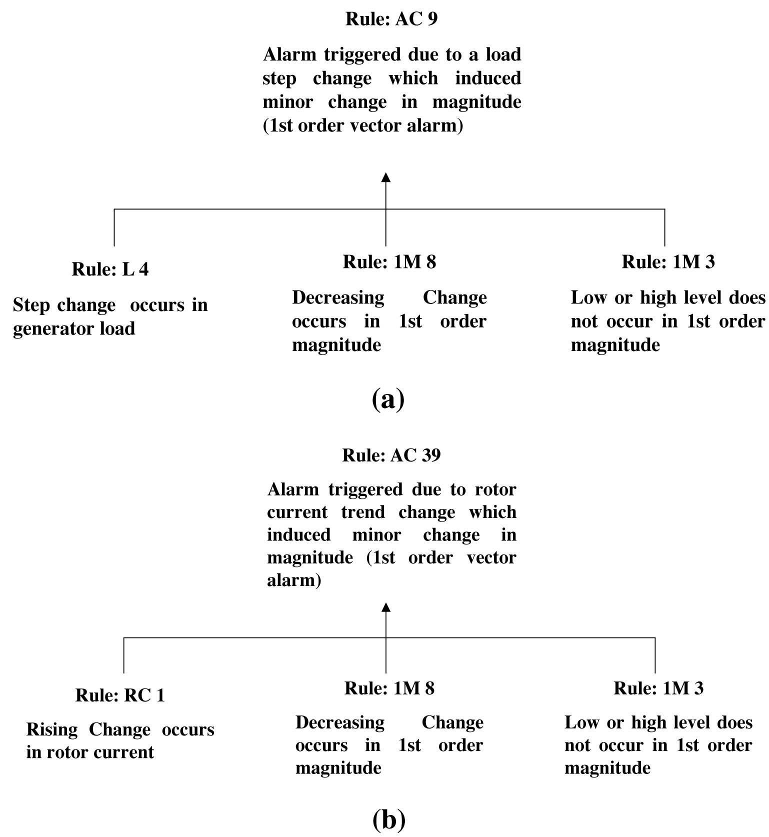


Figure 4.36: Rationale produced by the Expert System for assessment of alarm on turbine 2, channel 9, 08/01/06

It can be clearly seen from figure 4.37 that the Expert System has correctly located the operational changes and related then back to the change in 1st order magnitude. The expert agreed with this assessment but explained that the load change would be the primary contributing factor. This is similar to the alarm triggered on channel 5 on turbine 1 on 26/11/04 in section 4.6.1.1 where multiple causes of the alarm were identified. The explanation functions allow the expert to easily identify and verify the primary cause but as stated for the previous alarms, approaches to conflict resolution should be researched for future implementations of the system.

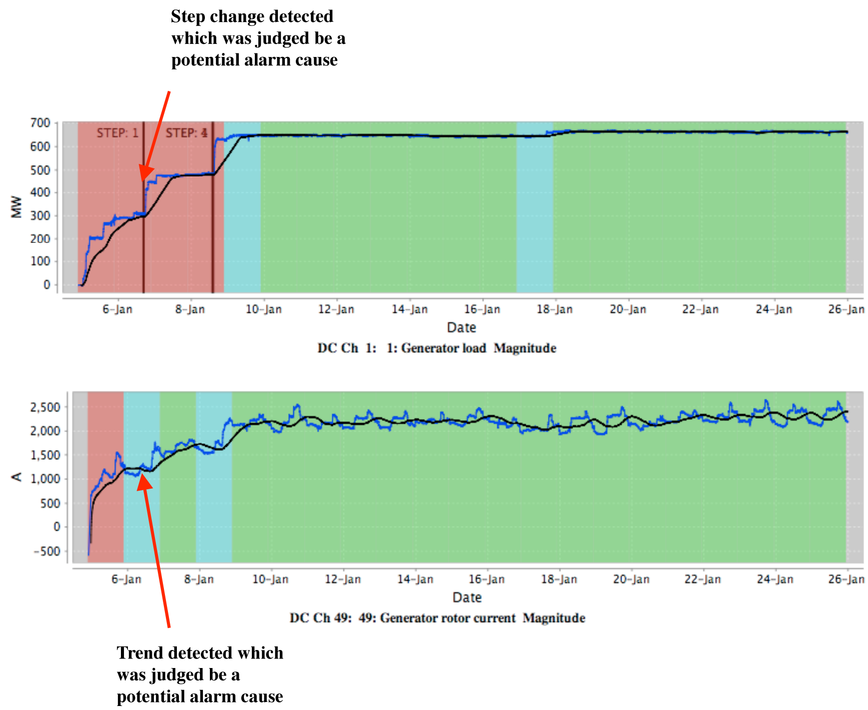


Figure 4.37: Feature extraction results highlight the features which produced the assessment rationale for the Beran alarm triggered on turbine 2, channel 9 on 08/01/06

Overall this example again demonstrates many of the benefits of the Expert System. The first is the accuracy of the numerical values calculated by the system. The second advantage is the transparency of the reasoning approach which allowed the expert to determine why there were discrepancies in the overall amplitude and 1st order magnitude values. This allows the system performance to be continually monitored and improved upon. This example also highlighted how the explanation facilities assist the expert in resolving conflict in the assessment conclusions and how a future development would be to introduce automated conflict resolution to the system using additional knowledge.

4.7 Evaluation

The results obtained from testing on real alarm case studies chosen from the British Energy archive given in section 4.6 demonstrate that the Expert System prototype designed and developed for British Energy is well suited to the turbine generator condition monitoring task. The system in all cases produced assessments of the alarms which the experts were able to verify as being the causes of the alarm. Most of the numerical entries derived by the Expert System were deemed to be more accurate than the manual analysis due to the error introduced from estimating values from a time series plot. The high performance of the system can be explained by the following important design factors:

- Comprehensive expert knowledge on alarm assessment
- Transparent rationale
- Visually aided feature extraction

One of the primary reasons for selecting the Expert System approach for this particular application was that expert knowledge on how to perform alarm assessment was readily available from the British Energy condition monitoring experts themselves. The system developers were able to extract and document this knowledge through the use of the formal knowledge engineering approach detailed in section 4.3. The performance of the Expert System in deriving the required data for the checksheet and the alarm assessments for the case studies demonstrates that the initial assumption that the turbine generator condition monitoring task could be performed using a rule based approach was proved to be correct. However, the case studies did highlight some areas where some improvement is required. The testing highlighted some instances where the system is not selecting the correct point on the time series data to derive some of the numerical entries. The transparent nature of the Expert System knowledge however provides a method of interrogating the results and the associated rationale to determine where the knowledge needs to be updated to provide the correct results. The initial knowledge engineering process attempts to mitigate this problem by engaging as many experts as possible and exploring a multitude of scenarios. The results also highlighted a common problem associated with rule based approaches known as conflict resolution. This effect is particularly problematic in cases where the Expert System is being used to give a definitive answer on some problem to users who are not experts in that particular area. In these instances, methods of resolving the conflict would have to be sought to identify which out of all the assessments is most probable. This system however is intended to act as an assistant to the expert and so multiple conclusions can be easily interrogated by the user to determine the actual cause. A future development for later versions of the system will be to explore methods of conflict resolution so that it is performed automatically. In particular I was found that the conflicts which arose during system testing could be rectified by adding additional rules to the Expert System knowledge base.

Much emphasis was placed on making the assessment rationales meaningful to the user so that he/she could quickly understand each step taken in the analysis. This was achieved primarily by providing each rule with a description explaining the information which could be taken from that particular rule having fired. The result of this approach is an explanation tree which clearly feeds back to the user what events have been located in the data and how these have resulted in the assessment conclusion. The rationale proved crucial in gaining the user confidence in the system since on occasions where there was conflict in the assessment conclusions or debate over the accuracy of the assessment, the user was able to easily check the rationale and then reference the appropriate signal to determine if the conclusions derived were in fact accurate. This explanation functionality can be enhanced in future versions of the system by generating the explanation tree structure automatically via the system user interface.

The last factor which enabled the prototype system to perform well in this particular application was the novel use of visual aids to allow the expert to quickly determine which features had been located by the system and subsequently used in the analysis.

This tool used in conjunction with the assessment rationale allowed the user to clearly deduce which events were located and subsequently used by the system to derive the conclusion. In addition, the events derived by the feature extraction module were shown to match closely to that of the expert's interpretation of the data.

The case studies demonstrate that the novel approach to combining these two forms of explanation, both rationale and features, greatly enhance the application of such rule based approaches in this type of application. The rationale forms the basis of the validation process by indicating the type of features or events which were located by the signal to symbol transformation and what conclusions were then drawn from this. The user can then reference the necessary signals and determine for themselves if the features highlighted are correctly identified and if the conclusions drawn are in fact correct. This approach provides a means of enhancing the symbolic information derived and utilised by Expert Systems which humans are able to identify and interpret. The next step to enhancing this functionality would be to improve the interface so that the linkage between both the rationale and the features extracted become more easily distinguished. At present none of the features identified in the data plots are linked back to the explanation. This could be improved upon by simply labelling each of the features and referencing them in any of the explanations which they appear in or by automatically directing the user to a feature used in the triggering of a certain rule when clicking on the rule within the explanation tree.

In addition to acting as an assistant for the expert user and therefore providing potential for the process of alarm assessment to become more efficient, the Expert System also has the potential to make the assessments more accurate than a purely manual approach. This is demonstrated in the case study results by the prototype picking up on inconsistencies in the original assessments. Using the Expert System to provide an initial analysis of the data will in some cases provide the user with assessment explanations which may have been overlooked or missed in some cases of manual assessment. Another advantage is the ability to assess the alarms prior to any manual inspection of the data. This provides a natural ordering of priority for the user as he/she logs onto the system. The pre-processing undertaken enables the experts to prioritise what alarms are assessed first and increases the chances of locating genuine problems in the turbine at an earlier stage than would be possible with no pre-processing. This is especially true in situations where vast amounts of alarms have been triggered and only a small percentage allude to a genuine problem on the machine.

The prototype will now be taken forward in an attempt to realise a full on-line implementation which can be integrated with British Energy's system. There are some improvements to be made to the current prototype which should be undertaken before a full on-line version of the system is developed. There is the development of the explanation facility to more effectively link the rationale and feature extraction results as already discussed. There is also the updating of certain elements of the knowledge base, based on the feedback received from the testing. Improvements could also be made to the alarm screen to visually indicate the priority of alarms through some type of traffic light indication. The system also needs extended to automatically assess and interpret FFT data. This can only be achieved by involving both Beran and British Energy in discussions on how to gain access to the FFT data in a digital format. An approach to also capturing the data automatically from the Beran system should also be discussed as should methods of integrating the Expert System with British Energy's on-line network.

The final consideration is concerned with extending the current knowledge base so that the system extends its capability in the signals that are analysed, such as, steam, temperature and pressure etc, in addition to the level of depth in which the system assessments can go to. The system currently gives a relatively high level assessment particularly when it arrives at assessments of genuine faults. There are a vast array of genuine faults which could occur on turbine generators such as misalignment, cracked shaft, stiction to name just a few. One of the next stages of development should see the current knowledge base extended so that the level of detail provided by the system is extended to cover such specific fault types. The first option available to the system developers is to undertake more knowledge elicitation as outlined in section 4.3. The second option would be to investigate methods of automatically or at least semi-automatically capturing this knowledge in an attempt to speed up the notoriously time consuming and labour intensive knowledge engineering process. It was decided to research methods of implementing the second option to develop a method of assisting in the knowledge engineering process. An area of AI which shows the most potential in assisting with the development of new knowledge is Machine Learning (ML) as discussed in chapter 3. A learning module has been developed using an ML approach to assist in the knowledge engineering process for the British Energy turbine generator condition project. This learning module will now be described in detail in the following chapter.

Chapter 5

5 Learning Module

5.1 Introduction

One of the main challenges facing automated systems which exhibit intelligence is their ability to learn the knowledge required to interpret data. This was demonstrated in chapter 4 when a detailed description of the specification, design and performance of the novel Expert System used to analyse alarms from a turbine generator condition monitoring system was described. Although the system performs well in processing the real data, the time consuming and labour intensive nature of capturing the necessary knowledge was shown to be the primary drawback of such a technique. An area of AI which aims to alleviate this bottleneck by automatically capturing the knowledge which is capable of interpreting data is Machine Learning (ML). Various ML techniques were discussed as were examples of how these techniques have been applied in various domains in chapter 3.

This chapter describes the novel learning module, which was developed for the condition monitoring Expert System using an adapted version of Explanation Based Generalisation (EBG). Section 5.2 outlines the objectives of the learning module. Following this a full explanation of why the learning module was developed using an adapted version of EBG is given in section 5.3. A detailed description of its design which includes a review of the modelling formalism used by the module and an explanation of the algorithm adopted is given in sections 5.4 and 5.5. A worked example is referred to throughout this section to illustrate how the module performs its analysis. The final part of the chapter reviews the performance of the module when tested on real fault data taken form the Beran system in section 5.6 and an evaluation of the approach is given in section 5.7.

5.2 Learning Module Objectives

Chapter 4 demonstrated that one of the main drawbacks of using the Expert System approach for automated turbine generator condition monitoring is the time consuming and labour intensive process required to capture the diagnostic knowledge. Even with this drawback, Expert Systems are still an attractive option for strategically important applications such as turbine generator condition monitoring where explanation and transparency are vital in gaining user confidence in such systems. The explicit knowledge base provides a rationale and subsequently an explanation to support each assessment. This level of transparency is important for verifying correct diagnoses and even more so in cases of misdiagnosis, which will undoubtedly occur with any automated system, since it allows users and system maintenance engineers to locate and correct areas of vulnerability within the system.

A learning module has been developed to augment the Expert System developed for British Energy. The learning module's primary function is to assist the experts in deriving new heuristics where an appropriate expression does not exist. The primary indicator for using the module will be in situations where the existing Expert System knowledge is not able to correctly assess one of the Beran alarms or does not provide enough detail in its assessment. The Expert System will output the alarm assessment to the user for verification as shown in figure 5.1. Alarms verified by the expert are uploaded to the alarm database along with any additional comments, but the learning module is activated if the expert determines that the assessment is incorrect as shown in figure 5.1.

The learning module aims to reduce the burden placed on the expert in deriving knowledge for the Expert System. However, it is not expected to derive the knowledge completely automatically without any input from the expert, therefore he/she is included in the process. Firstly, the user is expected to instantiate the module on occasions where an incorrect or non-detailed assessment has been performed by the system. Since the user has deemed the assessment incorrect or not detailed enough it is assumed that he/she knows the correct assessment. Part of the process therefore requires the expert to upload the correct alarm cause to the learning module to guide the analysis as shown in figure 5.1. Additionally the user is expected to upload all of the information associated with the alarm which has been derived by the Expert System (channel profile) as shown in figure 5.1. This data acts as the training example for the learning module.

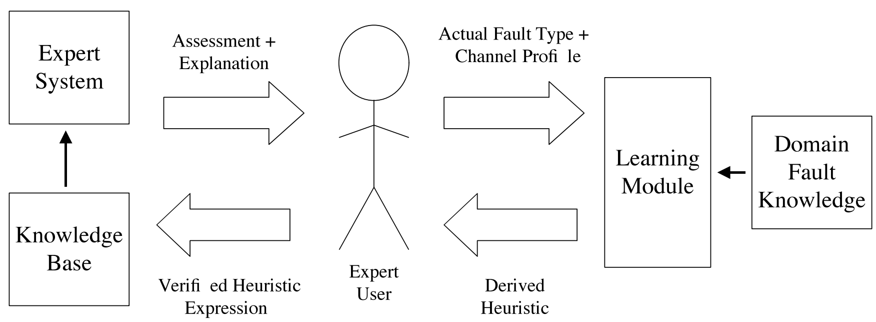


Figure 5.1: An overview of how the learning module integrates with the turbine generator condition monitoring approach.

The learning module will attempt to derive a suitable expression for the alarm using the correct classification and alarm information uploaded by the expert in conjunction with knowledge of the domain as shown in figure 5.1. Benefit can only be added to the project if the learning module were to reduce the amount of knowledge elicitation required over the course of the project, however this does not mean completely eliminating the knowledge engineering process. Therefore the knowledge used to assist in the derivation of the expressions may be captured through knowledge elicitation exercises carried out specifically for the capture of knowledge used by the learning module.

The Expert System will still be operating within a strategically important domain. Therefore, the knowledge derived by the learning module will still have to be verified before being uploaded to the system as shown in figure 5.1. This verification will be performed by the expert to ensure that a complete and correct heuristic is uploaded to the

Expert System's knowledge base. The expressions derived by the learning module should be clear and understandable to humans and compatible with the heuristic knowledge used by the Expert System. Additional explanation or rationale provided with the assessment will assist with the verification process.

The learning module objectives are to:

- Assist the experts in deriving turbine generator condition monitoring fault knowledge.
- Reduce but not eliminate the amount of knowledge elicitation required to derive heuristic expressions over the course of the project.
- Involve the expert in the knowledge derivation process by triggering the assessment and verifying the results.
- Derive knowledge which is compatible with the Expert System's knowledge base.
- Provide explanations to assist in verification of the expressions.
- Derive expressions from a single training example.

To emphasise how the module is intended to function a hypothetical example from the British Energy application will be considered. If the Expert System was to assess an alarm as being a genuine fault but on closer analysis of the data the expert deemed the cause to be a rotor rub, the module would be activated. Although the system is not incorrect in this instance the expert is adding detail to the assessment by indicating that the alarm is a rotor rub which is a sub-category of a genuine fault The expert would activate the module by uploading the channel profile derived by the Expert System in addition to the updated alarm cause. Before uploading the data the expert would have to check that the features extracted by the Expert System in the signal to symbol transformation were accurate, otherwise the cause of the incorrect feature extraction would have to be investigated. The learning module would then assess the uploaded data to derive an expression for a rotor rub fault which can then be verified by an expert. The main objective is to assist the expert in deriving new fault knowledge therefore the final expression derived by the module may have to be adapted or updated by the expert. Any explanations provided with the results which assist the expert would be of benefit.

5.3 Technique Selection

Given the objectives outlined in section 5.2 it can be surmised that the learning module developed for the Expert System would be required to derive knowledge in a format which is compatible with its knowledge base. Further to this the knowledge would have to be easily interpreted by a human expert during the verification process. Therefore a ML technique which can derive symbolic knowledge is more suited since it is easily interpreted by human experts and is compatible with the type of knowledge used in the Expert System detailed in chapter 4. This requirement eliminates the possibility of using ANNs since the knowledge derived is implicit within the network itself and is therefore not easily understood by humans. Another requirement of the selected ML technique was that it is capable of deriving knowledge from very few and in some instances single training examples since genuine turbine generator faults are rare. Furthermore, no laboratory set-up exists which can accurately simulate the faulty behaviour of a 660MW unit. Therefore, there is little training data available. This requirement eliminated the possibility of using techniques such as rule induction, Case-Based Reasoning (CBR) and statistical/probabilistic techniques such as HMMs since all of these techniques rely on a large training data set to derive knowledge.

Analytical learning is the only ML technique found during this research that is able to learn from single examples and where the resultant knowledge is symbolic based. Analytical learning techniques do however require background knowledge on the subject under investigation in order to determine what features are relevant to the problem. Fortunately this knowledge is available from the experts, but as stated in the learning module objectives, it is important that the level of knowledge elicitation is reduced otherwise the benefit provided by the module would be much diminished. The analytical learning approach adopted for the learning module was EBG. The EBG problem definition as defined in section 3.7.1 shows the greatest similarity with that of the learning module's problem. The learning module is required to construct a heuristic capable of identifying a particular fault which the Expert System is not able to assess. This requirement is facilitated in the EBG algorithm by the goal concept which is identified in the algorithms' problem definition in section 3.7.1. The learning module has access to fault diagnosis knowledge which will be captured from the experts. This knowledge is also identified in the EBG problem definition as the domain theory in section 3.7.1. The learning module has access to only a single training example since genuine turbine generator faults rarely occur. This training instance derived by the Expert System is identified in the EBG problem definition as the training example in section 3.7.1. Finally the learned heuristic must be in a form which is compatible with the existing knowledge and where the explicit symbols can be derived from the Beran condition monitoring data. This requirement is accounted for by the operationality criterion in the EBG problem definition in section 3.7.1. For this particular application the presence of an operationality criterion is an advantage since all of the expressions derived by the module would have to utilise symbols which can be derived by the Beran system, therefore the use of schemata as proposed by the EBL approach would be of no benefit.

It should be clear that the information required by the EBG problem definition outlined in section 3.7.1 is available in the turbine generator condition monitoring application. To summarise:

- goal concept – Fault type which new heuristic is required.
- domain theory - Fault diagnosis knowledge.
- training example – Channel profile derived by Expert System.
- operationality criterion – Features which can be derived from Beran data.

It should also be clear that the functionality provided by the EBG approach provides a means of fulfilling the objectives outlined in section 5.3. It is for this reason that the EBG approach was selected as the basis for the learning module approach. For reasons which are discussed later in section 5.4, the EBG approach in its purest form was not used. Instead an adapted novel version of the approach was developed and implemented for this particular application. Before these differences are investigated, it is necessary to discuss the knowledge modelling methodology adopted for the learning module. An in depth analysis of the knowledge modelling approach and the reasons for adopting this approach are discussed in the following section.

5.4 Knowledge Modelling

One of the difficulties with the EBG approach detailed in [Mitchell et al, 1986] was that in order for it to be effective at producing valid heuristics and not erroneous or inconsistent explanations the domain theory had to be complete and consistent. To produce a domain theory for turbine generator condition monitoring tasks using first order logic which is complete and consistent would at present be an insurmountable task. This is due to the level of complexity which arises in attempting to accurately model a domain which encompasses numerous technical disciplines such as thermodynamics, mechanical, fluid-dynamics, electromagnetism etc. A complete representation of all the theories, concepts and equations required to model the turbine generator domain is not possible. One of the difficulties with using first order logic is that the generalised theories have to be applicable over all of the data used to derive the heuristic expressions. The problem is that the 660MW turbine sets from which the real condition monitoring data is captured from are complex systems and therefore there would be many deviations from the generalised theories which describe such an area. One way of overcoming this would be to include further concepts which account for all of the exceptions from the generalised theory. Even if this was possible, which is questionable, it would be a complex and time consuming task.

An alternative to the use of first order logic would be to approximate the knowledge and make it less generalised and more specific to the problem being addressed much in the same way that production rules can be used to produce a generalisation of a domain theory. One approach which enables the knowledge to be made more specific to the problem area and allows for approximations to exist is that of causal modelling [Kobayashi & Nakamura, 1991]. Causal modelling attempts to represent the causal interactions present within a system when some event takes place. These causal interactions take place within a system in addition to the observable events detected. Causal models aim to merge the hidden causal events with observable ones to develop a more in-depth representation of the chain of interactions which lead to certain behaviours.

The most common approach to causal modelling is to take a concept and from that concept determine all of the causal interactions which would take place within the system in order for this concept to be realised. A conceptual example of this type of causal model is given in figure 5.2. Within the causal explanation would be observable states which could be used to recognise examples of the particular concept being analysed. There will also be unobservable states which are interactions which occur within a system but are hidden to normal methods of detection. Also within the model will be the concept itself. However the concept will be denoted as a standard node in the model in the same way as the observable and unobservable states are shown in figure 5.2. This approach to causal modelling has found application within second generation Expert Systems such as [Console & Torasso, 1988], [Console et al, 1989] and [Steels, 1985]. The causal knowledge used in these applications is sometimes referred to as "deep knowledge" [Steels, 1985] since the additional information provides further detail to the observable data.

One of the main disadvantages of using such an approach is that all of the causal interactions for a particular concept are mapped out when the model is developed. This is because the concept node is an integral part of the model and therefore has to be included from the start to define all of the relationships within the model. Therefore, all of the interactions which take place in a system for a given concept must be understood when developing the model. In practice this would require the knowledge elicitation process to work through a system's behaviour in the event of a fault and would, in effect, be very similar to that undertaken during Expert System development but with an additional deeper level of knowledge. Another facet of this approach is that there is relatively little structure to the nature of the observable event defined within the model. For example the causal events defined within the model in [Console & Torasso, 1988], [Console et al, 1989] are described using the type of language which may be expected of an expert during a knowledge elicitation interview. There are no rules governing the permitted states within the model, instead the expert would define such states as the knowledge engineering process proceeded.

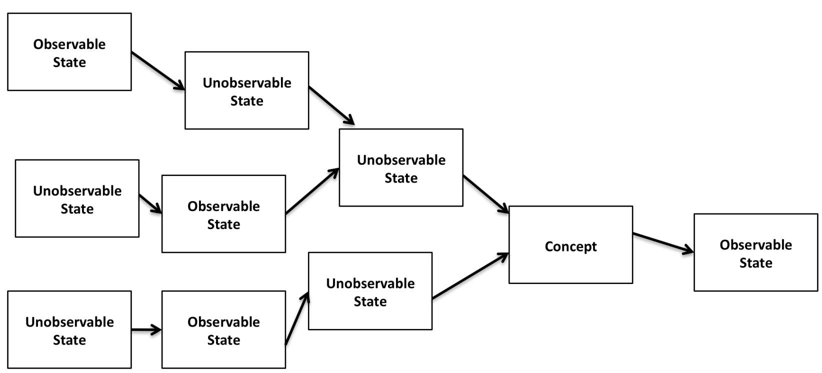


Figure 5.2: Traditional approach to causal modelling where all interactions for a system are defined up-front.

Another approach to causal modelling is to simply define the effect or effects associated with a cause or combination of causes at each stage in the process as opposed to prescribing every causal event associated with a particular behaviour. This is a more modular approach where the expert and knowledge engineer analyse the behaviour of a certain component of the system for a particular behaviour without considering the effect on the whole system. The behaviour of these components can be contained within a look-up table where the cause and effect relationships can be defined for each type of behaviour. A causal model can then be constructed from the look-up table behaviours as opposed to being explicitly defined at the knowledge elicitation stage. A conceptual example of the type of causal model constructed by such an approach is given in figure 5.3. Here it can be assumed that the majority of the causal nodes were generated by a default concept behaviour e.g. normal behaviour. There is however one node which has a concept node pointing towards it via a dashed arrow. This indicates that this node has been generated using a non-default concept behaviour e.g. faulty behaviour.

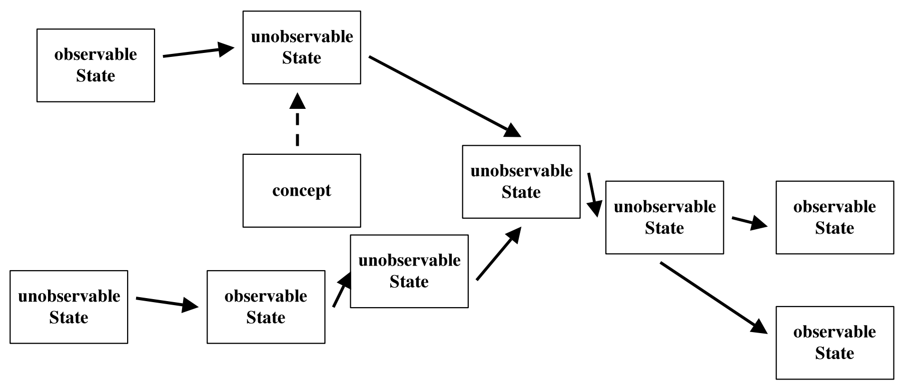


Figure 5.3: Alternative approach to causal modelling individual component interactions are defined across all behaviours and the derivation of the complete system behaviour is constructed using these individual behaviours.

This approach still enables a form of approximate reasoning which eliminates the problem associated with first order logic when used in EBG. It also has the advantage on not having to work through a complete system of behaviour for a particular fault type at the knowledge elicitation stage similar to that required in the development of Expert Systems. Instead the expert can simply define behaviours on a component by component basis leaving the overall system behaviour to be derived by combining these individual behaviours. For these reasons this modular approach to causal modelling was selected as the knowledge representation formalism for the novel adapted EBG approach used for the learning module. This approach is discussed in greater detail in the following sections.

5.4.1 Aims & Objectives

The aim of the causal knowledge modelling formalism for this project is twofold:

1. Eliminate the requirement for a complete understanding of the turbine generator behaviour under fault or normal behaviour conditions when the models are being developed.
2. Provide a formal structure for the modelling of the turbine generator behavioural knowledge.

The first aim was facilitated by adopting an approach to causal modelling which only defines the cause and effect of a particular action under a certain condition. This means that a complete understanding of the entire behaviour of the system under a particular condition did not have to be understood at the knowledge elicitation level. Instead, by combining all of the individual behaviours of each system component under a certain condition, a complete description of the system behaviour can be incrementally constructed without the need for an understanding of the whole system.

The second aim was realised by restricting the modelling of the turbine generator to defined components and properties. This was achieved by firstly defining the turbine generator in terms of its mechanical structure. The mechanical structure could then be easily divided into sub-structures such as generator, high pressure rotor, low pressure rotor, etc., and from this be further sub-divided into component parts. Properties associated with the component parts were then defined. For example the thermodynamic and electromechanical properties were identified and defined for each relevant component. From here the various states associated with the properties could be defined. The language in which the behaviour of the turbine generator could be described was restricted by formally defining the turbine generator set in this way. The causal models constructed for this project were limited to the mechanical structure of the turbine generator and the various properties derived from these components. For example the mechanical structure of a generator may be defined in terms of the components in figure 5.4. It can be seen that the sub-structure can be defined in terms of its components such as the shaft, stator, rotor etc.

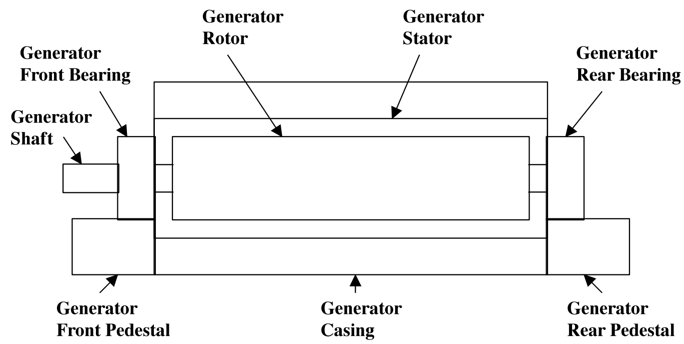


Figure 5.4: High-level overview of the mechanical structure of a generator.

Each individual component can then have its properties defined. Figure 5.5 lists some of the properties which may be identified with each generator component. For example, properties associated with the generator shaft are balance, dimension, temperature, vibration and speed. States for each of the properties can then be defined to complete the language in which the turbine generator behaviour is described. As an example, the temperature property might have its states defined as increase, decrease, high, low, etc.

The main advantage of using a modular approach to model the turbine generator is that various configurations can be constructed by combining the modules defined over multiple knowledge elicitation projects. This means that each configuration of turbine generator does not have to be taken on an individual basis for the purpose of knowledge elicitation. Instead typical sub-systems of varying types of turbine generator can be analysed over individual knowledge elicitation projects. These sub-systems can then be combined to create varying system configurations. For example, two turbine generators may each consist of a High Pressure (HP) turbine, Medium Pressure (MP) turbine, 3

Low Pressure (LP) turbines and a generator sub-system. If the LP modules were constructed using different blade types then this may lead to differing behaviours under normal and fault type conditions. If the knowledge modelling approach adopted required that all of the causal interactions under each condition had to be defined at the knowledge elicitation stage as used in [Console & Torasso, 1988], [Console et al, 1989] and [Kobayashi & Nakamura, 1991] then each turbine generator set would have to be treated as a separate entity. Therefore knowledge elicitation would have to be undertaken for both types for each condition of interest. Alternatively, if the modular approach, as adopted for the learning module, was used then models for each varying type of sub-system for each condition of interest can be developed through knowledge elicitation. The two turbine generator configurations can then be constructed by combining each of the relevant sub-systems. This means that the models developed for the HP, MP and generator can be reused and a separate analysis of both complete systems under the conditions of interest is avoided.

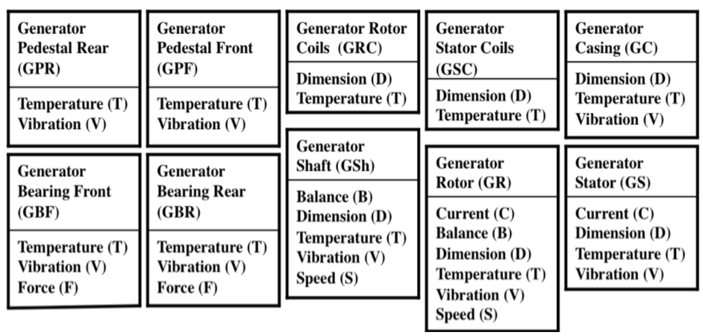


Figure 5.5: Properties associated with each of the components in the generator.

5.4.2 Formal Knowledge Modelling Approach

The previous section outlined the aims and objectives of the knowledge modelling approach for the learning module. This section describes how this has been achieved using a novel knowledge modelling formalism developed for this project. Section 5.4.2.1 describes the main component of the approach. These are the Look-Up Tables (LUTs) which are used to store the causal behaviours between component property pairs. Section 5.4.2.2 explains the use of '&' nodes in the LUTs to model behaviours which must be conjugated together. Section 5.4.2.3 explains the use of the 'no change' operator which allows the causal knowledge formalism to recognise instances where the there has been no change in a parameter following the change in some other parameter. Section 5.4.2.4 describes the use of operational and non-operational nodes and section 5.4.2.5 outlines the rules governing the definition of component/property pairs.

5.4.2.1 Look-Up Tables

The causal models, which represent the behaviour of the turbine generator components, are realised using LUTs. A LUT is constructed for each component/property pair and is defined for each behaviour of interest. All of the states associated with the property are then listed in the LUT and from this the affected component/property pairs are defined. An example of how each of the causal LUTs are formed is given in figure 5.6. The acronyms for each of the properties are defined in the LUTs. For example Generator Rotor Coil Temperature is represented as GRCT-OA. The OA part of the acronym stands for overall which implies it is the overall amplitude of the signal and not a 1^{st} or 2^{nd} order magnitude or phase plot. It is shown that changes in the generator rotor coil temperature cause changes in both the generator rotor coil dimension and the generator rotor temperature. Changes in the generator rotor coil dimension cause changes in the balance of the generator rotor. A stiction fault causes a change in the causal behaviour when there is a change in the generator rotor coil temperature. This is why there are two LUTs for the generator rotor coil temperature, one for normal behaviour and one for faulty stiction behaviour. It should be noted that the symptoms of a sticion fault are only apparent when the generator rotor coil temperature is decreasing. In this case, the generator rotor coil dimension contracts unevenly. No unique behaviour is exhibited under stiction fault conditions if the generator rotor coil temperature was to increase, which is why 'N/A' is the entry here for the generator rotor coil dimension. Instead only normal behaviour would be exhibited which is for the generator rotor coil dimension to expand evenly.

Normal Behavior		
Cause	**Effects**	
Generator Rotor Coil Temperature (GRCT-OA)	**Generator Rotor Coil Dimension (GRCD-OA)**	**Generator Rotor Temperature (GRT-OA)**
Change Increase (Ch-Inc)	Expand Even (EE)	Change Increase (Ch-Inc)
Change Decrease (Ch-Dec)	Contract Even (CE)	Change Decrease (Ch-Dec)
Level High (Lev-Hi)	N/A	N/A
Level Low (Lev-Lo)	N/A	N/A
Step Increase (St-Inc)	Expand Even (EE)	Change Increase (Ch-Inc)
Step Decrease (St-Dec)	Contract Even (CE)	Change Decrease (Ch-Dec)

Stiction		
Cause	**Effects**	
Generator Rotor Coil Temperature (GRCT-OA)	**Generator Rotor Coil Dimension (GRCD-OA)**	**Generator Rotor Temperature (GRT-OA)**
Change Increase (Ch-Inc)	N/A	Change Increase (Ch-Inc)
Change Decrease (Ch-Dec)	Contract Uneven (CU)	Change Decrease (Ch-Dec)
Level High (Lev-Hi)	N/A	N/A
Level Low (Lev-Lo)	N/A	N/A
Step Increase (St-Inc)	N/A	Change Increase (Ch-Inc)
Step Decrease (St-Dec)	Contract Uneven (CU)	Change Decrease (Ch-Dec)

Normal Behavior	
Cause	**Effects**
Generator Rotor Coil Dimension (GRCD-OA)	**Generator Rotor Balance (GRB-OA)**
Expand Even (EE)	Change Normal (Ch-Norm)
Expand Uneven (EU)	Change Significant (Ch-Sig)
Contract Even (CE)	Change Normal (Ch-Norm)
Contract Uneven (CU)	Change Negligible (Ch-Neg)

Figure 5.6: Example of typical look-up tables constructed which derive the causal models.

5.4.2.2 The '&' Operator

By default the effects which are derived from the causal LUTs are disjunctions of one another. For example the occurrence of cause 1 from the LUT in figure 5.7 results in effect 1 or effect 2 or effect 3. The modelling of turbine generator behaviour requires that certain causal events lead to multiple effects which must all be present for the initial causal event to have occurred. Therefore, causal effects which are the conjugation of one another must be defined in order for the knowledge modelling language to effectively represent the turbine generator domain.

Normal Behaviour			
Cause	**Effects**		
Component A	**Component B**	**Component C**	**Component D**
cause 1	**effect 1**	**effect 2**	**effect 3**
cause 2	**effect 4**	**effect 5**	**effect 6**
cause 3	**effect 7**	**effect 8**	**effect 9**

Figure 5.7: The effects of each cause within this look-up table are the disjunctions of one another

This requirement has been facilitated in the learning module modelling language by defining the special operator '&'. This operator is added to any effects which are required to be the conjunction of one another. Each '&' operator has associated with it a number to denote which effects are to be conjugated with one another. For example the occurrence of cause 1 from the LUT in figure 5.8 results in effect 1 and effect 2 being conjugated with one another, and effect 4 and effect 5 being conjugated together. Therefore cause 1 = (effect 1 $\wedge$ effect 2) $\vee$ effect 3 $\vee$ (effect 4 $\wedge$ effect 5), where, $\wedge$, is a conjunction and, $\vee$, is a disjunction. Notice that a numerical label is attached to each of the '&' operators to indicate which effect is conjugated with one another. When a node appears in an explanation tree as described in section 5.5.3 which is derived using an '&' operator then it is defined as an '&' node.

Normal Behaviour			
Cause	**Effects**		
Component A	**Component B**	**Component C**	**Component D**
cause 1	**effect 1 $^{\&1}$** **effect 2 $^{\&1}$**	**effect 3**	**effect 4 $^{\&2}$** **effect 5 $^{\&2}$**
cause 2	**effect 6**	**effect 7**	**effect 8**
cause 3	**effect 9**	**effect 10**	**effect 11**

Figure 5.8: Look-up tables which permit effects to be conjugated together using the '&' special operator.

5.4.2.3 No Change Operator

Causal models traditionally only permit the modelling of cause and effects. Both the cause and effects are changes which occur in a system and can be identified in some way. In addition to this, the effects by definition must occur after the cause. This property of causal modelling is a limitation within the turbine generator domain. In some instances the property of interest is the absence of an apparent change in a component in the event of some causal interaction. For example, if some property of a component was not to increase and instead stay the same in the event of some causal interaction then this may imply a certain type of behaviour. This phenomenon is traditionally not well modelled using causal models since there is an interest in the behaviour both before and after the causal event has occurred. Therefore a special state has been defined for the learning module which addresses this limitation. The special state is known simply as 'no change' and is denoted as NCh. When the causal model indicates that a certain component/parameter pair has a NCh state in relation to a causal event then this implies that the level property of the state should be unchanged both before and after the causal event.

5.4.2.4 Operational and Non-Operational Nodes

Another facet of the knowledge modelling formalism for the novel adapted version of EBG is the labelling of nodes in the explanation structure as either operational or non-operational. Explanation structures are derived by the learning module algorithm through the instantiating of the LUTs and are explained in greater detail in section 5.5.3. The explanation structures are constructed using nodes which are defined as either operational or non-operational nodes. This description relates back to the operationality criterion discussed in section 5.3. If a node is defined as operational then this implies that the Beran system is able to monitor and analyse the particular component defined within that node. For example, if a node contains the component generator rotor current then this would be defined as an operational node since this is a signal which is captured by the Beran system and can therefore be analysed. Alternatively, if a node is defined as non-operational then this implies that it cannot be directly accessed or analysed by the Beran system. For example, a node which contains the component generator rotor dimension would be defined as non-operational since the Beran system does not have access to information relating to the dimension of the generator rotor.

5.4.2.5 Component/Property Definition

In order to construct a LUT for each component/property pair, the relevant components and their associated pairs must be defined. This is achieved by firstly identifying each structural subsystem associated with the turbine generator under investigation. From here the components associated with each of the subsystems are identified along with their properties. All of this is defined by the condition monitoring experts during the knowledge elicitation exercises. It is important to note that any form of system modelling involves some level of abstraction. The causal modelling undertaken for the learning module project was in this respect no different. The condition monitoring experts were required to trade off the level of detail in which the components were modelled against the level of complexity. The general rule of thumb which emerged from the knowledge elicitation exercises was that components not used directly in the description of turbine generator behaviour under both normal and fault conditions were omitted. Once each component was identified the properties associated with them were defined.

5.5 Learning Module Approach

It has been demonstrated that the most suitable approach to modelling the turbine generator behavioural knowledge in this particular instance would be to adopt a modular causal approach. In adopting causal models the need for a complete and consistent domain theory as required in first order logic is eliminated and a modular approach allows the reusability of models over the course of the project. In addition to these advantages, the proposed approach allows the experts to analyse the system behaviour on a component by component basis therefore reducing the complexity of the elicitation process. The use of a causal method also enables approximate knowledge to be utilised by the adapted EBG approach developed for the British Energy turbine generator condition monitoring project. This adapted EBG approach is fully described this section. Section 5.5.2 describes the general approach used by the learning module. This is followed by an in depth analysis of the algorithm developed to achieve the novel adapted EBG approach in section 5.5.3. The in depth analysis is augmented by a worked example to assist in the explanation of how the learning module performs its analysis. The type of fault chosen for the worked example is a stiction fault. A brief explanation of a stiction fault is given in the following section.

5.5.1 Stiction Fault Description

Stiction faults occur when the rotor windings of the generator do not contract evenly after cooling, which in turn causes the rotor to become out of balance. The out of balance rotor results in a prolonged increase in the vibration magnitude of the generator rotor bearings. Stiction faults commonly manifest themselves following an increase in electric rotor current followed by a decrease. The increase in electric rotor current causes the rotor coil temperature to rise. The increased temperature evenly expands the rotor coils which displaces them within the rotor slots. A change in rotor balance results from the rotor coil displacement and this leads to a change in rotor bearing vibration magnitude.

A later decrease in the rotor current causes a temperature reduction in the rotor coils which causes them to contract. Two outcomes are possible from the contraction of the rotor coils. The first possibility is for the rotor coils to contract evenly and move back to their original position within the rotor slots. This would result in the rotor rebalancing itself, which causes the rotor bearing's vibration magnitude to recover to approximately the same level recorded prior to the rotor current increase. The second possibility is for the rotor coils to contract unevenly, maintaining the rotor in an out of balance state. The rotors continuing out of balance state would maintain a similar level of bearing vibration magnitude recorded prior to the increase in rotor current. This uneven contraction is an effect of the rotor coils not returning to their original position within the rotor slots and is caused by stiction.

5.5.2 General Approach

The learning module has adopted a novel EBG approach which uses causal fault and behavioural models, along with a single training example of a particular fault, to derive a heuristic expression capable of classifying the fault under analysis. The learning module firstly derives an explanation of why the training example is of a particular fault type using the causal fault and behavioural knowledge. Information is then selected from the causal explanation and used as symptoms for the resulting heuristic. The causal fault and behavioural models used by the learning module do not define the temporal constraints between each causal event. For example the time t between event A and B, plus the duration d of both event A and B are not defined. Instead, the temporal information contained in the training example is used to define the temporal constraints within the explanation and subsequently the heuristic.

All problems undertaken by the learning module are defined in terms of the following EBG problem definition:

- Target Concept: The fault type which the learning module must derive a heuristic for.
- Training Example: An example of the fault type being learned.
- Domain Theory: The causal fault and behavioural models.

- Operationality Criterion: Defines the explicit symbols which must be used to define the heuristic.

As with the EBG approach described in section 3.7.1, all of the learning module problems require a target concept. In practice this target concept will be the fault type defined by the expert which in the worked example is stiction. The training example used will have to be an example of the target concept and in practice the training example will be derived from the signal to symbol transformation performed by the Expert System. The training example for the stiction worked example is the channel profile given in table 5.1.

Table 5.1: example of the channel profile derived for a stiction fault.

Operational Profile	Vibration Profile	FFT Profile
GRC-OA Ch-Inc (0) (1) GRC-OA Ch-St (1) (6) GRC-OA Ch-Dec (7) (1) GRC-OA Ch-St (8) (5) GRC-OA Lev-St (0) (13) MVAR-OA Ch-St (0) (13) MVAR-OA Lev-St (0) (13) GL-OA Ch-St (0) (13) GL-OA Lev-St (0) (13)	GBRV-OA Ch-St (0) (2) GBRV-OA Ch-Inc (2) (1) GBRV-OA Ch-St (3) (10) GBRV-OA Lev-St (0) (3) GBRV-OA Lev-Hi (3) (10)	GBRV-50Hz P (5) GBRV-100Hz S (5) GBRV-SS Low (5)

The channel profile given in table 5.1 consists of facts regarding the behaviour of a rear rotor bearing. These facts are indicative of the symbolic data derived by the Extract Channel Profile module described in section 4.4. Each instance consists of a variable name, the type of signal, an abbreviation denoting the variable state and two temporal values within brackets. The variable is a combination of the structural element and the associated property. For example, the element generator rotor (GR) and the associated property current (C) is denoted as GRC. The type of signal defines if it is an overall amplitude signal or some order of the overall amplitude. For example OA denotes overall amplitude. The variable state defines if it is a change or a level and its state. For example Ch-Inc denotes that it is a change and is in an increased state. The first temporal value denotes when the event was triggered (manifestation time) and the second denotes its duration (period value). Figure 5.9 is an example of the variable generator rotor current overall amplitude in an increased change state with a manifestation time of 0 and period of 1.

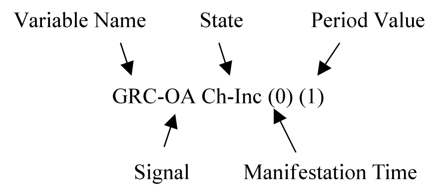


Figure 5.9: An example of how a generator rotor current variable which occurs at time period 0 and lasts for a period of 1 is represented in the channel profile.

The domain theory will consist of LUT models derived by the condition monitoring experts which follow the formal knowledge modelling approach outlined in section 5.4.2. The relevant LUT's for the stiction fault example are given in appendix C. Finally the operationality criterion is dictated by the parameters which can be derived by the Beran monitoring system as described in section 2.2. From this the learner must derive a heuristic which is sufficient for the fault type using explicit symbols defined in the operationality criterion.

The heuristic is derived using the following two stage process:

1. Explain: Construct an explanation of how the training example fulfills the target concept using the causal fault and behavioural models. Temporal information from the training example should be applied to the relevant nodes within the explanation structure.
2. Generalise: Determine a set of sufficient conditions on which the explanation holds by generalising the temporal data of the explanation structure. The resulting heuristic is the conjunction of each operational node of the generalised explanation structure.

Notice how the second stage in this approach is different from the one performed by the actual EBG approach described in section 3.7.1. The EBG approach employs a reduced form of regression to generalise the expression which is derived using the first order logic approach. This step cannot be performed on explanations which are derived using the causal modelling approach described in the previous section, however the temporal constraints within the explanation which are specific to the training example must be generalised.

If the expert concludes that the alarm was caused by a stiction fault then the learning module would be activated to help derive heuristics which could diagnose this particular fault. The learner firstly composes an explanation of the target concept using the channel profile given in table 5.1, the causal fault and behavioural knowledge stored as LUT's as described in section 5.4.2.1, and the algorithm which is explained in detail in the following section.

5.5.3 Formal Algorithm

The approach adopted by the learning module is a novel adapted version of EBG since the adopted knowledge modelling methodology uses causal models as opposed to first order logic. A formal algorithm therefore is required in order to implement a practical implementation of the general approach outlined in the previous section. The learning module algorithm is divided into the following three stages:

1. Generate causal explanations.
2. Generalise temporal constraints.
3. Derive heuristic.

A detailed explanation of each of the three stages now follows.

5.5.3.1 Generate Causal Explanations

The first stage which the learning module undertakes in attempting to derive a heuristic expression is to compile an explanation of the example being analysed. This explanation is required to verify that the training example is an instance of the fault type identified by the expert. The explanation is derived by verifying that the training example is consistent with the causal models for the expected behaviour. This is achieved by inputting the training example observables to the causal models and then matching the causal effects against the remaining observables in the training example. The first stage in generating the causal explanation is to upload the training example to the learning module along with the expected fault type. This is performed manually by the expert user. This data is then applied to the causal fault models to derive a causal explanation. The causal explanation is generated by running the training example through a novel algorithm developed for the learning module. The algorithm takes an observable from the training example and matches this against the causal look-up tables in an attempt to find an effect. As each effect is derived the observables in the training example are checked to determine if there are any matches. When a match is found against any of the effects then this indicates that the training data is consistent with that part of the model. This process is repeated for all of the observables in the training example. Once this is completed, any causal chains remaining, which have been shown to be consistent with the data and are of the type of behaviour of interest, are saved as the causal explanation.

The algorithm which generates this causal graph is governed by the following rules:

1. If an observed node is not of special state NCh, as described in section 5.4.2.3, it must have a manifestation time greater than the last observed node in the branch.
2. The level property of an observed node of special state NCh, as described in section 5.4.2.3, must not change state during the causal change.

3. A causal explanation can only begin and end on an operational node.
4. A causal explanation must contain more than one node.
5. A branch terminates if no LUT exists for the node under analysis.
6. If a branch is terminated which contains a special operator '&' node, as described in section 5.4.2.2, then all branches which contain a related special operator & node are terminated.
7. A completed explanation is terminated if it does not contain a node denoting the behaviour of interest.
8. If the node under analysis has both a fault and normal behaviour LUT then both are used.
9. If the node under analysis does not have a fault LUT matching the fault under analysis then the normal behaviour LUT is used.
10. If there is data in the profile which matches a node in the chain and meets either the first or second rule then the temporal data is added to the node.
11. A branch terminates if the number of un-observed nodes exceeds the allowed maximum.
12. A branch terminates if a node matches that of an earlier node in the branch and there are no observed nodes between the two of them.

The algorithm which implements these rules and generates the causal explanation by applying the training data to the causal knowledge is summarised by the activity diagram in figure 5.10.

The algorithm starts by selecting an event from the profile in order to find a root node for the causal graph. The first stage of the algorithm is concerned with assessing all of the unread nodes in the graph to determine if they generate any causal effects. This is achieved by searching the causal look-up tables to determine if any of them match with the component/property pair. The algorithm at all times generates as many causal branches as possible. Therefore if both a normal behaviour and fault behaviour look-up table exists then both are used to generate additional causal effects as defined in algorithm rule 8. If only a single look-up table exists, either fault or normal behaviour, then that look-up table is used to generate the causal effects as defined in algorithm rule 9. If no LUT exists for the event under analysis then the branch is terminated as defined in algorithm rule 5. This process is repeated for all of the unread nodes in the graph.

The next stage in the algorithm aims to identify observables in the training example which match against the causes generated by the causal behavioural knowledge. The algorithm attempts to determine if the model generated by the causal knowledge is consistent with the training data, therefore indicating that the turbine generator set is behaving in a particular fashion. It starts by selecting a cause node, if available, and determining if the node already exists in its particular branch. This check is put in place to determine if the causal graph will be caught in an infinite loop. If the node is in the branch and there is no observed node between them then the branch is terminated as defined in algorithm rule 12. Alternatively if there is an observed node between them or if the node does not already exist in the branch then the analysis continues. Firstly it must be determined if the node is operational as described in section 5.4.2.4. A non operational node will not exist in the training data whereas an operational node might. If the node is operational then the training data is checked for a potentially matching event. If a matching event is found within the training data then its state is checked to determine if it is NCh as defined in algorithm rule 2. This is because NCh is a special operator, which requires additional interpretation as explained in section 5.4.2.3. If the state is not NCh and a matching event is found then the temporal data is checked to determine if the event occurred after that of the last observed causal event in the branch as defined in algorithm rule 1. If the temporal data fulfils this criterion then the status of the node in the causal graph is set to 'observed', which entails the temporal data being added to that node. If any of these conditions are not met then the branch under analysis is checked to ensure that it does not breach the maximum allowed unobserved iterations. If it does the branch is terminated as defined in algorithm rule 11, if not the whole process is repeated for any remaining cause nodes.

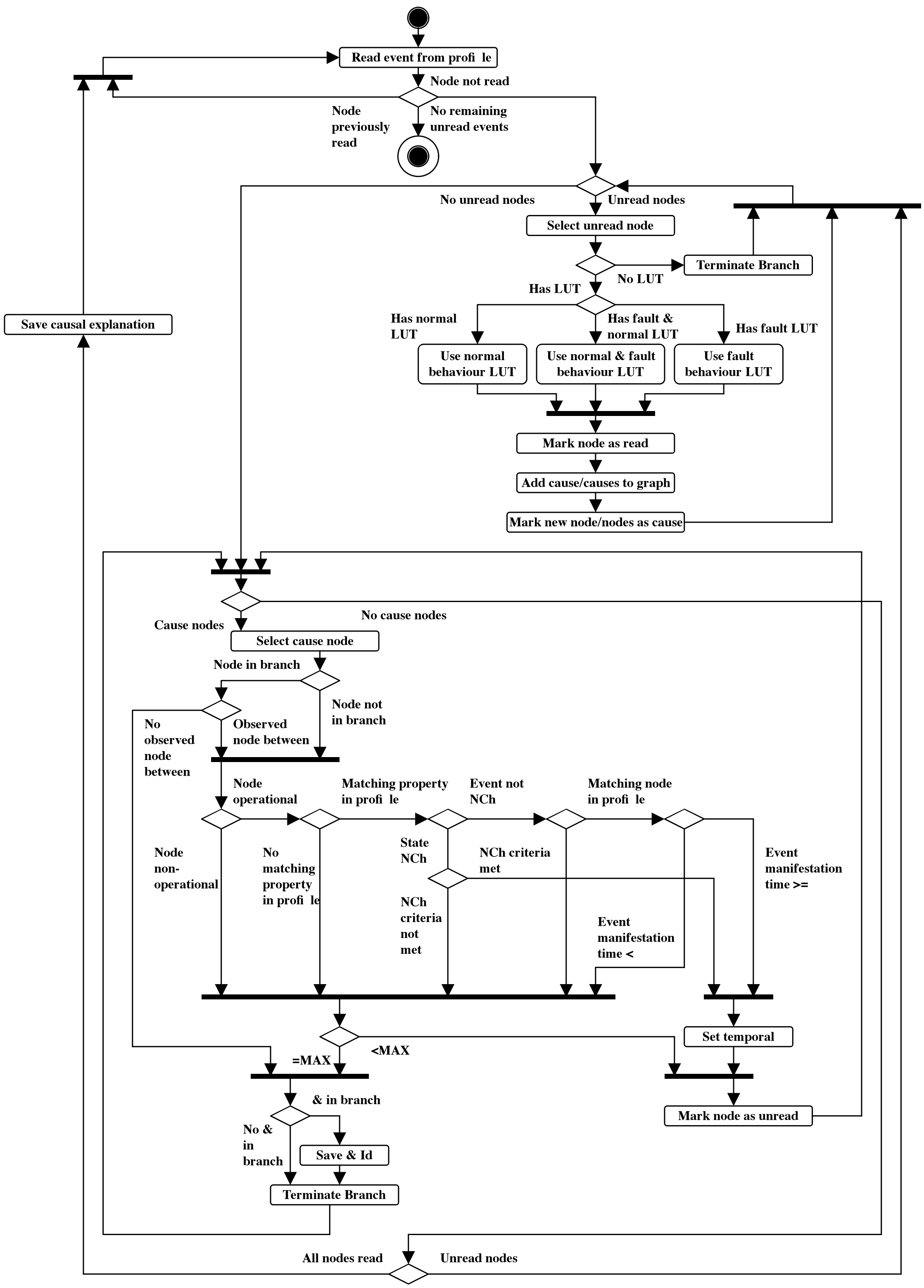


Figure 5.10: Activity diagram which details the algorithm used to generate the causal explanations for the learning module.

All of the analysis outlined above is repeated until all of the nodes are read. When all of the nodes are read, the causal explanation is saved. The activity diagram in figure 5.11 highlights how each causal explanation is saved. Firstly each of the branches are checked for '&' nodes present, as described in section 5.4.2.2, where the corresponding '&' node has been deleted. Where this type of node is found, the branch in which it is present is deleted as defined in algorithm rule 6. Following this, any branches which do not contain a node which indicates the behaviour type of interest are deleted as defined in algorithm rule 7. Any branches which are greater than one node are saved and any branches equal to one are deleted as defined in algorithm rule 4. It should be noted that the only time a branch of one node exists is where there is only one node in the causal explanation i.e. the event is not consistent with the causal knowledge.

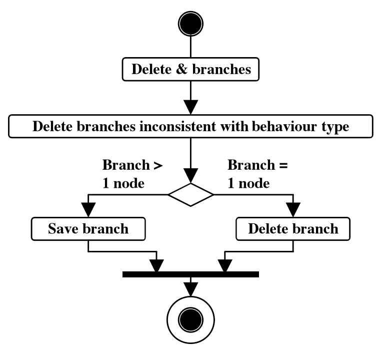


Figure 5.11: Activity diagram which details the steps taken in saving an explanation.

The activity diagram in Figure 5.12 demonstrates the steps taken in terminating a branch within the explanation. When a branch is terminated, the branch is checked for any '&' nodes as described in section 5.4.2.2. If an '&' node does exist, it is saved for reference at the save causal explanation stage so that the corresponding '&' nodes can be deleted. The branch is then deleted up to the last observed node.

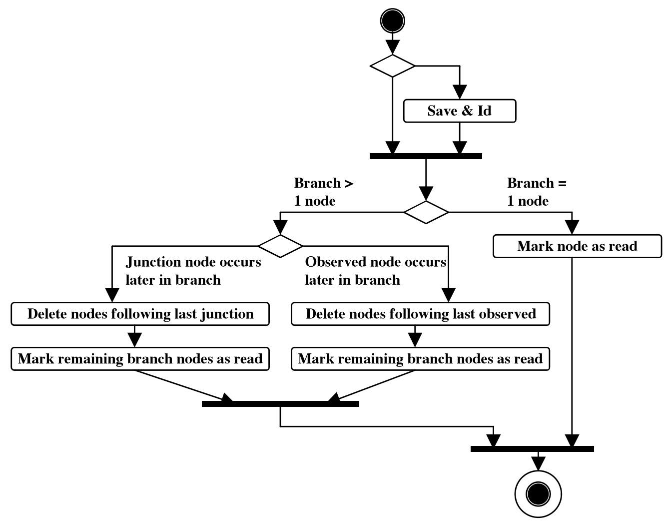


Figure 5.12: Activity diagram which details the steps taken in terminating a branch.

More explanation on how the algorithm derives explanations will now be given by reverting back to the stiction worked example. If the expert has verified that the channel profile given in table 5.1 is an example of stiction then the learning module can be activated by uploading the fault type stiction and the channel profile. This will in turn activate the algorithm. The algorithm will take each state from the channel profile in turn and match them against each LUT in the knowledge base in an attempt to generate a causal explanation. For the given stiction example the explanation structure is activated by the GRC-OA Ch-Dec (7) (1) state given in the channel profile in table 5.1. This becomes the root node in the stiction explanation tree. The algorithm searches the knowledge base for a LUT which corresponds with this node and finds the Generator Rotor Current LUT for normal behaviour given in figure C.1 in appendix C. This LUT instantiates the explanation structure given in figure 5.13, in accordance with algorithm rule number 9.

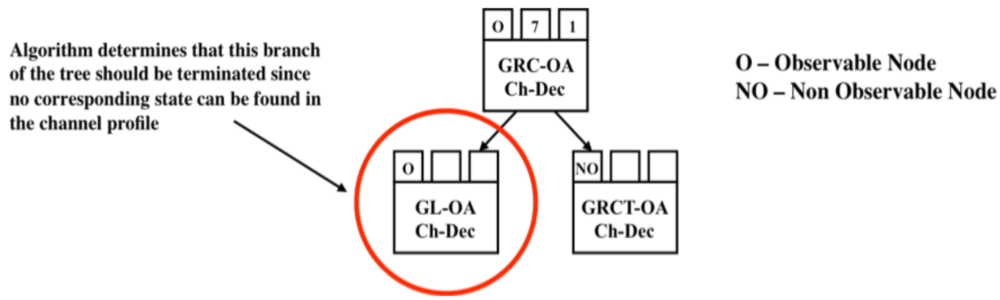


Figure 5.13: Explanation generated from the Generator Rotor Current normal behaviour LUT.

The algorithm will recognise the GL-OA Ch-Dec as an operational node as described in section 5.4.2.4 and therefore searches the channel profile in figure 5.1 to determine if there are any states which match against this node. The algorithm will also check that any matching states occur following the root note in accordance with algorithm rule number 1 i.e. the manifestation time is greater than the 7 given in the root node. No suitable state is found. The algorithm then searches the knowledge base for LUTs which relate to any of the nodes given in the explanation tree. No LUT is found for the GL-OA node, therefore this branch is terminated in accordance with algorithm rule number 5. The algorithm will then move to the remaining node in the explanation. The node is non-operational as explained in section 5.4.2.4, therefore the algorithm does not search the channel profile to determine if there is a relevant state. Instead the knowledge base is searched to determine if there is a suitable LUT. The normal behaviour and stiction fault behaviour LUTs for the Generator Rotor Coil Temperature state, given in figures C.3 and C.2 in appendix C respectively, are used to give the explanation structure in figure 5.14. This is in accordance with algorithm rule number 8.

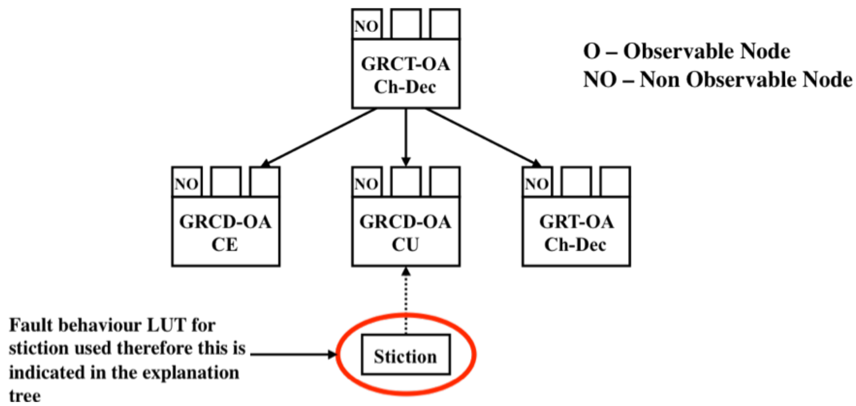


Figure 5.14: Explanation generated from the Generator Rotor Coil Temperature normal and stiction fault behaviour LUTs.

The node generated from the stiction behaviour LUT is denoted by the stiction node. All of the nodes are non-operational as described in section 5.4.2.4, therefore the algorithm does not search for any relevant states in the channel profile given in table 5.1. Instead the knowledge base is checked for associated LUTs. The LUT for the Generator Rotor Temperature node, given in figure C.6 in appendix C, generates the explanation given in figure 5.15 in accordance with algorithm rule number 9.

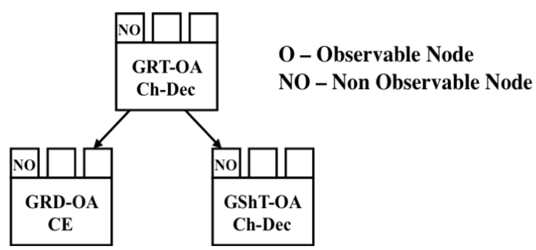


Figure 5.15: Explanation generated from the Generator Rotor Temperature normal behaviour LUT.

Both of these nodes are non-operational as explained in section 5.4.2.4, therefore the algorithm does not search for any relevant states in the channel profile given in table 5.1. The knowledge base is therefore checked for associated LUTs. The LUT for the Generator Rotor Dimension along with the Generator Rotor Balance LUT, presented in figures C.9 and C.4 in appendix C respectively, generate the explanation tree given in figure 5.16 in accordance with algorithm rule number 9.

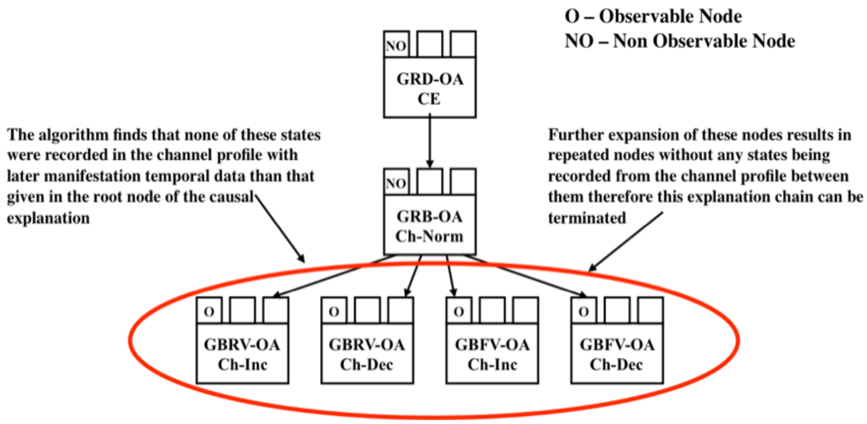


Figure 5.16: Explanation generated from the Generator Rotor Balance and Generator Rotor Dimension normal behaviour LUTs.

The algorithm finds that none of the operational nodes circled in figure 5.16 fulfil algorithm rule number 1. That is, having an associated state in the channel profile in table 5.1 which has an instantiation temporal value occurring later than that at the root of the explanation in figure 5.13, i.e. later than 7. The algorithm further expands this explanation tree only to find that nodes are repeated further down the branches. All of these branches can therefore be terminated back up the tree to the GRT-OA Ch-Dec node in figure 5.15 in accordance with algorithm rule number 12. This is because there are no nodes between those repeated which have been recorded in the channel profile.

The next stage in the process is to analyse the GShT-OA Ch-Dec node given in figure 5.15. The explanation tree, given in figure 5.17, is generated in accordance with algorithm rule number 9, using the normal behaviour LUTs for Generator Shaft Temperature, Generator Shaft Dimension and Generator Rotor Balance presented in figures C.7, C.8 and C.4 in appendix C.

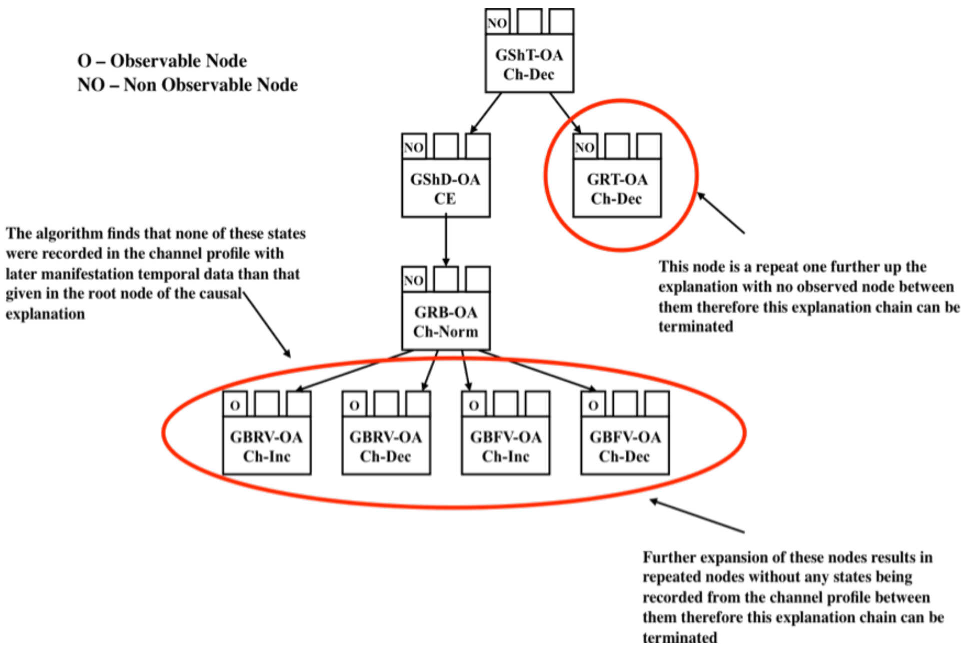


Figure 5.17: Explanation generated from the Generator Rotor Balance, Generator Shaft Temperature and Generator Shaft Dimension normal behaviour LUTs.

The GRT-OA Ch-Dec node is a repeat of that further up that particular branch of the explanation. There is no node within this branch which relates to any of the information in the channel profile therefore this branch is terminated up to the GShT-OA Ch-Dec node in accordance with algorithm rule number 12. The remaining branches are also terminated in accordance with algorithm rule number 12 for the same reasons given for the previous explanation structure in figure 5.16.

The algorithm then moves back up the explanation tree to assess the GRCD-OA CE node given in the in figure 5.14. This node generates the explanation structure in figure 5.18 in accordance with algorithm rule number 9 using the normal behaviour LUTs for Generator Rotor Coil Dimension and Generator Rotor Balance, presented in figures C.5 and C.4 in appendix C.

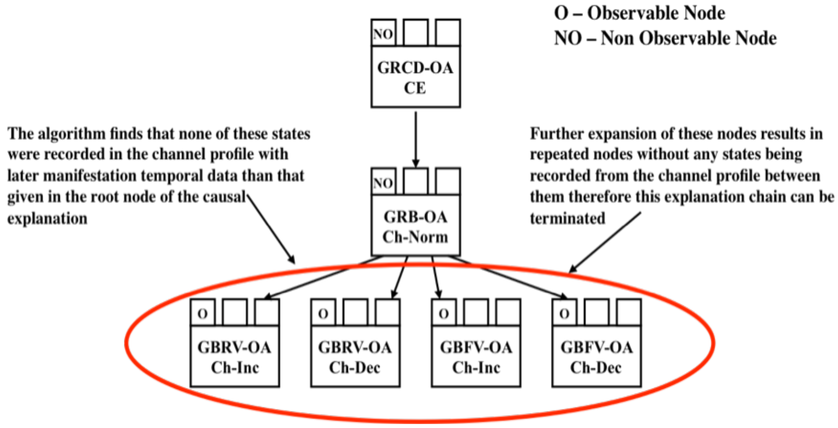


Figure 5.18: Explanation generated from the Generator Rotor Coil Dimension and Generator Rotor Balance normal behaviour LUTs.

The algorithm is able to terminate all of the branches in figure 5.18 using algorithm rule number 12 for the same reasons given in the previous two examples shown in figures 5.16 and 5.17.

The final node in figure 5.14 which the learning algorithm generates an explanation structure for is the stiction fault node GRCD-OA CU. The algorithm uses the normal behaviour LUTs for Generator Rotor Coil Dimension and Generator Rotor Balance to construct the explanation given in figure 5.19 in accordance with algorithm rule number 9.

Both of the branch nodes circled in figure 5.19 are operational as described in section 5.4.2.4. Also both nodes include the special operator no change (NCh) discussed in section 5.4.2.3. This operator requires a special condition in order to find a match within the channel profile. This special condition is defined in algorithm rules 1 and 2. The algorithm will search the channel profile to determine if there are any GBFV-OA or GBRV-OA level states which are unchanged from before and after the state which triggered the explanation i.e. the root node. The manifestation time of the root node is 7 as shown in figure 5.13. Therefore, to meet the NCh criteria the state would require a manifestation time less than 7 but a finishing time greater 7. The GBRV-OA Lev-Hi (3)

(10) state listed in the channel profile in table 5.1 fulfils this NCh condition since the manifestation time occurs at 3 and the end time is 13 (3+10). This state can therefore be added to the explanation tree. The temporal data is added to the node and the state is changed from Lev-Hi in the profile to Lev-NCh in the explanation tree.

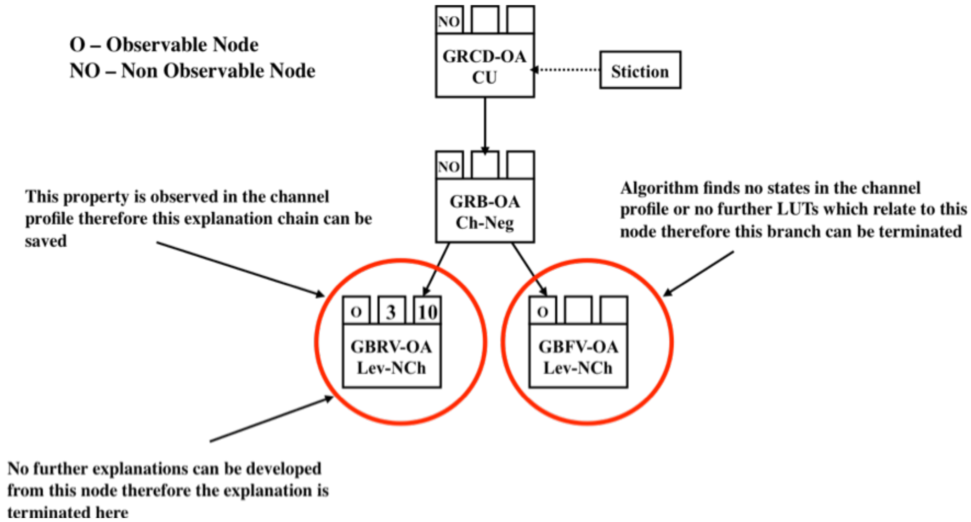


Figure 5.19: Explanation generated from the Generator Rotor Coil Dimension and Generator Rotor Balance normal behaviour LUTs.

This explanation structure is consistent with the channel profile since it starts with an observed operational node and terminates with an observed operational node as defined in algorithm rule 3 and can therefore be saved. The algorithm also checks the channel profile for any states consistent with the 'GBFV-OA Lev-NCh' node but there are none. Therefore that node is deleted in accordance with algorithm rule 5. The resulting explanation is shown in figure 5.20.

The above process described above is repeated for all of the states in the channel profile. Any explanations which are shown to be consistent with the channel profile are saved by the algorithm for further analysis. Once the algorithm has completed its analysis the expert must verify that all of the explanations are valid for the target concept.

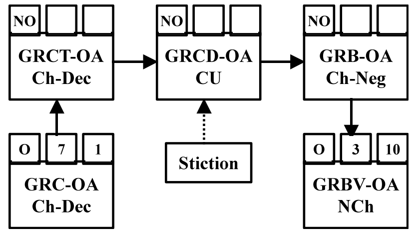


Figure 5.20: Final explanation structure generated for the stiction example given in table 5.1.

5.5.3.2 Explanation Validation

The first stage in which the user is required to analyse the module results is following the explanation generation stage. It is entirely possible that the module produces some explanations which comply with the causal knowledge but are not directly related to the target concept heuristic expression. It is important that these explanations are identified and eliminated at this stage as opposed to being filtered through to the derived heuristic. It is easier for the expert to eliminate an invalid explanation at this stage because the causal explanation is still present which assists in the verification. This would be more difficult to achieve if left until the heuristic is derived since the rationale behind all terms in the expression is not present.

The expert therefore examines each of the causal chains generated for each explanation to ensure that the reasoning generated by the learning module is correct and relevant to the fault under analysis. In this example the only explanation generated by the algorithm for the stiction fault example deemed valid is in figure 5.20. The next stage following the validation of the explanation is to generalise the temporal constraints associated with the explanation. The process for generalising the temporal constraints of an explanation structure is described in the following section.

5.5.3.3 Generalise Temporal Constraints

The next stage in the learning module approach is to generalise the manifestation temporal constraints contained within the causal explanations. Two types of temporal constraints are assigned to any observed feature within the explanation structure. These are the manifestation time and the period value. The manifestation time denotes the time in which the event started to occur. The period value indicates how long the event occurred for. The exact time in which the event starts to occur is not important when defining heuristic knowledge. For example a causal explanation for a particular fault type may indicate that an observable event occurred on 1st June 2006 at 12.45. Not every other example of this fault requires the observable event to occur at that exact time. Therefore, the instantiation time within the causal explanation structure is particular to the training example used to derive the model. These instantiation times can be generalised so that other instances which show the same characteristics but at different times can be recognised by the derived expression. This is achieved by searching the saved causal explanations to find the earliest manifestation time. The earliest manifestation time is therefore substituted by t and all further manifestation times are substituted with (t + original manifestation time – earliest manifestation time). The generalisation of the stiction fault explanation tree in figure 5.20, results in the explanation given in figure 5.21.

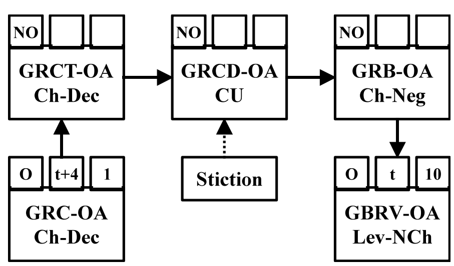


Figure 5.21: Explanation structure with its temporal constraints generalised.

The period temporal constraint is treated differently from the instantiation value because the length of time in which an event occurs or the time taken for an event to occur in relation to another event is of importance to classify other examples of that particular classification. These constraints, however, can't be further generalised automatically without performing some inductive interpretation on the expression which in this application is not possible due to the lack of training data. This means that the period constraints of the derived heuristic will be particular to the training example used and so another approach to generalising these must be used. The method of achieving this is discussed in section 5.5.4.

5.5.3.4 Derive Heuristic

The final step in the learning module approach is to derive the heuristic expression from the causal explanations. This is achieved by simply conjugating together all of the operational nodes which appear in the causal explanations. Therefore the heuristic generated by the learning module for the stiction fault example would be that given in figure 5.22.

Fault:	Stiction
Symptoms:	GRC-OA Ch-Dec (t+4)(1)
	GBRV-OA Lev- NCh (t)(10)

Figure 5.22: Heuristic derived from the explanation structure given in figure 5.21.

The heuristic proposed by the learning module for a stiction fault indicates that a decrease in generator rotor current lasting a period of 1 should cause no change in the level of the rear generator bearing vibration for a period of 10. The manifestation temporal data indicates that the decrease in generator rotor current should be preceded by a period of 4 by the no level change in rear generator bearing vibration,

5.5.4 Heuristic Verification

The learning module approach was not designed to be a fully automated system which produced a comprehensive heuristic expression on its own. Instead the module has been designed to assist the expert in deriving heuristic knowledge, therefore, interaction with the expert user is required. The user input primarily focuses on verifying the module results at two key stages. These stages are:

- Verification of explanation
- Verification of heuristic

The explanation verification stage was described in section 5.5.3.2. The second verification stage is there to perform multiple functions. As has already been explained in section 5.5.3.3 the period temporal constraints of the derived heuristic will be particular to the training example used to derive the expression. These temporal constraints must be generalised further in order to allow further examples of that concept to be properly assessed. One possibility is to automatically derive further generalised temporal constraints using a data intensive symbolic machine learning method such as candidate elimination specific to general search [Mitchell, 1982]. This method could further generalise the temporal constraints so that the heuristic can correctly assess all of the training examples. However, as discussed in section 5.3 there is a lack of training data available due to the rarity of genuine faults on the turbine generator which prohibits the use of such a technique. Instead the expert user is required to call upon their knowledge and experience to perform this function. They must make a judgement on what temporal constraints should be included within the heuristic so that it is an adequate general description of the target concept.

Another consideration at this stage is whether any important parameters to the target concept have been omitted from the heuristic. If the expert felt that any features should be added then this would have to be facilitated through discussions with them and the knowledge engineers. This review of the heuristic must be carried out in strategically important applications such as this. The users have to be sure that the knowledge being uploaded to the knowledge base encompasses their full range of experience so that the assessments performed by the Expert System are as accurate as possible.

The previous sections have described in detail how the learning module assists the expert in deriving heuristic fault knowledge for the Expert System. It has explained the novel adapted version of EBG used by the learning module including the novel causal knowledge modelling formalism and learning algorithm developed specifically for this project. These sections have also described how the expert user interacts with the learning module to develop and refine the results. All of these aspects of the approach have been explained using a worked example to show how the learning module results are derived in practice. The following section will now analyse how the learning module performed on genuine fault data taken from one of British Energy's turbine generators.

5.6 Learning Module Results

This section shows how the learning module performed when tested on real turbine generator condition monitoring data taken from British Energy's Beran system. A British Energy condition monitoring expert was asked to select data from the Beran system which were examples of genuine faults. The purpose of this was to test the learning module's capabilities when exposed to real condition monitoring data. As already explained in section 5.3, genuine faults on British Energy turbine generator sets are rare occurrences, therefore it was difficult to find suitable data. There was however an instance in July/August of 2006 where one of the units was scheduled for a maintenance outage. Prior to the outage it was acknowledged by the condition monitoring team that the set was showing typical signs of stiction. Stiction is a problem which affects the generator and is fully explained in section 5.5.1. Once the turbine generator was put back on-line the condition monitoring team found signs of an additional problem known as looseness.

5.6.1 Looseness Fault Description

A looseness fault on a turbine generator describes a behaviour which manifests itself from a pedestal not sufficiently secured to its foundations. Looseness does not commonly occur while the turbine generator set is running on-line, since it is unlikely that any significant change will occur between any of the pedestals and its foundations. Instead it normally occurs following an outage when there is normally some kind of overhaul or maintenance undertaken on the equipment. This maintenance normally involves components being taken out for repair or inspection which subsequently increases the chances of components becoming less secure when re-installed. When looseness occurs the affected pedestal suffers from higher levels of 2nd order vibration in the affected bearings with the increasing speed of the turbine generator set. The components affected by looseness problems are primarily the pedestals and the bearings supported by the pedestals. The pedestals mechanically support the bearing and so the increase in 2nd order vibration in the pedestal is reverberated through to the bearings.

These behaviour types are not catastrophic faults which would cause any significant level of damage to the turbine generator sets, but they do demonstrate distinguishable states which the condition monitoring experts use to monitor the condition of the turbine generator sets. The data is therefore a suitable test for the learning module to demonstrate that the learning module approach outlined in section 5.5 could be used by the company experts and system maintenance engineers to assist in deriving fault diagnostic knowledge.

The data was captured from one of British Energy's nuclear power stations. The turbine analysed is known as Turbine 2. The first set of data was captured between the period of

15/06/2006 and 02/08/2006 prior to the turbine being run-down. The second set of data was captured during the period of 14/08/2006 and 29/08/2006 following the shutdown. The performance of the learning module on both of these real faults is reviewed in the following sections.

5.6.2 Stiction Fault Training Example

The data selected by the British Energy condition monitoring expert as an example of stiction fault data was from 15/06/2006 till 02/08/2006. The overall, 1st order magnitude and phase, 2nd order magnitude and phase, and temperature data for bearings 6, 7, 8, 9 and 10, as well as the generator load, generator MVArs, generator rotor current and rpm data were selected. All of this data can be viewed in graphical format in appendix D. The profile for the data was generated by the Expert System described in chapter 4 and can be found in appendix D. The explanation structures generated by the learning module are given in figure 5.23. All of the LUTs which were used to build this explanation can be found in appendix C. The behaviours which are not enclosed within a box denote normal behaviour, whereas those enclosed within a box denote that they have been derived from stiction fault behaviour. The first bracketed number is the manifestation temporal value and the second is the period temporal value for that node. Both of these values are in samples where one sample equates to 10 minutes. If there is no number in the brackets and reads "null" then this implies that the node was not observed in the channel profile data and there is therefore no temporal data associated with it.

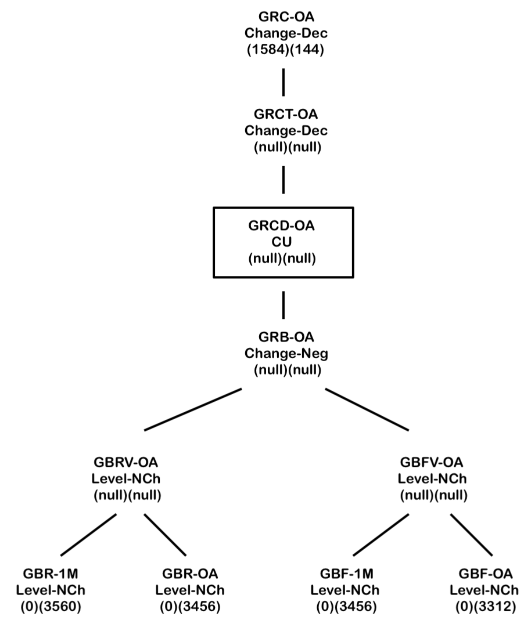


Figure 5.23(a): Explanation structure generated by learning module for a stiction fault. Expert confirmed that this explanation did not locate the actual fault.

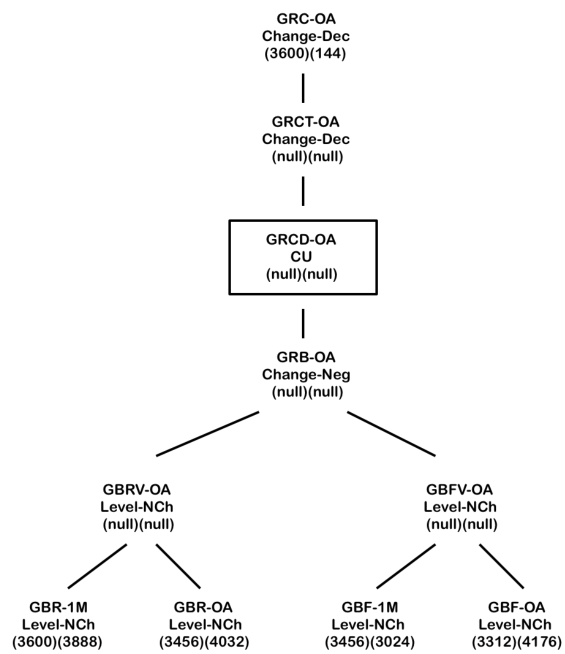


Figure 5.23(b): Explanation structure generated by learning module for a stiction fault. Confirmed by the expert as having located the fault.

A description of each abbreviation in the stiction fault explanations is given in table 5.2.

Table 5.2: Description of the abbreviations found in the stiction fault explanation in figure 5.23.

Abbreviations	Full Text
GRC	Generator Rotor Current
GRCT	Generator Rotor Current Temperature
GRCD	Generator Rotor Current Dimension
GRB	Generator Rotor Balance
GBRV	Generator Bearing Rear Vibration
GBFV	Generator Bearing Front Vibration
GBR	Generator Bearing Rear
GBF	Generator Bearing Front
1M	1st Order Magnitude
OA	Overall Amplitude
NCh	No Change
Neg	Negligible
CU	Contract Uneven
Dec	Decrease
Hi	High

Both graphs in figure 5.23 possess nodes which are derived from a stiction fault behavioural LUT. This implies that the profile derived from the real data exhibits behaviour which is consistent with that of a stiction fault according to the behavioural models. The explanation in figure 5.23(a) shows that a decrease in generator rotor current starting at sample 1584 (22/06/2006) causes a decrease in the generator rotor coil temperature whereas the explanation in figure 5.23(b) relates it to a decrease which starts at sample 3600 (06/07/2006). Under stiction fault conditions this decrease would cause the generator rotor coil dimension to contract unevenly which would result in the generator rotor balance undergoing a negligible change. This negligible change would mean that the vibration profile of both generator bearings would not change which is verified by both the 1st order magnitude and overall level parameters not changing over the period in which the initiating generator rotor current decrease took place.

Both explanations were structurally the same but on analysis of the raw data the expert concluded that the explanation structure in figure 5.23(b) had located the rise in rotor current and subsequent no change in vibration which indicated a stiction fault. Therefore the explanation given in figure 5.23(b) was put forward for the instantiation constraints to be further generalised. The result of which is given in figure 5.24.

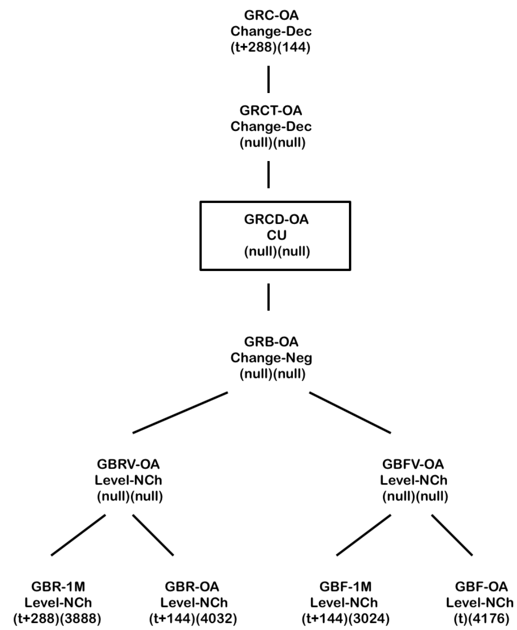


Figure 5.24: Stiction explanation structure after manifestation temporal constraints have been generalised.

The generalised explanation structure was then used to compile a heuristic. The results of the heuristic composition are shown in figure 5.25.

Fault: Stiction

Symptoms:
GRC-OA Change-Dec (t+288)(144)
GBR-1M Level-NCh (t+288)(3888)
GBR-OA Level-NCh (t+144)(4032)
GBF-1M Level-NCh (t+144)(3024)
GBF-OA Level-NCh (t)(4176)

Figure 5.25: Stiction Heuristic derived by the learning module from the Beran Data.

This heuristic, although correct, for the training data set will not pick up all examples of stiction fault behaviour from all data sets due to the specificity of the temporal constraints. Therefore the heuristic temporal constraints act as a guide for the expert to use his/her experience and expertise to derive more general ones. In this instance the expert felt that providing there is no change in the overall and 1st order vibration magnitudes following the decrease in rotor current then this would be consistent with stiction behaviour. Therefore there is no requirement for the change in rotor current to be delayed which means that the temporal constraint for the rotor current decrease could be set to t as opposed to t + 288. In addition the instantiation constraints for all of the remaining no change features can be set to <= t since the only requirement is for there to be no change following the decrease. The length of the no change in vibration and the decrease in rotor current was more debatable but the expert felt that for a turbine generator set behaving normally there would be a decrease in vibration within 15 to 60 minutes following the decrease in rotor current. Any delay beyond the hour would start to suggest a stiction fault. Therefore a no change time period of at least an hour (6 samples) would indicate stiction behaviour. As for the duration of a change in rotor current, the expert felt that sudden changes which react to demand on the grid system were as short as 10 minutes (1 sample), but longer changes during re-fuelling can take as long as 24 hours (144 samples). Therefore the decrease in rotor current would require a maximum time period of 1 day (144 samples).

The learning module located the key behaviour from the data, but there were additional features which had to be specified by the expert before an adequate heuristic could be derived. One of these features was that the vibration level would have to be high in the case of stiction. This high level of vibration should be maintained for the full duration of no change in vibration. This aspect is not specified in the derived heuristic since the observable states, no change in the overall and 1st order magnitudes, are derived purely based on the previous state in the explanation, no change in vibration. When deriving these observable states there is no memory of the fact that the fault is stiction. Instead the observable states are only derived from the fact that there is no change in the vibration magnitude. This highlights two aspects of the learning module approach. The first is that the module is unable to exhibit memory of the state of the system beyond a single causal step. The second is the semi-autonomous operation of the learning module since the module's primary goal is to assist the expert in deriving the fault diagnostic knowledge, not to derive a complete heuristic on its own.

Once the expert had completed her analysis and put forward suggestions for both the temporal constraints and the additional features which were to be included in the final heuristic, the Expert System maintenance engineers were able to translate the results into a heuristic which would be suitable for the Expert System knowledge base. The profile for each signal is derived by compiling a description of the raw data over a 1 day period using the approach described in section 4.5.3.1. This period was chosen during the development and testing of the Expert System in conjunction with the experts. Various periods were experimented with and output to the user interface as demonstrated in section 4.5.6.3. The experts felt that the channel profile derived by setting the period length to 1 day gave a good representation of each of the signals. Therefore the periods of each feature contained in the heuristic would have to be set to a minimum of 144 samples. This means that the period of no change would move from greater than or equal to 6 samples (1 hour) to 144 samples and the rotor current decrease period would be set to 144 samples which was the maximum period specified by the expert.

This heuristic is shown in figure 5.26. The heuristic can be summarised as when the rotor current decreases and both of the generator bearings have no reduction in the overall and 1st order vibration magnitude, and the magnitude is at a high level then the turbine generator is showing signs of stiction behaviour. Note from the heuristic that all of the vibration magnitudes should have been at a high level before the change in rotor current occurs and that the vibration behaviour should not change

Fault: Stiction

Symptoms: GRC-OA Change-Dec (t)(144)
GBR-1M Level-NCh (<=t)(>= 144)
GBR-1M Level-High (<=t)(>= 144)
GBR-OA Level-NCh (<=t)(>= 144)
GBR-OA Level-High (<=t)(>= 144)
GBF-1M Level-NCh (<=t)(>= 144)
GBF-1M Level-High (<=t)(>= 144)
GBF-OA Level-NCh (<=t)(>= 144)
GBF-OA Level-High (<=t)(>= 144)

Figure 5.26: Stiction fault heuristic suitable for the Expert System knowledge base.

5.6.3 Looseness Training Example

The data selected by the British Energy condition monitoring expert as an example of looseness fault data was from 14/08/2006 till 29/08/2006. The overall, 1st order magnitude and phase, 2nd order magnitude and phase, and rpm data for bearings 6, 7, 8, 9 and 10, as well as the generator load, generator MVArs, generator rotor current and temperature data were selected. All of this data can be viewed in graphical format in appendix E. The profile for the data was generated by the Expert System described in chapter 4 and can be found in appendix D. The explanation structure generated by the learning module is given in figure 5.27. All of the LUTs which were used to build this explanation can be found in appendix C.

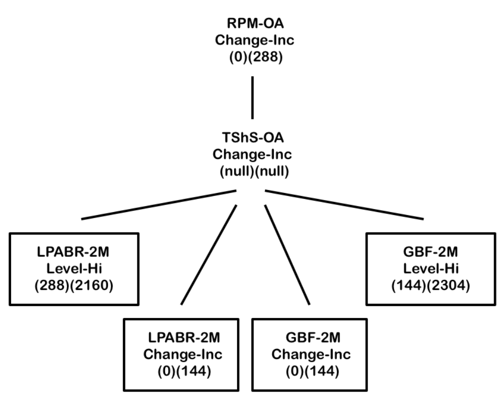


Figure 5.27: Explanation structure for Looseness fault generated by the learning module.

The behaviours which are not enclosed within a box denote normal behaviour whereas the ones enclosed within a box denote that they have been derived from looseness fault behaviour. The fault behaviour LUT used to generate this explanation is the Generator Front Pedestal Looseness LUT given in appendix C. This LUT implies that the behaviour exhibited by the channel is indicative of looseness in the pedestal at the front of the generator. The first bracketed number is the manifestation temporal value and the second is the period temporal value for that node. These values denote the number of samples where each sample represents 10 minute intervals. If there is no number in the brackets and reads "null" then this implies that the node was not observed in the channel profile data and there is therefore no temporal data associated with it. A description of each abbreviation in the looseness fault explanation chain is given in table 5.3

Table 5.3: Description of the abbreviations found in the looseness fault explanation in figure 5.27.

Abbreviations	Full Text
RPM	Rotations Per Minute
TShS	Turbine Shaft Speed
LPABR	Low Pressure A Bearing Rear
LPABRT	Low Pressure A Bearing Rear Temperature
GBF	Generator Bearing Front
GBFT	Generator Bearing Front Temperature
2M	2nd Order Magnitude
OA	Overall Amplitude
Inc	Increase
Hi	High

The explanation generated a branch which exhibits behaviour consistent with a looseness fault in the front pedestal of the generator. This is implied in the firing of the Generator Front Pedestal Looseness LUT given in appendix C. This is the pedestal which supports both the low pressure B rear bearing and the generator front bearing. An increase in the turbine rotational speed following the system outage causes an increase in the turbine shaft speed. The looseness present in the pedestal which supports both the low pressure B rear bearing and the generator front bearing causes the 2nd order vibration magnitude in both bearings to increase and move to a high level. Both of the increases and high levels were picked up by the learning module. The expert verified that this explanation had highlighted the main characteristics of the looseness fault behaviour and so the next stage was for the temporal constraints of the explanation structure to be generalised. The results of which are given in figure 5.28.

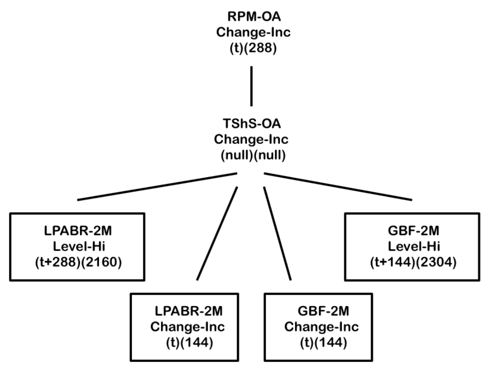


Figure 5.28: Looseness explanation structure following the generalisation of the instantiation temporal constraints.

Following the generalisation of the temporal constraints a heuristic was compiled. The results of the heuristic composition are shown in figure 5.29.

Fault:	Looseness
Symptoms:	RPM-OA Change-Inc (t)(288) LPABR-2M Level-Hi (t+288)(2160) LPABR-2M Change-Inc (t)(144) GBF-2M Change-Inc (t)(144) GBF-2M Level-Hi (t+144)(2304)

Figure 5.29: Heuristic derived by the learning module a Looseness fault.

This heuristic was particular to the training example used to induce the explanation therefore the period temporal constraints would have to be generalised to identify other looseness type faults. On inspection of the heuristic the expert confirmed that the increase in 2nd order vibration should occur in line with the RPM increase as had been highlighted in the derived heuristic. The expert stipulated that the increase in 2nd order vibration would not continue for a longer period than the increase in RPM. Therefore the period for both the increase in RPM and 2nd order vibration was set to the same value which was less than or equal to 288 samples. The expert determined that the 2^{nd} order vibration should fall within a high level within these increases therefore the instantiation constraint for the 2^{nd} order vibration high states should be set to <= t + 288. The expert also felt that the high level of vibration would last for as long as the turbine generator set was kept on-line. This period could vary dramatically but the expert felt that is was safe to assume that the set would be on for at least a day in the worst case scenario. Therefore the high 2^{nd} order vibration level period constraint was set to >= 144.

The learning module had identified the key characteristics associated with a looseness fault. Features not identified by the module were those which were able to distinguish a looseness fault from other behaviour types that showed similar characteristics. The expert felt that in particular looseness has similar characteristics to a shaft crack and misalignment fault. The feature which distinguishes looseness from a shaft crack fault is that a cracked shaft will normally develop on-line but looseness occurs following an outage. Since the rpm should remain constant at 3000rpm while on-line, the only time an increase in rpm realistically occurs is following an outage, therefore it was felt that the increase in rpm feature adequately distinguished between both of these behaviours. Misalignment also generates similar features to looseness, except that misalignment is expected to give an increase in the temperature of the affected bearings since the lubricating oil dissipates more energy caused by the extra loading on the bearing. Therefore the expert felt that the heuristic should explicitly state that there should be no increase within the temperature of the affected bearings while the RPM is increasing. This highlights one aspect of the learning module which is that it is currently unable to identify negative literals within the heuristic which can help distinguish between fault types of similar characteristics.

Once the expert had completed her analysis and put forward suggestions for both the temporal constraints and the additional features which were to be included in the final heuristic, the Expert System maintenance engineers were able to translate the results into a heuristic suitable for the knowledge base. This heuristic is given in figure 5.30. The heuristic can be summarised as following an outage, if a high level of 2nd order vibration is detected on the low pressure B rear bearing and the generator front bearing and there is no increase in bearing temperature then the turbine generator is exhibiting the behaviour of a looseness fault on the front generator pedestal.

Fault: Looseness

Symptoms: RPM-OA Change-Inc (t)(<= 288)
LPABR-2M Level-Hi (<= t+288)(>= 144)
LPABR-2M Change-Inc (t)(<= 288)
-LPABRT-OA Change-Inc (t)(<= 288)
-GBFT-OA Change-Inc (t)(<= 288)
GBF-2M Change-Inc (t)(<= 288)
GBF-2M Level-Hi (<= t+288)(>= 144)

Figure 5.30: Looseness fault heuristic which could be uploaded to the Expert System knowledge base.

5.7 Evaluation

The previous section has shown how the learning module performed in assisting the expert in deriving heuristics using actual condition monitoring data taken from the Beran system. The module was able to use the causal fault and behavioural knowledge to automatically distinguish between features which were relevant to the fault type under analysis to those that were not. Both the causal explanations and the resulting derived heuristic produced by the learning module assisted the expert in producing the final verified heuristic for the Expert System. The explanation allowed the expert to follow the rationale produced by the system to determine that the chain of events derived from the data agreed with their assessment. The explanation also provided an opportunity for the module to highlight certain features relevant to the behaviour under analysis which the expert may not have otherwise picked up on. The heuristic derived from the explanation allowed the expert to further refine the expression so that it was made more representative of other examples of that fault type and so that it could distinguish between different faults, which exhibit similar characteristics. In both cases, the learning module produced accurate explanations of a sufficient level of detail and the expert was able to expand on these to give a fully verified heuristic.

The intention for the learning module is for it to be used as a standalone program which can interactively work with the expert user on his/her own to help derive heuristic expressions. However the two tests outlined in this chapter involved both the expert and the knowledge engineer using the learning module results to derive the final heuristic. The knowledge engineer was still required to ask probing questions which would get the expert thinking on alternative faults which may show similar characteristics to determine what other features would have to be included in the expression. As already pointed out in the previous section these additional features normally identify events which should not take place and are known as negative literals. One way this role could be achieved automatically by the learning module would be to try and prove all other fault types and indicate the ones which would have produced an explanation with the same features as the fault under analysis but required additional features to complete the proof of the explanation. These additional features could be used as negative literals in the heuristic and could therefore be automatically posed as a question to the expert user. For example, if an algorithm existed for the learning module which tested alternative theories then it may have indicated that misalignment shared the same features as looseness plus additional temperature features which could not be validated against the data used in the example. The absence of the temperature behaviour expected for a looseness fault indicates that this could potentially be used as a negative literal in the case of looseness and could therefore be posed as a question to the expert. This question was posed to the expert during the testing of the learning module but through the knowledge engineer, not by the learning module automatically. Another algorithm would have to be developed for the learning module to achieve this functionality. The algorithm would have to be capable of testing all known fault types in the causal fault and behavioural knowledge base against the data. It would have to recognise instances which contain the same events as the suspected fault type but with additional events which can not be verified by the training data. This would be part of the future worked planned for this research.

Another aspect of the approach adopted by the learning module highlighted during testing was that no states are propagated through the causal explanations. In other words there is no memory of state in the causal models. This feature was highlighted in the stiction fault example where the high level of vibration included within the final verified heuristic was not picked up on by the learning module. The reason for this was that the stiction fault behaviour was picked up in the middle of the explanation structure so when it came to determining the observables, the module was only able to infer an increase in both overall and 1st order vibration because there was no memory of the stiction fault. This feature is deliberate in the design because the alternative would be to specify the effect that an increase in vibration would have under a stiction fault, meaning that the fault behaviour would have to be specified multiple times in the LUTs. This would, therefore, defeat one of the initial objectives for the causal modelling approach adopted which is to indicate the effect that the fault behaviour has at one particular point and not propagate the behaviour through to see the effects. This is the job of the learning module. The expert will therefore always be required to consider any special requirements for the level states of any parameters. Again the learning module could be programmed to pose this as a standard question to the expert user to eliminate the need for the knowledge engineer.

The testing of the learning module on genuine condition monitoring data has proven that the novel adapted version of EBG developed for this project can automatically derive knowledge to assist the expert in deriving heuristics. This is one of the primary objectives of the learning module outlined in section 5.2. However it has also been indicated that further research is required to enhance the functionality of the module so that the type of assistance currently provided by the knowledge engineer could be automatically facilitated. The module must also be assessed against the second objective outlined in section 5.2, which is to reduce the amount of knowledge elicitation required over the course of the project. A thorough assessment of this is clearly not possible until the module is developed further and used throughout the project. Therefore, the only comment which can be made on this issue at this stage is that the modular approach adopted by the learning module to construct the causal fault and behaviour knowledge should enable the reusability of knowledge over different configurations of turbine generator sets, eliminating the need to perform separate knowledge elicitation exercises for every unique item. This reusability is expected to provide savings on the amount of knowledge elicitation undertaken throughout the course of the project as discussed in section 5.4.

Chapter 6

6 Conclusions & Further Work

6.1 Conclusions

This body of research has reported on work carried out in the area of automated condition monitoring for turbine generators. The primary objective of this work was to develop tools which could assist British Energy experts in the analysis of large volumes of data produced by their on-line condition monitoring system. The chosen approach was to develop an Expert System which employed explicit expert knowledge to interpret the raw data in order to diagnose common behaviours. A major drawback of the Expert System approach encountered during the development of the system was the time consuming and expensive nature of the knowledge engineering exercises required to capture the necessary knowledge. Novel methods of assisting with the knowledge engineering process were researched. This resulted in the development of a novel semi-autonomous learning module which used causal knowledge of the turbine generator behaviour along with examples of a certain fault type to assist the expert and knowledge engineer in the derivation of heuristic rules which could be utilised by the Expert System.

The strategically important nature of the application dictated that any system which was to assist the British Energy experts had to be designed in such a way that the user could be confident in the given assessments. This was achieved by focussing on providing transparency in the assessments performed by the system through the application of the explicit symbolic knowledge employed by the Expert System. Key features identified by the experts at the knowledge elicitation stage and therefore extracted by the signal to symbol transformation module were fed back to the user at the verification stage by visually highlighting them in the raw data plots. In addition, each of the diagnostic rules were given explanatory descriptions so that the chaining of the rules fired during the assessment provided the user with the system rationale. Both of these explanation facilities were used to complement one another to provide the user with a more complete description on how the system arrived at its conclusions.

The system was tested on genuine historical case studies selected by the experts from the alarm archive. The results of the tests when compared with the previous manual assessments and by the current team of experts indicated that the system was able to perform accurately in its analysis. In some cases inconsistencies in previous assessments were identified indicating that the automated system has the potential to make the assessment process more accurate when used in conjunction with the expert. The testing also highlighted the benefits of the novel approach to explanation integrated into the system. The experts were able to track the system reasoning using the rationale trees and identify the features used in the assessment within the raw data plots. The results obtained during the testing phase combined with the positive expert response to the explanation facilities highlighted that the Expert System approach was well suited to this strategically important application. One of the common drawbacks associated with this approach did identify itself during the testing. This was conflict resolution where the system provides multiple conclusions since the data is consistent with more than one behaviour type. This was less of a problem in this application since the system is designed to act as an assistant therefore multiple assessments allowed the user to explore all avenues before confirming the actual behaviour. However it was highlighted that the instances of conflict resolution encountered during the testing could be rectified in future versions of the system by updating the knowledge base with additional knowledge.

The work carried out on the Expert System identified the difficulty with acquiring the expert knowledge and the growing need for new methods that can assist with the knowledge capture process. A learning module was developed based on the Machine Learning (ML) technique Explanation Based Generalisation (EBG) which was able to assist the expert in deriving knowledge. The novel approach uses causal turbine generator behavioural models in addition to a training example to derive heuristic rules associated with that behaviour type. Explanations are generated which are consistent with the behaviour of interest and the training example. These explanations can then be verified by the expert before being transformed into suggested heuristic rules. The expert and knowledge engineer can then refine the solution to determine a suitable heuristic for the Expert System knowledge base. The module was tested using genuine faults selected by the experts from the Beran system. The module performed well, each time producing valid explanations and therefore identifying key features which were developed into a suitable heuristic for the knowledge base.

In terms of novelty this body of research has demonstrated through the design and testing of both the Expert System and learning module the three following contributions.

- **Augmentation of the existing condition monitoring approach through the introduction of intelligent automated processing.**
  The application of the Expert System developed for this project is novel. The system had to augment a well established approach to turbine generator condition monitoring within British energy. It had to be designed to interface with the existing condition monitoring system and provide the British Energy experts with the well defined information required for them to perform their assessment.

- **Novel use of graphical approaches to provide explanation of Expert System rationale.**
  Central to the ethos of the Expert System design was the effectiveness of how the assessment explanation was fed back to the user. In addition to the use of rule-based explanation, novel approaches to graphically highlight features used within the assessment have been incorporated into the developed prototype to assist the user during verification.

- **Novel semi-autonomous approach to diagnostic condition monitoring knowledge derivation.**

The learning module developed to assist in deriving explicit heuristic knowledge for the Expert System is novel in both its approach and application. The learning module designed and developed uses an adapted approach of EBG that utilises causal fault and behavioural models. The module aims to reduce the burden associated with capturing knowledge for Expert System applications.

6.2 Further Work

The next stage in the development of the Expert System is to expand the knowledge base so that it is capable of identifying more behaviours. Any further knowledge elicitation exercises should make use of the learning module in order to determine with more certainty its ability to reduce the knowledge elicitation effort. British Energy has requested that the system is developed further so that it can be used by the experts as an on-line assistant. This will require the module to automatically interpret the FFT data, therefore methods of capturing the raw FFT data must be investigated and there will have to be more development in the signal to symbol transformation module so that the correct features are extracted from the raw data. The system will also have to be integrated with the existing British Energy on-line network so that all of the analysis can be carried out automatically for the experts to log on and check the results. The explanation facility should also be developed further so that both the rationale and the visual indication of the features are enhanced. This can be achieved by linking both, so that when the user selects one of the triggered rules from the explanation structure, he/she is then directed to the features within the raw data which triggered the rule.

The learning module needs to be tested further and more rigorously assessed against time saved during the knowledge elicitation process. As stated in the objectives for the learning module, the usefulness and success of the module will ultimately be dependent on whether the knowledge engineering approach is made easier. The simplicity of the algorithmic approach should be maintained since this allows the process of capturing the causal behaviour knowledge more structured and less device specific. One improvement however would be to incorporate negative literals into the assessment and therefore include them in both the explanations and the derived heuristics. As demonstrated during the testing, negative literals play a major role in differentiating certain behaviour types from one another. There does need to be a lot of development on the user interface itself so that the module reaches a stage whereby the system can be used on its own without the need for the system developer to operate it and interpret the results. One function which would go some way to achieving this would be for the module to automatically ask the user some generic questions on the module results which a knowledge engineer would commonly ask the expert in order to explore all avenues of that particular area.

There should also be some consideration given to extending the Expert System and learning module beyond the turbine generator condition monitoring application. Applications, which should be considered for the application of the learning module, are those that are strategically important or even safety critical to the business. It is this type of application which will normally require a high level of user confidence in the system due to its importance to the business, making the symbolic nature of the knowledge derived by the learning module desirable. The application must also have access to expert knowledge in some form so that the heuristic knowledge, or causal models, required by the Expert System, or learning module respectively, are available or can be developed. For example, an application within British Energy, which may be suitable for extending the application of the Expert System or learning module, would be cooling water motor/pump fault diagnosis.

Identifying additional applications and developing both the Expert System and learning module is especially pertinent for British Energy with the possibility of new nuclear power stations being built in the coming decades. If suitable applications can be identified prior to the construction of the power stations, then systems can be developed before the power stations come on-line. A potential benefit of developing the systems in parallel with the construction of the nuclear plants is that the commissioning of the equipment can provide an opportunity to test the performance of the Expert System and learning module. It is likely that sate-of-the-art equipment would be used in the construction of new generation nuclear power stations, therefore, the current team of condition monitoring experts will not have the years of experience and knowledge on how the equipment behaves under normal and faulty conditions. This would mean that any Expert System, or learning module, developed prior to the start-up of the nuclear plants, would rely heavily on the experience of the equipment manufacturers, or alternatively, users who may have acquired a certain level of experience from using the equipment at alternative locations.

References

[Awadallah & Morcos, 2003] M. A. Awadallah, M. M. Morcos, "Application of AI Tools in Fault Diagnosis of Electrical Machines and Drives – An Overview," IEEE Transactions on Energy Conversion, Vol. 18, No. 2, 2003, pp. 245-251.

[Baum et al, 1970] L. Baum, T. Petrie, G. Soules, N. Weiss, "A Maximisation Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," Annals of Mathematical Statistics, Vol. 41, No. 1, 1970, pp. 164-171.

[Barr & Feigenbaum, 1981] A. Barr and E. A. Feigenbaum, The Handbook of Artificial Intelligence, Vol. 1, Los Altos CA: Morgan Kaufmann, 1981.

[Booth & McDonald, 1998] C. Booth, J. R. McDonald, "The use of artificial neural networks for condition monitoring of electrical power transformers," Neurocomputing, Vol. 23, 1998, pp. 79-109.

[Brown et al, 2007] A. J. Brown, V. M. Catterson, M. Fox, D. Long, S. D. J. McArthur, "Learning Models of Plant Behaviour for Anomoly Detection and Condition Monitoring," International Conference on Intelligent Systems Applications to Power Systems, 2007, ISAP 2007, 5-8 November 2007, pp. 1-6.

[Cauvin et al, 1998] S. Cauvin, M. O. Cordier, C. Dousson, P. Laborie, F. Levy, J. Montmain, M. Porcheron, I. Servet, L. Trave-Massuyes, "Monitoring and Alarm Interpretation in Industrial Environments," AI Communications, Vol. 11, No. 3, 1998, pp. 139-173.

[Chien et al, 2002] C. F. Chien, S. L. Chen, Y. S. Lin, "Using Bayesian Network for Fault Location on Distribution Feeder," IEEE Transactions on Power Delivery, Vol. 17, No. 13, 2002, pp. 785-793.

[CIGRE, 2003] "Life Management Techniques for Power Transformers," CIGRE WG A2.18, 20 January 2003.

[Clocksin & Mellish, 1984] W. F. Clocksin and C. S. Mellish, Programming in PROLOG, 2nd Edition, Springer-Verlag. 1984.

[Console & Torasso, 1988] L. Console, P. Torasso, "Heuristic and Causal Reasoning in CHECK," Proc. 12th IMACS World Conference on Scientific Computation 88, Paris, July 1988, pp. 283-286.

[Console et al, 1989] L. Console, D.T. Dupre, and P. Torasso, "A Theory of Diagnosis for Incomplete Causal Models," Proc. llth. IJCAI Morgan Kaufmann, San Francisco, Calif., 1989, pp. 1311-1317.

[Culbert & Rohdes, 2007] I. M. Culbert, W. Rohdes, "Using Current Signature Analysis Technology to Reliably Detect Cage Winding Defects in Squirrel Cage Induction Motors," IEEE Transaction on Industry Applications, Vol. 43, No. 2, March/April 2007, pp. 422-428.

[Davidson et al, 2003] E. M. Davidson, S. D. J. McArthur, J. R. McDonald, "A Tool-set for Applying Model-based Reasoning Techniques to Diagnostics for Power System Protection," IEEE Transactions on Power Systems, Vol. 18, Issue 2, May 2003, pp. 680-687.

[Davidson et al, 2005] E. M. Davidson, S. D. J. McArthur, J. R. McDonald, "Model-based Reasoning Techniques for the Analysis of Protection Operation," International Journal of Engineering Intelligent Systems for Electrical Engineering and Communications, Vol. 13, No. 2, 2005, pp. 97-104.

[Davis & Hamscher, 1988] A. Davis and W. Hamscher, "Model-based reasoning: Troubleshooting" In Exploring Artificial Intelligence: Survey Talks from the National Conferences on Artificial Intelligence, Morgan Kaufmann, 1988, pp. 297-346.

[DeJong & Mooney, 1986] G. DeJong, R. Mooney, "Explanation-Based Learning: An Alternative View," Machine Learning, Vol. 1, No. 2, 1986, pp. 145-176.

[Derere, 2000] L. Derere, "Case-Based Reasoning: Diagnosing Of Faults In Complex Systems Through Reuse Of Experience," ITC International Test Conference, 2000, pp. 27-34.

[Embrechts & Benedek, 2004] M. J. Embrechts, S. Benedek, "Hybrid Identification of Nuclear Power Plant Transients With Artificial Neural Networks," IEEE Transactions on Industrial Electronics, Vol. 51, No. 3, 2004, pp. 686-693.

[Fenton et al, 2001] W. G. Fenton, T. M. McGinnity, L. P. Maguire, "Fault Diagnosis of Electronic Systems Using Intelligent Techniques: A Review," IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews, Vol. 31, No. 3, 2001, pp. 269-281.

[Forney, 1973] G. D. Forney, "The Viterbi Algorithm," Proceedings of the IEEE, Vol 61, No. 3, 1973, pp 268-278.

[Gemmell, 1995] B. D. Gemmell, "A Consultative Expert System for Intelligent Diagnosis on Steam Turbine Plant," Ph.D. Thesis, University of Strathclyde, UK, 1995.

[Gonzalez et al, 1986] A. J. Gonzalez, R. L. Osborne, C. T. Kemper, "On-Line Diagnosis of Turbine-Generators Using Artificial Intelligence," IEEE Transactions on Energy Conversion, Vol. 1, No. 2, June 1986, pp 68-74.

[Haji & Toliyat, 2001] M. Haji, H. A. Toliyat, “Pattern Recognition – A Technique for Induction Machines Rotor Broken Bar Detection,” IEEE Transactions on Energy Conversion, Vol. 16, No. 4, 2001, pp 312-317.

[Hamscher et al, 1992] W. Hamscher, L. Console, J. de Kleer, “Readings in Model-Based Diagnosis,” Morgan Kaufmann, 1992.

[Han & Song, 2003a] Y. Han, Y. H. Song, “Condition Monitoring Techniques for Electrical Equipment – A Literature Survey,” IEEE Transactions on Power Delivery, Vol. 18, No. 1, 2003, pp. 4-13.

[Haykin, 1999] S. S. Haykin, Neural Networks: a comprehensive foundation, 2nd Edition, Prentice Hall, 1999.

[Jackson, 1999] P. Jackson, Introduction to Expert Systems, Third Edition, Addison Wesley Longman, 1999.

[Judd et al, 2002] M. D. Judd, S. D. J. McArthur, J. R. McDonald, O. Farish, “Intelligent Condition Monitoring and Asset Management: Partial Discharge Monitoring for Power Transformers,” Power Engineering Journal, Vol. 16, No. 6, Dec. 2002, pp. 297-304.

[Kobayashi & Nakamura, 1991] S. Kobayashi, K. Nakamura, “Knowledge Compilation and Refinement for Fault Diagnosis,” IEEE Expert, Vol. 6, No. 5, 1991, pp. 39-46.

[Kolodner, 1993] J. L. Kolodner, Case-Based Reasoning, San Mateo, CA: Morgan Kaufmann, 1993.

[Li et al, 2000] B. Li, M. Y. Chow, Y. Tipsuwan, J. C. Hung, "Neural-Network-Based Motor Rolling Bearing Fault Diagnosis," IEEE Transactions on Industrial Electronics, Vol. 47, No. 5, 2000, pp. 1060-1069.

[Luger & Stubblefield, 1998] G. F. Luger and W. A. Stubblefield, Artificial Intelligence: Structures and Strategies for Complex Problem Solving, Third Edition Addison Wesley Longman, 1998.

[Lynn & Fuerst, 1998] P. A. Lynn, W. Fuerst, Introductory Digital Signal Processing with computer applications, 2nd Edition, Wiley, 1998.

[McArthur et al, 2004] S. D. J. McArthur, S. M. Strachan and G. Jahn. "The Design of a Multi-Agent Transformer Condition Monitoring System", IEEE Transactions on Power Systems, Vol. 19, No. 4, 2004, pp. 1845-1852.

[McCulloch & Pitts, 1943] W.S. McCulloch & W. Pitts, "A logical calculus of the ideas immanent in nervous activity," Bulletin of Mathematical Biophysics, Vol. 5, 1943, pp. 115-133.

[Mejia-Lavelle & Rodriguez-Oritz, 1998] M. Mejia-Lavelle, G. Rodriguez-Oritz, "Obtaining Expert System Rules Using Data Mining Tools from a Power Generation Database," Expert Systems with Applications, Vol. 14, No. 1, 1998, pp. 37-42.

[Milne et al, 2001] R. Milne, C. Nicol, L. Trave-Massuyes, "TIGER with model based diagnosis: initial deployment," Knowledge-Based Systems, Vol. 14, No. 3-4, June 2001, pp. 213-222.

[Minakawa et al, 1995] T. Minakawa, M. Kunugi, K. Shimada, N. Wada, M. Utsunomiya, "Development and Implementation of a Power System Fault Diagnosis

Expert System," IEEE Transactions on Power Systems, Vol. 10, No. 2, May 1995, pp. 932-940.

[Mitchell, 1982] T. M. Mitchell, "Generalization as Search," Artificial Intelligence, Vol. 18, No. 2, 1982, pp. 203-226.

[Mitchell et al, 1986] T. M. Mitchell, R. M. Keller, S. T. Kedar-Cabelli, "Explanation-Based Generalization: A Unifying View," Machine Learning, Vol. 1, No. 1, 1986, pp. 47-80.

[Mitchell, 1997] T. M. Mitchell, Machine Learning, McGraw-Hill, 1997.

[Nakamura et al, 2007] H. Nakamura, Y. Yamamoto, Y. Mizuno, "Diagnosis of Short Circuit Fault of Induction Motor Based on Hidden Markov Model," Conference on Electrical Insulation and Dielectric Phenomena, CEIDP 2007, 2007, pp. 61-64,

[Newell et al, 1963] A. Newell, J. C. Shaw, H. A. Simon, "Empirical explorations with the Logic Theory Machine: a case study in heuristics," In Feigenbaum and Feldman, Computers and though: A collection of articles, McGraw-Hill, 1963.

[Newell & Simon, 1963] A. Newell and H. Simon, "GPS: a program that simulates human thought," In Feigenbaum and Feldman, Computers and though: A collection of articles, McGraw-Hill, 1963.

[Newell & Simon, 1972] A. Newell and H. Simon, Human Problem Solving, Prentice-Hall, 1972

[Ocak & Loparo, 2001] H. Ocak, K. A. Loparo, "A New Bearing Fault Detection and Diagnosis Scheme Based on Hidden Markov Modelling of Vibration Signals," IEEE

International Conference on Acoustics, Speech, and Signal Processing, ICASSP, Vol 5, 2001, pp. 3141-3144.

[Porter, 1986] Porter, B., "PROTOS: an experiment in knowledge acquisition for heuristic classification tasks", Proceedings In First International Meeting on Advances in Learning, Les Arcs, France, 1986, pp. 159-174.

[Protopapas et al, 1991] C. A. Protopapas, K. P. Psaltiras, A. V. Machias, "An Expert System for Substation Fault Diagnosis and Alarm Processing," IEEE Transactions on Power Delivery, Vol. 6, No. 2, April 1991, pp. 648-655.

[Quinlan, 1986] J. R. Quinlan, "Induction of decision trees," Machine Learning, Vol. 1, No. 1, 1986, pp. 81-106.

[Quinlan, 1993] J. R. Quinlan, C4.5: Programs for Machine Learning, San Mateo, Calif: Morgan Kaufmann, 1993.

[Rabiner, 1989] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proceedings of the IEEE, Vol. 77, Issue 2, 1989, pp. 257-286.

[Rosenblatt, 1958] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," Psychological Review, Vol 65, No. 6, 1958, pp. 386-408.

[Royo & Arcega, 2007] R. J. Royo, F. J. Arcega, "Machine Current Signature Analysis as a Way for Fault Detection in Squirrel Cage Wind Generators," IEEE International Symposium on Diagnostics for Electric Machines, Power Electronics and Drives, SDEMPED, 2007, pp. 383-387.

[Russell & Norvig, 1995] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, 1995.

[Schreiber et al, 2000] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. V. de Velde, B. Wielinga, Knowledge Engineering and Management: The CommonKADS Methodology, MIT Press, 2000.

[Shannon, 1948] C. E. Shannon, “A mathematical theory of communication,” Bell System Technical Journal, Vol. 27, 1948, pp. 379-423.

[Shortliffe, 1981] E. H. Shortliffe, A. C. Scott, M. B. Bischoff, W. van Melle and C. D. Jacobs, “ONCOCIN: an expert system for oncology protocol management,” In Proc. 7th International Joint Conference on Artificial Intelligence, 1981, pp. 876-881.

[Simon, 1983] H. A. Simon, “Why should machines learn?” In R. S. Michalski, J. G. Carbonell, T. M. Mitchell, Machine Learning: An Artificial Intelligence Approach, Los Altos CA: Morgan Kaufmann, 1983.

[Sonka et al, 1999] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis, and Machine Vision, PWS, 1999.

[Steele et al, 2003] J. A. Steele, L. A. Martin, S. D. J. McArthur, A. Moyes, J. R. McDonald, D. Howie, R. Elrick, I. Y. Yule, "An Intelligent System for Interpreting the Nuclear Refuelling Process within an AGR," Proceedings of the Institution of Mechanical Engineers Part A, Journal of Power and Energy, Vol. 217, No. 2, 2003, pp. 159-167.

[Steels, 1985] L. Steels, “Second Generation Expert Systems,” Future Generation Computer Systems, Vol. 1, Issue 4, 1985, pp. 213-221.

[Strachan et al, 2008] S. M. Strachan, S. Rudd, S. D. J. McArthur, M. D. Judd, Knowledge-Based Diagnosis of Partial Discharges in Power Transformers," IEEE Transactions on Dielectrics and Electrical Insulation, Vol. 15, No. 1, February 2008, pp. 259-268.

[Strachan et al, 2007] S. M. Strachan, S. D. J. McArthur, B. Stephen, J. R. McDonald, A. Campbell, "Providing Decision Support for Condition-Based Maintenance of Circuit Breakers Through Data Mining of Trip Coil Current Signatures," IEEE Transactions on Power Delivery, Vol. 22, No. 1, January 2007, pp. 178-186.

[Strachan, 2005] S. M. Strachan, "Decision Support System Development for Plant Condition Monitoring and Protection Design Reuse using Structured Methods of Knowledge Engineering and Knowledge Discovery," Ph.D. Thesis, University of Strathclyde, UK, 2005.

[Thomson & Fenger, 2001] W. T. Thomson, M. Fenger, "Current Signature Analysis to Detect Induction Motor Faults," IEEE Industry Applications Magazine, Vol. 7, Issue 4, July/August 2001, pp. 26-34.

[Trave-Massuyes & Milne, 1997] L. Trave-Massuyes, R. Milne, "Gas-Turbine Condition Monitoring Using Qualitative Model-Based Diagnosis," IEEE Expert, Vol. 12, No. 3, May-June 1997, pp. 22-31.

[Wang et al, 2000] Z. Wang, Y. Liu, P. J. Griffin, " Neural Net and Expert System Diagnose Transformer Faults," IEEE Computer Applications in Power, Vol. 13, No. 1, January 2000, pp. 50-55.

[Wang et al, 1998] Z. Wang, Y. Liu, P. J. Griffin, "A Combined ANN and Expert System Tool for Transformer Fault Diagnosis," IEEE Transactions on Power Delivery, Vol. 13, No. 4, October 1998, pp. 1224-1229.

[West et al, 2006] G. M. West, G. J. Jahn, S. D. J. McArthur, J. R. McDonald, J. Reed, "Data Mining Reactor Fuel Grab Load Trace Data to Support Nuclear Core Condition Monitoring," IEEE Transactions on Nuclear Science, Vol. 53, No. 3, June 2006, pp. 1494-1503.

[Wu & Chow, 2004] S. Wu, T. W. S. Chow, "Induction Machine Fault Detection Using SOM-Based RBF Neural Networks," IEEE Transactions on Industrial Electronics, Vol. 51, No, 1, February 2004, pp 183-194.

[Yan & Lanqin, 2006] W. Yan, G. Lanqin, "Bayesian Network Based Fault Section Estimation in Power Systems", TENCON IEEE Region 10 Conference, November 2006, pp. 1-4.

[Yongli et al, 2006] Z. Yongli, H. Limin, L. Jingling, "Bayesian Networks-Based Approach for Power Systems Fault Diagnosis," IEEE Transactions on Power Delivery, Vol. 21, No. 2, April 2006, pp. 634-639.

Appendix A. CommonKADS Knowledge Models

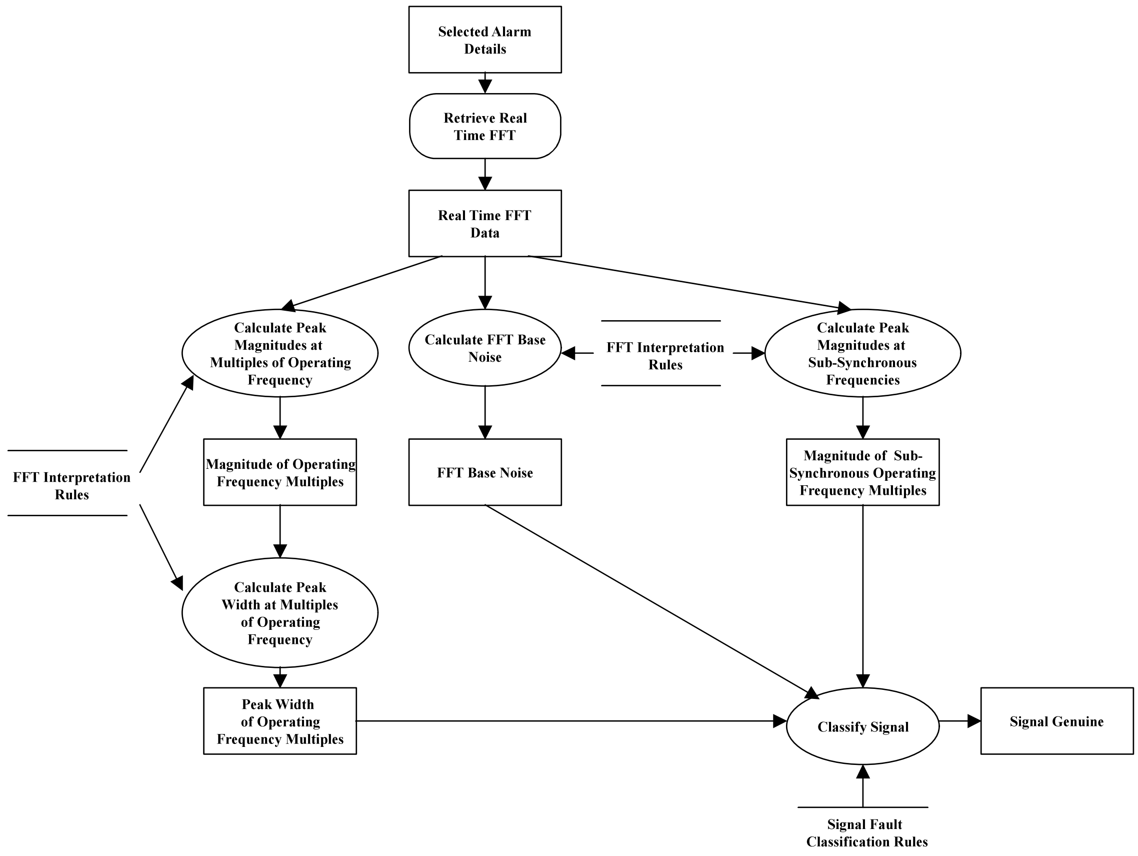


Figure A.1: Inference model for the determine if signal is genuine task

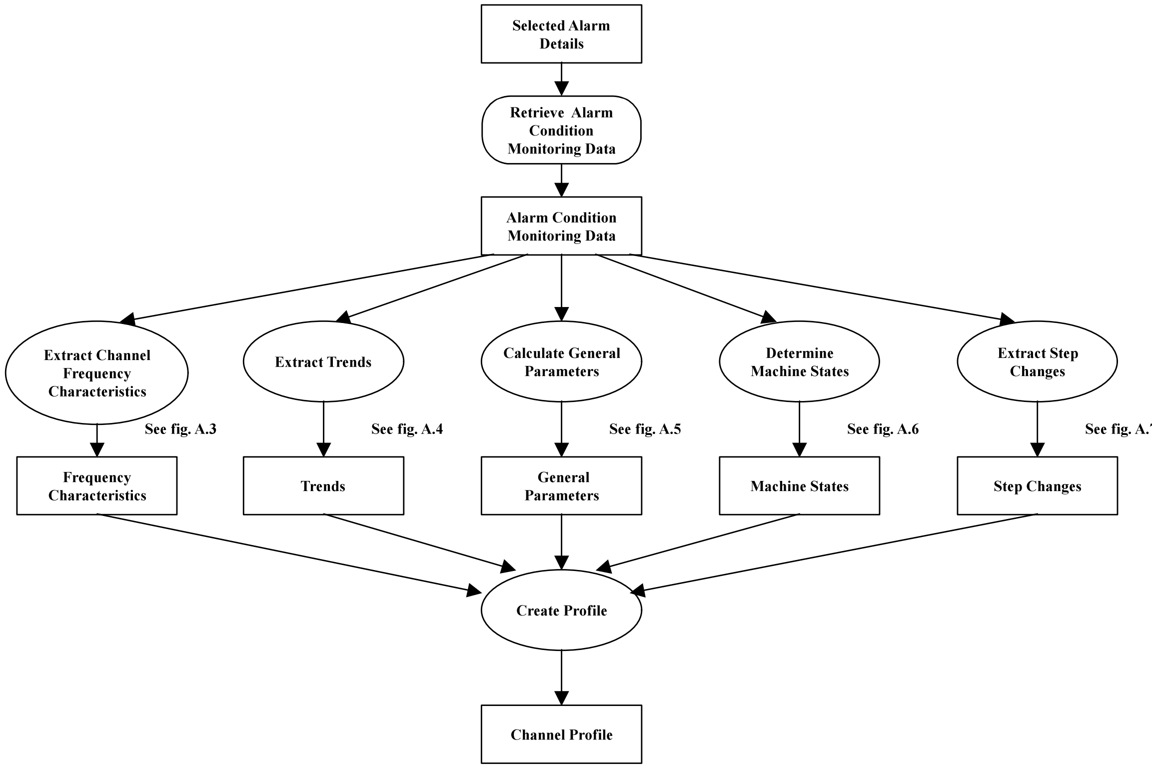


Figure A.2: Inference model for the build channel profile task

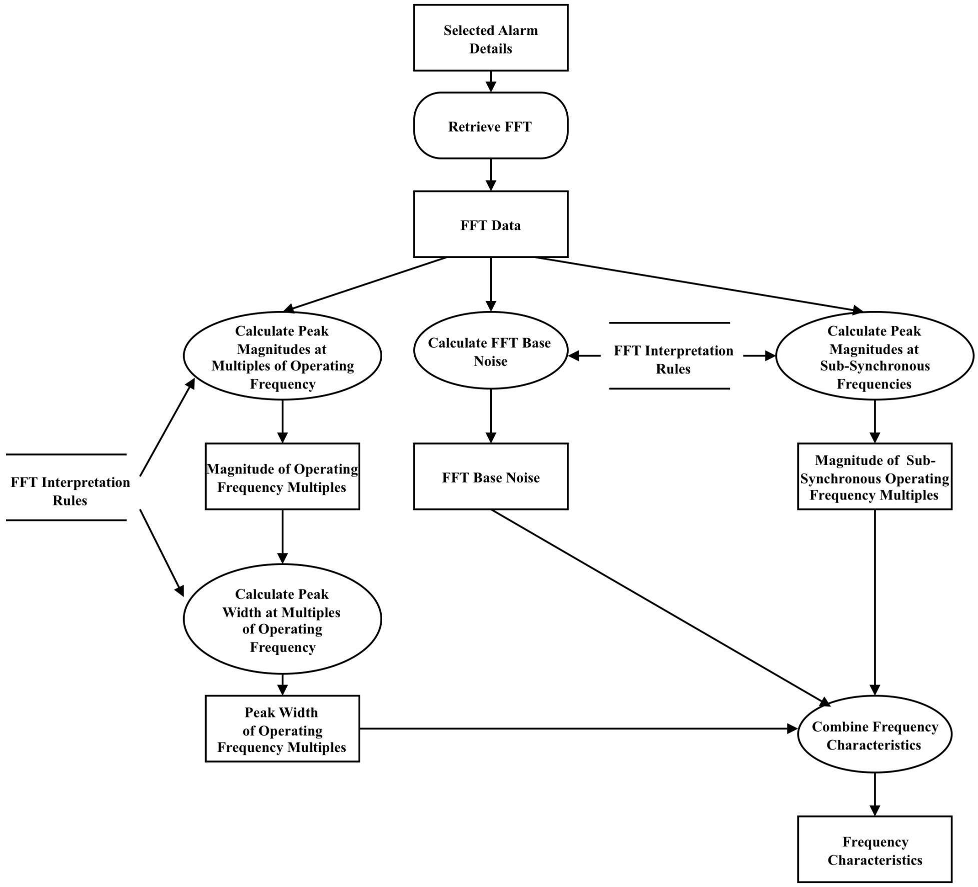


Figure A.3: Inference model for the extract channel frequency characteristics task

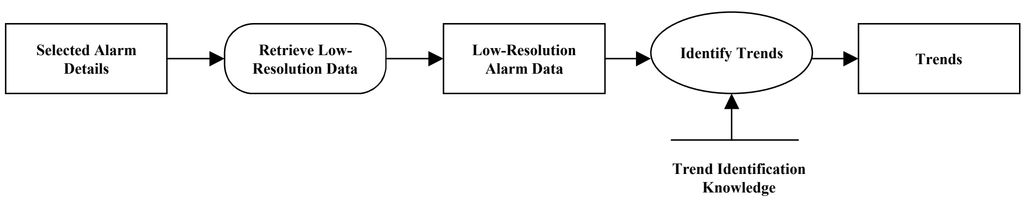


Figure A.4: Inference model for the extract trends task

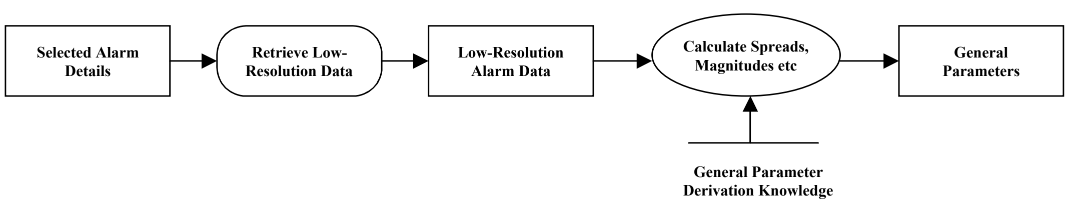


Figure A.5: Inference model for the calculate general parameters task

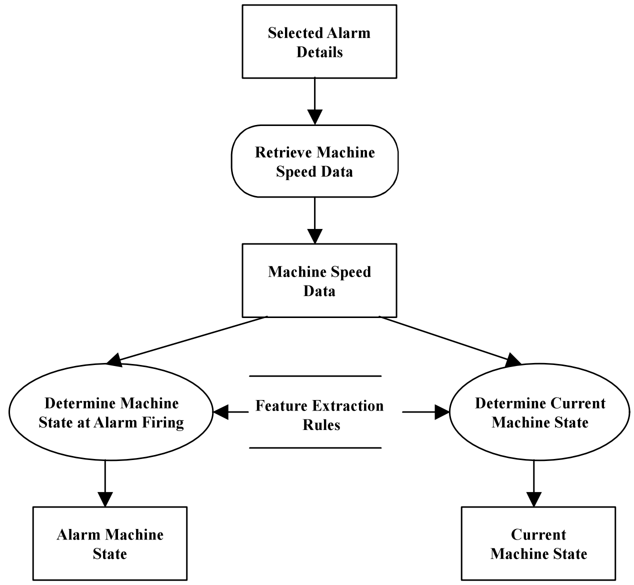


Figure A.6: Inference model for the determine machine state task

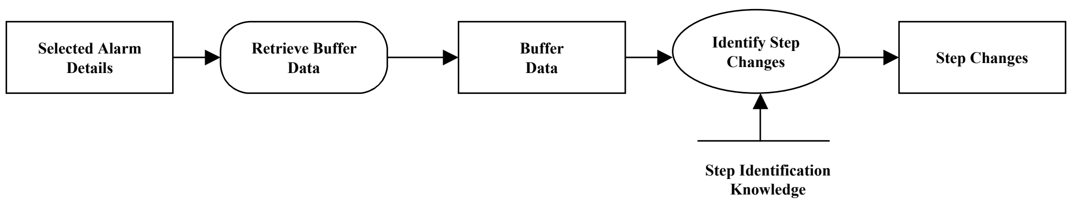


Figure A.7: Inference model for the extract step changes task

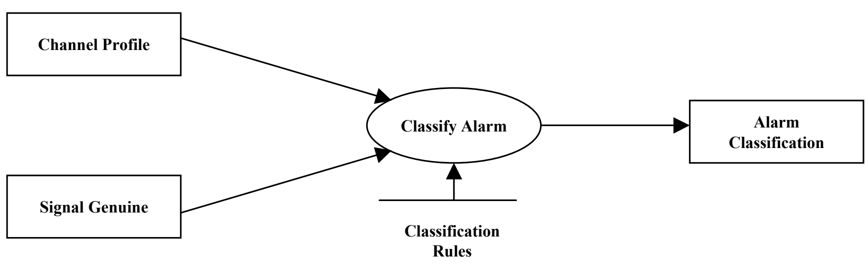


Figure A.8: Inference model for the classify alarm task

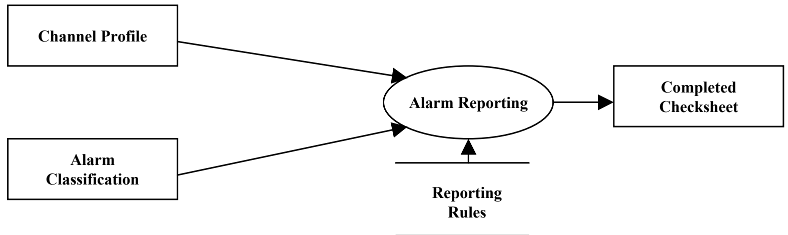


Figure A.9: Inference model for the alarm reporting task

Appendix B. Production Rules

Vibration Event Rules

Rules *file*1 to *file*27 are used to determine any events which occur in either of the 6 vibration files analysed by the Expert System. Table B.1 can be used for the substitution of *file* in each of the rules.

Table B.1: file abbreviations which can be inserted into the file rules

file	*file description*
OA	overall amplitude
1M	first order magnitude
1P	first order phase
2M	second order magnitude
2P	second order phase
SSA	sub-synchronous amplitude

For example "*file*1" can be substituted for "OA1," implying overall amplitude rule 1. Additioanly "*file* trend period > alarm period" can be substituted for "OA trend period > alarm period," implying the overall amplitude trend period occurs after the alarm period.

*file*1: Determine if high level in *file* within period after alarm
if *file* trend level = high
file trend period > alarm period
file trend period <= alarm period + defined period
then *file* high level after alarm

*file*2: Determine if low level in *file* within period after alarm
if *file* trend level = low
file trend period > alarm period
file trend period <= alarm period + defined period
then *file* low level after alarm

*file*3: Determine if low or high level not in *file* within period after alarm
if *file* trend level ≠ low
file trend level ≠ high
file trend period > alarm period
file trend period <= alarm period + defined period
then *file* not low or high level after alarm

*file*4: Determine if stable level in *file* within period before alarm
if *file* trend level = stable

file trend period < alarm period
file trend period >= alarm period - defined period
then *file* stable level before alarm

*file*5: Determine if no stable level in *file* within period before alarm
if *file* trend level ≠ stable
file trend period < alarm period
file trend period >= alarm period - defined period
then *file* no stable level before alarm

*file*6: Determine if stable level in *file* within period after alarm
if *file* trend level = stable
file trend period > alarm period
file trend period <= alarm period + defined period
then *file* stable level after alarm

*file*7: Determine if no stable level in *file* within period after alarm
if *file* trend level ≠ stable
file trend period > alarm period
file trend period <= alarm period + defined period
then *file* no stable level after alarm

*file*8: Determine if rising change in *file* within period after alarm
if *file* trend change = rising
file trend period > alarm period
file trend period <= alarm period + defined period
then *file* change after alarm

*file*9: Determine if decreasing change in *file* within period after alarm
if *file* trend change = decreasing
file trend period > alarm period
file trend period <= alarm period + defined period
then *file* change after alarm

*file*10: Determine if no change in *file* within period after alarm
if *file* trend change ≠ rising
file trend change ≠ decreasing
file trend period > alarm period
file trend period <= alarm period + defined period
then *file* no change after alarm

*file*11: Determine if stable change in *file* within period after alarm
if *file* trend change = stable
file trend period > alarm period

file trend period <= alarm period + defined period
then *file* stable change after alarm

*file*12: Determine if no stable change in *file* within period after alarm
if *file* trend change ≠ stable
file trend period > alarm period
file trend period <= alarm period + defined period
then *file* no stable change after alarm

*file*13: Determine if *file* is completely stable
if *file* trend change descriptors are stable = 100%
then *file* completely stable

*file*14: Determine if *file* is relatively stable
if *file* trend change descriptors are stable >= 70% & <100%
then *file* relatively stable

*file*15: Determine if file is unstable
if *file* trend change descriptors are stable <70%
then *file* unstable

*file*16: Determine if *file* is noisy
if *file* trend change descriptors are noisy >30%
then *file* noisy

*file*17: Determine if *file* is not noisy
if *file* trend change descriptors are noisy <=30%
then *file* not noisy

*file*18: Calculate amplitude if *file* has no stable level before alarm and high level after
if *file* no stable level before alarm
file high level after alarm
then calc *file* amplitude

*file*19: Calculate amplitude if *file* has stable level before alarm and high level after
if *file* stable level before alarm
file high level after alarm
then calc *file* amplitude
calc *file* amplitude high

*file*20: Calculate amplitude if *file* has no stable level before alarm and low level after
if *file* no stable level before alarm
file low level after alarm
then calc *file* amplitude

*file*21: Calculate amplitude if *file* has stable level before alarm and low level after
if *file* stable level before alarm
  file low level after alarm
then calc *file* amplitude
  calc *file* amplitude low

*file*22: Calculate amplitude if *file* is completely stable
if *file* completely stable
then calc *file* amplitude

*file*23: Calculate amplitude if *file* is relatively stable with no low or high level after alarm
if *file* completely stable
  file low level after alarm
  file high level after alarm
then calc *file* amplitude

*file*24: Calculate amplitude if *file* is unstable with no low or high level after alarm
if *file* unstable
  file low level after alarm
  file high level after alarm
then calc *file* amplitude

*file*25: Calculate min & max amplitude if *file* is noisy
if *file* noisy
then calc *file* amplitude min
  calc *file* amplitude max

*file*26: Determine if *file* recovers
if *file* stable level after alarm
then *file* recovers

*file*27: Determine if step occurred in *file*
if step in *file*
then *file* step occurred

Checksheet Rules

Rules CH1 to CH17 are used to determine the information required to complete the checksheet.

CH1: Operational change due to load change
if load trend before alarm
then load operational change

CH2: Operational change due to load step
if load step before alarm
then load operational change

CH3: Operational change due to rotor current change
if rotor current trend before alarm
then rotor current operational change

CH4: Operational change due to rotor current step
if rotor current step before alarm
then rotor current operational change

CH5: Operational change due to generator mvars change
if generator mvars trend before alarm
then generator mvars operational change

CH6: Operational change due to generator mvars step
if generator mvars step before alarm
then generator mvars operational change

CH7: Determine if overall amplitude is in zone 1 (HP/IP)
if HP/IP channel
overall amplitude < 42μm
then overall amplitude = zone 1

CH8: Determine if overall amplitude is in zone 2 (HP/IP)
if HP/IP channel
overall amplitude >= 42μm
overall amplitude < 63μm
then overall amplitude = zone 2

CH9: Determine if overall amplitude is in zone 3 (HP/IP)
if HP/IP channel
overall amplitude >= 63μm
overall amplitude < 100μm

then overall amplitude = zone 3

CH10: Determine if overall amplitude is in zone 4 (HP/IP)
if HP/IP channel
overall amplitude > 100μm
then overall amplitude = zone 4

CH11: Determine if overall amplitude is in zone 1 (LP/GEN)
if LP/GEN channel
overall amplitude < 63μm
then overall amplitude = zone 1

CH12: Determine if overall amplitude is in zone 2 (LP/GEN)
if LP/GEN channel
overall amplitude >= 63μm
overall amplitude < 100μm
then overall amplitude = zone 2

CH13: Determine if overall amplitude is in zone 3 (LP/GEN)
if LP/GEN channel
overall amplitude >= 100μm
overall amplitude < 163μm
then overall amplitude = zone 3

CH14: Determine if overall amplitude is in zone 4 (LP/GEN)
if LP/GEN channel
overall amplitude >= 163μm
overall amplitude < μm
then overall amplitude = zone 4

CH15: Determine if the addition of the 1st and 2nd order magnitude approximately equals the overall amplitude
if 1st order amplitude + 2nd order amplitude <= overall amplitude + tolerance
1st order amplitude + 2nd order amplitude >= overall amplitude - tolerance
then 1st order amplitude + 2nd order amplitude approximately equals overall amplitude

CH16: Determine if the addition of the 1st and 2nd order magnitude does not approximately equal the overall amplitude by being too large
if 1st order amplitude + 2nd order amplitude > overall amplitude + tolerance
then 1st order amplitude + 2nd order amplitude does not approximately equal overall amplitude

CH17: Determine if the addition of the 1st and 2nd order magnitude does not approximately equal the overall amplitude by being too small

if 1st order amplitude + 2nd order amplitude < overall amplitude - tolerance
then 1st order amplitude + 2nd order amplitude does not approximately equal overall amplitude

Operational Event Rules

Rules *op*1 to *op*6 are used to determine any events which occur in either of the 3 operational files analysed by the Expert System. Table B.2 can be used for the substitution of *op* in each of the rules.

Table B.2: operational file abbreviations which can be inserted into the operational rules

op	*op description*
L	load
RC	rotor current
MVA	generator MVArs

For example "*op*1" can be substituted for "RC1," implying rotor current rule 1. Additioanly "*op* trend period > alarm period" can be substituted for "RC trend period < alarm period," implying the rotor current trend period occurs before the alarm period.

*op*1: Determine if rising trend in *op* within period before alarm
if *op* trend change = rising
op trend period < alarm period
op trend period >= alarm period - defined period
then *op* trend before alarm

*op*2: Determine if decreasing trend in *op* within period before alarm
if *op* trend change = decreasing
op trend period < alarm period
op trend period >= alarm period - defined period
then *op* trend before alarm

*op*3: Determine if no trend in *op* within period before alarm
if *op* trend change ≠ decreasing
op trend change ≠ rising
op trend period < alarm period
op trend period >= alarm period - defined period
then *op* no trend before alarm

*op*4: Determine if rising step in *op* within period before alarm
if *op* step change = rising
op step period < alarm period
op step period >= alarm period - defined period
then *op* step before alarm

*op*5: Determine if decreasing step in *op* within period before alarm

if *op* step change = decreasing
op step period < alarm period
op step period >= alarm period - defined period
then *op* step before alarm

*op*6: Determine if no step in *op* within period before alarm
if *op* step change ≠ decreasing
op step change ≠ rising
op step period < alarm period
op step period >= alarm period - defined period
then *op* no step before alarm

Alarm Cause Rules

Rules AC1 to AC13 are used to determine the cause of the alarm. Table B.3 can be used for the substitution of *alarm* in each of the rules. In addition Table B.3 lists the signals associated with each alarm.

Table B.3: alarm abbreviations which can be inserted into the alarm cause rules

alarm	*alarm* description	Associated signals (*file*)
1X	first order phase	first order magnitude (1M)
		first order phase (1P)
2X	second order phase	second order magnitude(2M)
		second order phase (2P)
Z2, Z3, Z4	zone 2, zone 3 & zone 4	overall amplitude (OA)
SSH	sub-sync high	sub-sync amplitude (SSA)

For example "*op* change which affected *file* (*alarm*)" can be substituted for "RC change which affected 1M (1X)," implying a change in rotor current affected the first order magnitude which triggered the first order alarm. Note that in AC13 both the magnitude and phase files have to be included in the rule.

AC1: Electrical noise through frequency spikes
if spikes at operating frequency multiples
spikes narrow
then electrical noise through frequency spikes

AC2: Electrical noise through FFT base noise
if base noise in FFT
then electrical noise through FFT base noise

AC3: Zero sensor error reading
if machine state online
overall amplitude at alarm firing < zero sensor magnitude
then zero sensor error

AC4: Excessive sensor error reading
if overall amplitude at alarm firing > excessive sensor magnitude
then excessive sensor error

AC5: Loss of rpm sensor error reading
if rpm at alarm firing < loss of rpm value
then loss of rpm sensor error

AC6: *op* change which affected *file* (*alarm*)
if *alarm* triggered
file change after alarm
file not low or high level after alarm
op trend before alarm
then *op* change

AC7: *op* step which affected *file* (*alarm*)
if *alarm* triggered
file change after alarm
file not low or high level after alarm
op step before alarm
then *op* change

AC8: *op* change induced level change in *file* (*alarm*)
if *alarm* triggered
file stable level after alarm
file high level after alarm
op trend before alarm
then *op* change

AC9: *op* step induced level change in *file* (*alarm*)
if *alarm* triggered
file stable level after alarm
file high level after alarm
op step before alarm
then *op* change

AC10: *op* change induced genuine change in *file* (*alarm*)
if *alarm* triggered
file no stable level after alarm
file high level after alarm
op trend before alarm
then *op* genuine change

AC11: *op* step induced genuine change in *file* (*alarm*)
if *alarm* triggered
file no stable level after alarm
file high level after alarm
op step before alarm
then *op* genuine change

AC12: unattributed genuine change in *file* (*alarm*)
if *alarm* triggered

file no stable level after alarm
file high level after alarm
then unattributed genuine change

AC13: drift in *alarm*
if *alarm* triggered
file stable level after alarm
file stable level after alarm (for first and second order alarms)
no load trend before alarm
no load step before alarm
no rotor current trend before alarm
no rotor current step before alarm
no generator mvars trend before alarm
no generator mvars step before alarm
then drift

Appendix C. Selected Learning Module Look-Up-Tables

Each look-up-table has the initiating cause on the left hand side of the table with the associated effects on the right hand side of the table. Each cause and effect is numbered so that the relevant effects are matched with their associated effects. For example, if the cause is labelled 1 then all the effects labelled 1 are associated with that cause.

Normal Behavior		
Cause	**Effects**	
Generator Rotor Current (GRC-OA)	**Generator Load (GL-OA)**	**Generator Rotor Coil Temperature (GRCT-OA)**
1. Change Increase (Ch-Inc)	1. Change Increase (Ch-Inc)	1. Change Increase (Ch-Inc)
2. Change Decrease (Ch-Dec)	2. Change Decrease (Ch-Dec)	2. Change Decrease (Ch-Dec)
3. Level High (Lev-Hi)	3. N/A	3. N/A
4. Level Low (Lev-Lo)	4. N/A	4. N/A
5. Step Increase (St-Inc)	1. Change Increase (Ch-Inc)	5. Change Increase (Ch-Inc)
6. Step Decrease (St-Dec)	2. Change Decrease (Ch-Dec)	6. Change Decrease (Ch-Dec)

Figure C.1: Generator rotor current look-up-table (normal behaviour)

Stiction		
Cause	**Effects**	
Generator Rotor Coil Temperature (GRCT-OA)	**Generator Rotor Coil Dimension (GRCD-OA)**	**Generator Rotor Temperature (GRT-OA)**
1. Change Increase (Ch-Inc)	1. N/A	1. Change Increase (Ch-Inc)
2. Change Decrease (Ch-Dec)	2. Contract Uneven (CU)	2. Change Decrease (Ch-Dec)
3. Level High (Lev-Hi)	3. N/A	3. N/A
4. Level Low (Lev-Lo)	4. N/A	4. N/A
5. Step Increase (St-Inc)	5. N/A	5. Change Increase (Ch-Inc)
6. Step Decrease (St-Dec)	6. Contract Uneven (CU)	6. Change Decrease (Ch-Dec)

Figure C.2: Generator rotor coil temperature look-up-table (stiction behaviour)

Normal Behavior		
Cause	**Effects**	
Generator Rotor Coil Temperature (GRCT-OA)	**Generator Rotor Coil Dimension (GRCD-OA)**	**Generator Rotor Temperature (GRT-OA)**
1. Change Increase (Ch-Inc)	1. Expand Even (EE)	1. Change Increase (Ch-Inc)
2. Change Decrease (Ch-Dec)	2. Contract Even (CE)	2. Change Decrease (Ch-Dec)
3. Level High (Lev-Hi)	3. N/A	3. N/A
4. Level Low (Lev-Lo)	4. N/A	4. N/A
5. Step Increase (St-Inc)	5. Expand Even (EE)	5. Change Increase (Ch-Inc)
6. Step Decrease (St-Dec)	6. Contract Even (CE)	6. Change Decrease (Ch-Dec)

Figure C.3: Generator rotor coil temperature look-up-table (normal behaviour)

Normal Behaviour		
Cause	Effects	
Generator Rotor Balance (GRB-OA)	Generator Bearing Rear Vibration (GBRV-OA)	Generator Bearing Front Vibration (GBFV-OA)
1. Change Normal (Ch-Norm)	1. Change Increase (Ch-Inc) 1. Change Decrease (Ch-Dec)	1. Change Increase (Ch-Inc) 1. Change Decrease (Ch-Dec)
2. Change Significant (Ch-Sig)	2. Change Increase (Ch-Inc) [&1] 2. Level High (Lev-Hi) [&1]	2. Change Increase (Ch-Inc) [&2] 2. Level High (Lev-Hi) [&2]
3. Change Negligible (Ch-Neg)	3. Nevel No Change (Lev-NCh)	3. Nevel No Change (Lev-NCh)

Figure C.4: Generator rotor balance look-up-table (normal behaviour)

Normal Behavior	
Cause	Effects
Generator Rotor Coil Dimension (GRCD-OA)	Generator Rotor Balance (GRB-OA)
1. Expand Even (EE)	1. Change Normal (Ch-Norm)
2. Expand Uneven (EU)	2. Change Significant (Ch-Sig)
3. Contract Even (CE)	3. Change Normal (Ch-Norm)
4. Contract Uneven (CU)	4. Change Negligible (Ch-Neg)

Figure C.5: Generator rotor coil dimension look-up-table (normal behaviour)

Normal Behavior		
Cause	Effects	
Generator Rotor Temperature (GRT-OA)	Generator Rotor Dimension (GRD-OA)	Generator Shaft Temperature (GShT-OA)
1. Change Increase (Ch-Inc)	1. Expand Even (EE)	1. Change Increase (Ch-Inc)
2. Change Decrease (Ch-Dec)	2. Contract Even (CE)	2. Change Decrease (Ch-Dec)
3. Level High (Lev-Hi)	3. N/A	3. N/A
4. Level Low (Lev-Lo)	4. N/A	4. N/A
5. Step Increase (St-Inc)	5. Expand Even (EE)	5. Change Increase (Ch-Inc)
6. Step Decrease (St-Dec)	6. Contract Even (CE)	6. Change Decrease (Ch-Dec)

Figure C.6: Generator rotor temperature look-up-table (normal behaviour)

Normal Behavior		
Cause	Effects	
Generator Shaft Temperature (GShT-OA)	Generator Shaft Dimension (GShD-OA)	Generator Rotor Temperature (GRT-OA)
1. Change Increase (Ch-Inc)	1. Expand Even (EE)	1. Change Increase (Ch-Inc)
2. Change Decrease (Ch-Dec)	2. Contract Even (CE)	2. Change Decrease (Ch-Dec)
3. Level High (Lev-Hi)	3. N/A	3. N/A
4. Level Low (Lev-Lo)	4. N/A	4. N/A
5. Step Increase (St-Inc)	5. Expand Even (EE)	5. Change Increase (Ch-Inc)
6. Step Decrease (St-Dec)	6. Contract Even (CE)	6. Change Decrease (Ch-Dec)

Figure C.7: Generator shaft temperature look-up-table (normal behaviour)

Normal Behavior	
Cause	**Effects**
Generator Shaft Dimension (GShD-OA)	Generator Rotor Balance (GRB-OA)
1. Expand Even (EE) 2. Expand Uneven (EU) 3. Contract Even (CE) 4. Contract Uneven (CU)	1. Change Normal (Ch-Norm) 2. Change Significant (Ch-Sig) 3. Change Normal (Ch-Norm) 4. Change Negligible (Ch-Neg)

Figure C.8: Generator shaft dimension look-up-table (normal behaviour)

Normal Behavior	
Cause	**Effects**
Generator Rotor Dimension (GRD-OA)	Generator Rotor Balance (GRB-OA)
1. Expand Even (EE) 2. Expand Uneven (EU) 3. Contract Even (CE) 4. Contract Uneven (CU)	1. Change Normal (Ch-Norm) 2. Change Significant (Ch-Sig) 3. Change Normal (Ch-Norm) 4. Change Negligible (Ch-Neg)

Figure C.9: Generator rotor dimension look-up-table (normal behaviour)

Normal Behaviour		
Cause	**Effects**	
Generator Bearing Rear Vibration (GBRV-OA)	Generator Shaft Vibration (GShV-OA)	Generator Bearing Rear (GBRV-OA)
1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-How) 5. Level No Change (Lev-NCh) 6. Step Increase (St-Inc) 7. Step Decreas (St-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. N/A 4. N/A 5. N/A 6. Change Increase (Ch-Inc) 7. Change Decrease (Ch-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-How) 5. Level No Change (Lev-NCh) 6. Change Increase (Ch-Inc) 7. Change Decrease (Ch-Dec)
	Generator Bearing Rear (GBRV-1M)	Generator Bearing Rear (GBRV-2M)
	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-How) 5. Level No Change (Lev-NCh) 6. Change Increase (Ch-Inc) 7. Change Decrease (Ch-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. N/A 4. N/A 5. N/A 6. Change Increase (Ch-Inc) 7. Change Decrease (Ch-Dec)

Figure C.10: Generator bearing rear vibration look-up-table (normal behaviour)

Normal Behaviour		
Cause	**Effects**	
Generator Bearing Front Vibration (GBRV-OA)	**Generator Shaft Vibration (GShV-OA)**	**Generator Bearing Front (GBRV-OA)**
1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-Low) 5. Level No Change (Lev-NCh) 6. Step Increase (St-Inc) 7. Step Decreas (St-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. N/A 4. N/A 5. N/A 6. Change Increase (Ch-Inc) 7. Change Decrease (Ch-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-Low) 5. Level No Change (Lev-NCh) 6. Change Increase (Ch-Inc) 7. Change Decrease (Ch-Dec)
	Generator Bearing Front (GBRV-1M)	**Generator Bearing Front (GBRV-2M)**
	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-How) 5. Level No Change (Lev-NCh) 6. Change Increase (Ch-Inc) 7. Change Decrease (Ch-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. N/A 4. N/A 5. N/A 6. Change Increase (Ch-Inc) 7. Change Decrease (Ch-Dec)

Figure C.11: Generator bearing front vibration look-up-table (normal behaviour)

Normal Behaviour		
Cause	**Effects**	
Rotations Per Minute (RPM-OA)	**Generator Shaft Speed (GShS-OA)**	**Low Pressure B Shaft Speed (LPBShS-OA)**
1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-Low) 5. Step Increase (Step-Inc) 6. Step Decrease (Step-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-Low) 5. Step Increase (Step-Inc) 6. Step Decrease (Step-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-Low) 5. Step Increase (Step-Inc) 6. Step Decrease (Step-Dec)
	Low Pressure A Shaft Speed (LPAShS-OA)	**Intermediate Pressure Shaft Speed (IPShS-OA)**
	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-Low) 5. Step Increase (Step-Inc) 6. Step Decrease (Step-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-Low) 5. Step Increase (Step-Inc) 6. Step Decrease (Step-Dec)
	High Pressure Shaft Speed (HPShS-OA)	**Turbine Shaft Speed (TShS-OA)**
	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-Low) 5. Step Increase (Step-Inc) 6. Step Decrease (Step-Dec)	1. Change Increase (Ch-Inc) 2. Change Decrease (Ch-Dec) 3. Level High (Lev-Hi) 4. Level Low (Lev-Low) 5. Step Increase (Step-Inc) 6. Step Decrease (Step-Dec)

Figure C.12: Rotations per minute look-up-table (normal behaviour)

Generator Pedestal Front Looseness		
Cause	**Effects**	
Turbine Shaft Speed (TShS-OA)	**Generator Bearing Front (GShS-2M)**	**Low Pressure B Bearing Rear (LPBBR-2M)**
1. Change Increase (Ch-Inc)	1. Change Increase (Ch-Inc) [&1]	1. Change Increase (Ch-Inc) [&1]
	1. Level High (Lev-Hi) [&1]	1. Level High (Lev-Hi) [&1]
2. Change Decrease (Ch-Dec)	2. Change Decrease (Ch-Dec)	2. Change Decrease (Ch-Dec)
3. Level High (Lev-Hi)	3. N/A	3. N/A
4. Level Low (Lev-Low)	4. N/A	4. N/A
5. Step Increase (St-Inc)	5. Change Increase (Ch-Inc) [&2]	5. Change Increase (Ch-Inc) [&2]
	5. Level High (Lev-Hi) [&2]	5. Level High (Lev-Hi) [&2]
6. Step Decrease (St-Dec)	6. Change Decrease (Ch-Dec)	6. Change Decrease (Ch-Dec)

Figure C.13: Turbine shaft speed look-up-table (generator pedestal front looseness fault)

Stiction Fault Beran Data

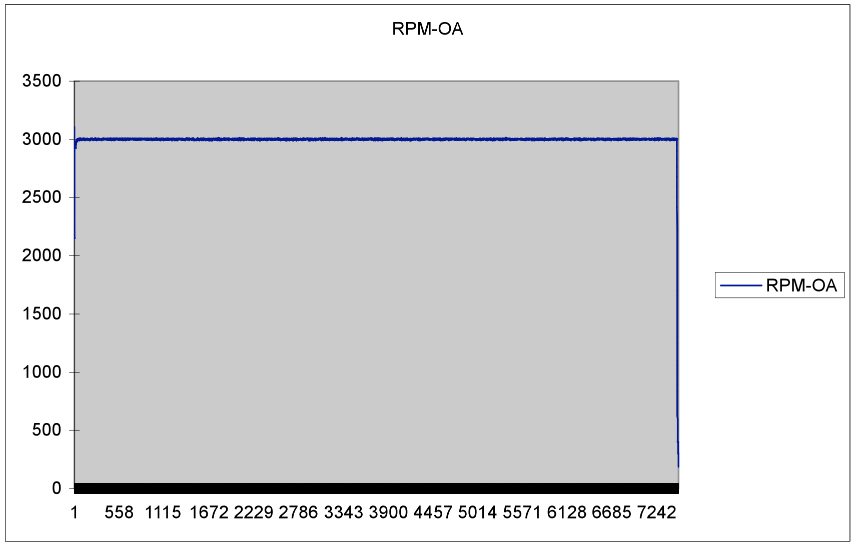


Figure D.1: Turbine 2, 15/06/2006 – 02/08/2006, rotations per minute raw data

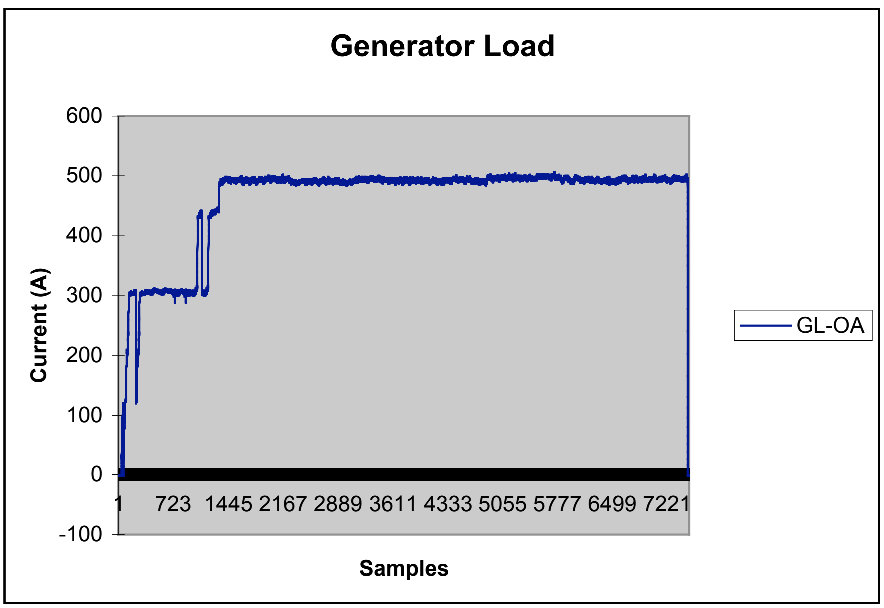


Figure D.2: Turbine 2, 15/06/2006 – 02/08/2006, generator load raw data

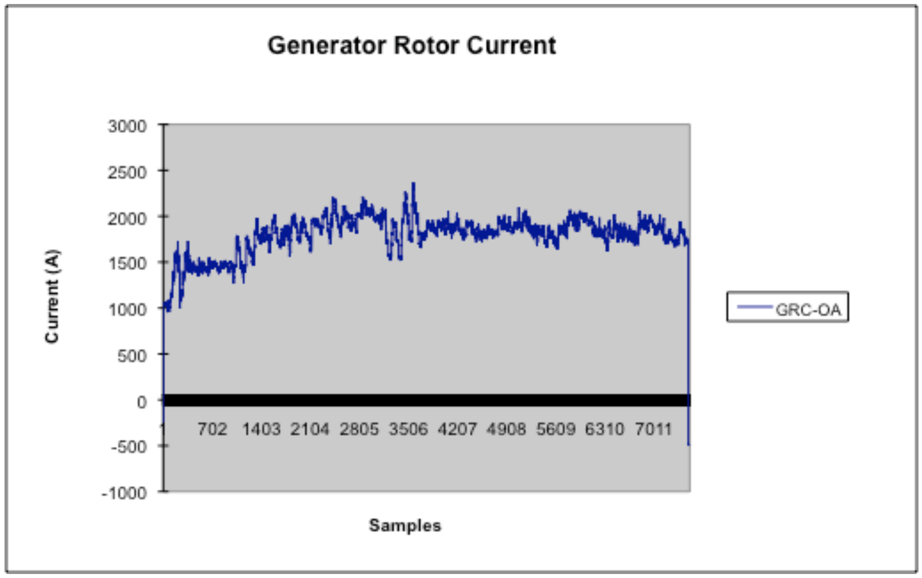


Figure D.3: Turbine 2, 15/06/2006 – 02/08/2006, generator rotor current raw data

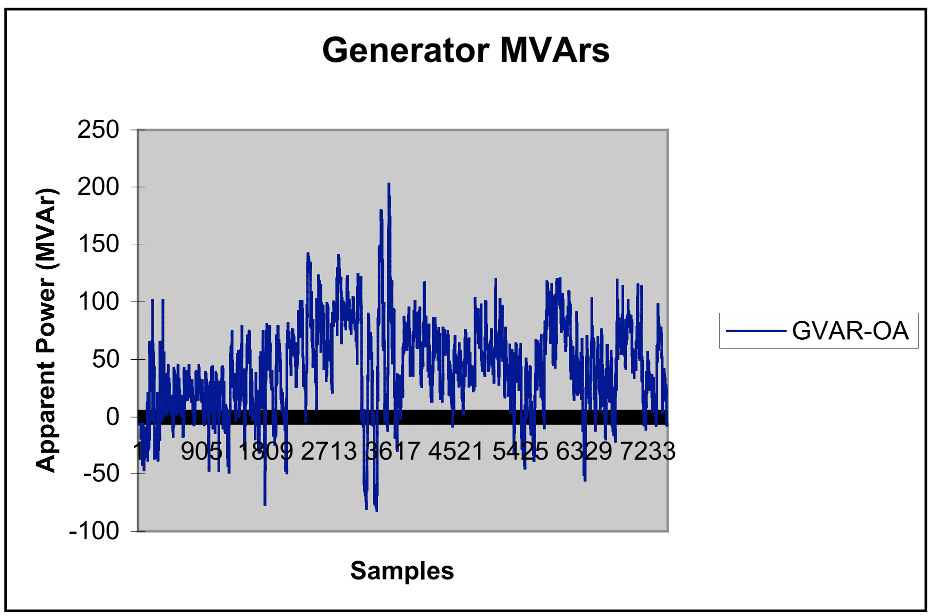


Figure D.4: Turbine 2, 15/06/2006 – 02/08/2006, generator rotor MVArs raw data

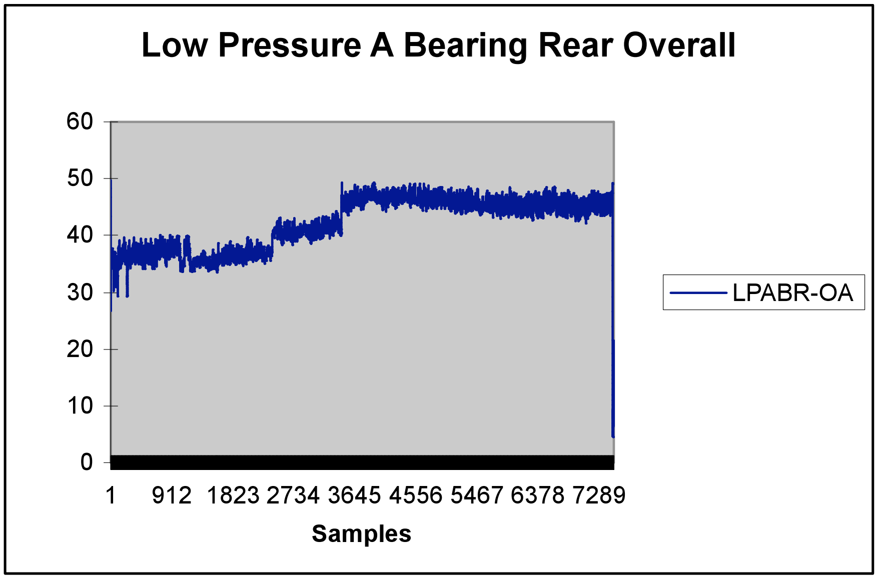


Figure D.5: Turbine 2, channel 6, 15/06/2006 – 02/08/2006, low pressure A bearing rear overall vibration raw data

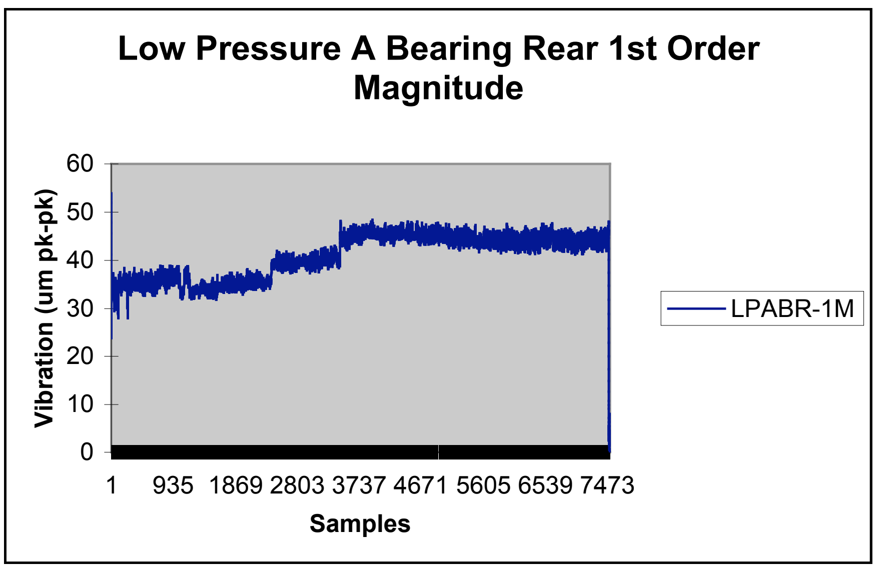


Figure D.6: Turbine 2, channel 6, 15/06/2006 – 02/08/2006, low pressure A bearing rear 1st order magnitude vibration raw data

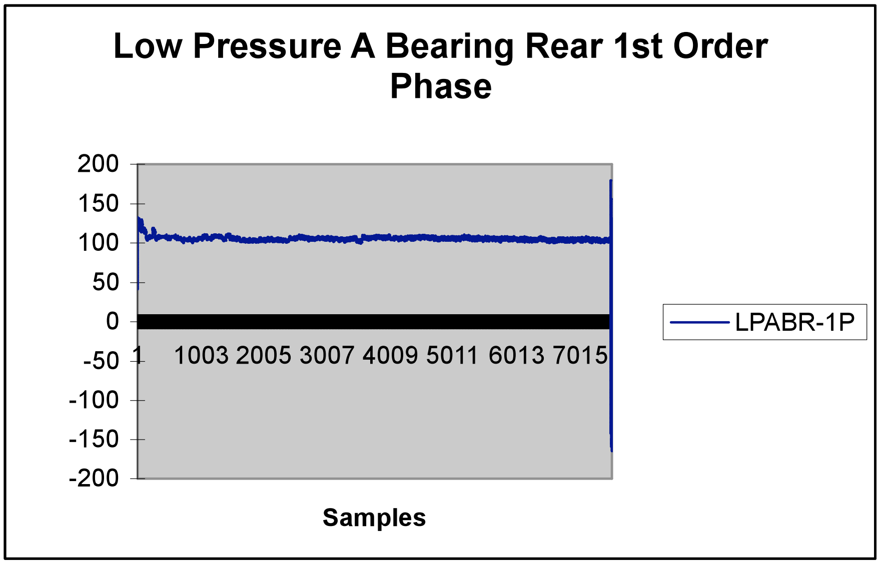


Figure D.7: Turbine 2, channel 6, 15/06/2006 – 02/08/2006, low pressure A bearing rear 1st order phase vibration raw data

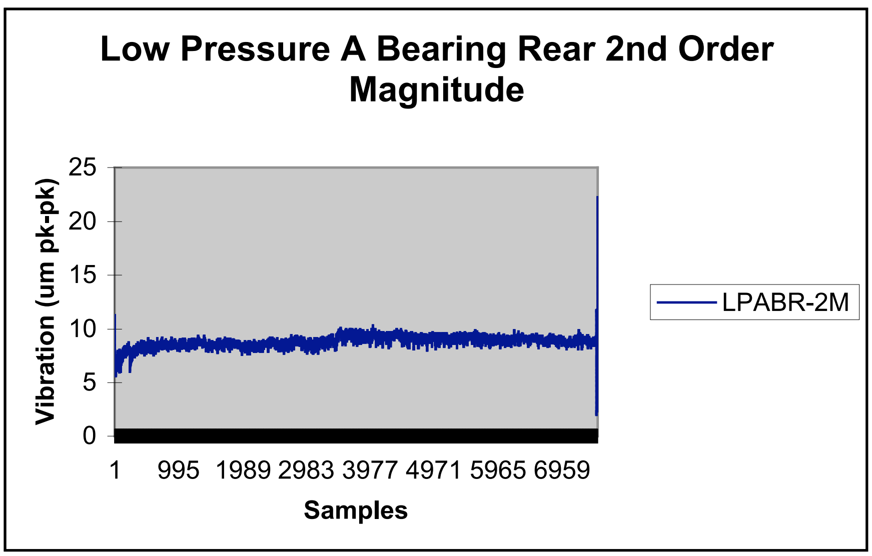


Figure D.8: Turbine 2, channel 6, 15/06/2006 – 02/08/2006, low pressure A bearing rear 2nd order magnitude vibration raw data

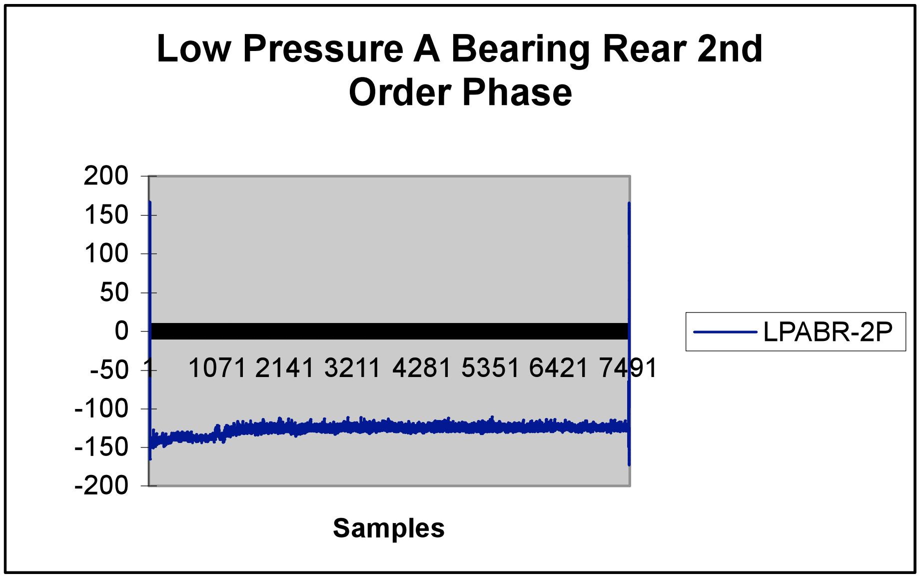


Figure D.9: Turbine 2, channel 6, 15/06/2006 – 02/08/2006, low pressure A bearing rear 2nd order phase vibration raw data

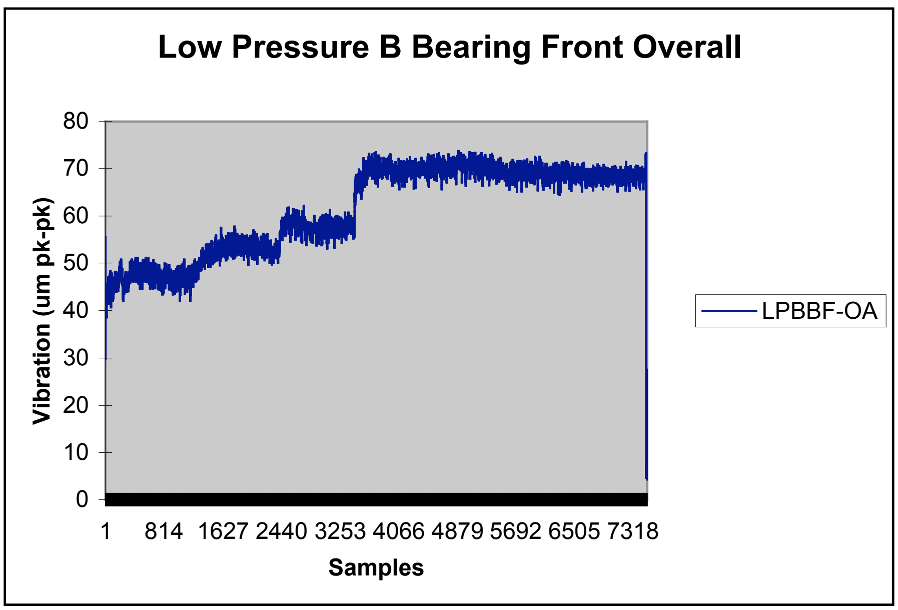


Figure D.10: Turbine 2, channel 7, 15/06/2006 – 02/08/2006, low pressure B bearing front overall vibration raw data

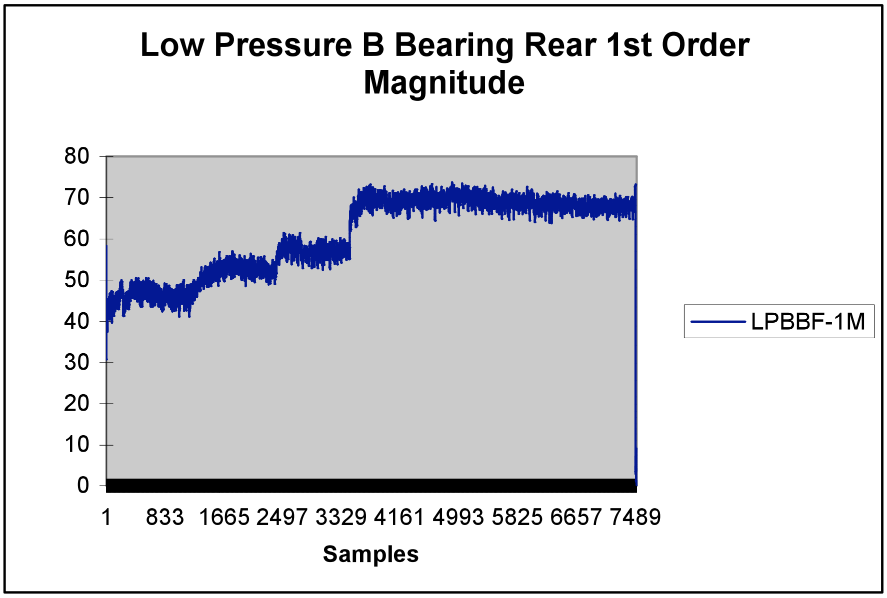


Figure D.11: Turbine 2, channel 7, 15/06/2006 – 02/08/2006, low pressure B bearing front 1st order magnitude vibration raw data

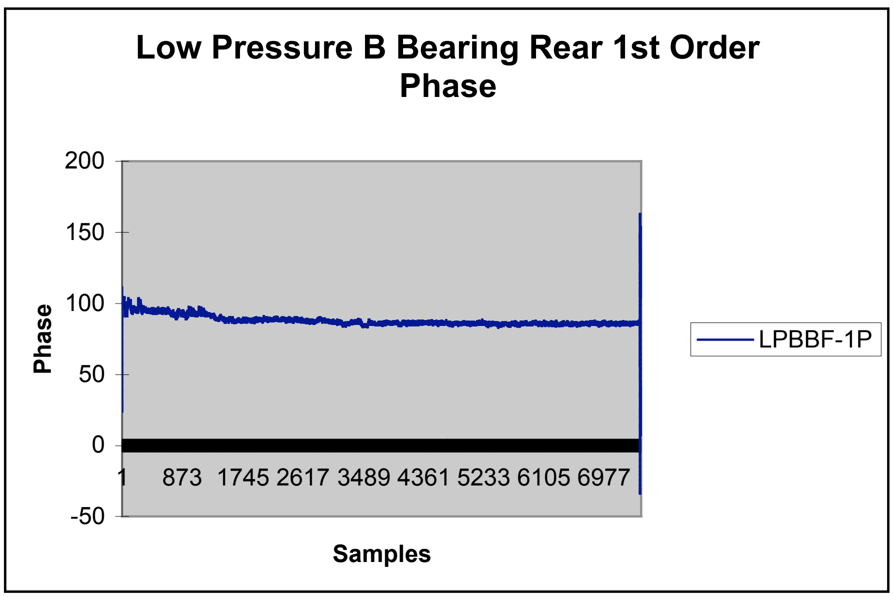


Figure D.12: Turbine 2, channel 7, 15/06/2006 – 02/08/2006, low pressure B bearing front 1st order phase vibration raw data

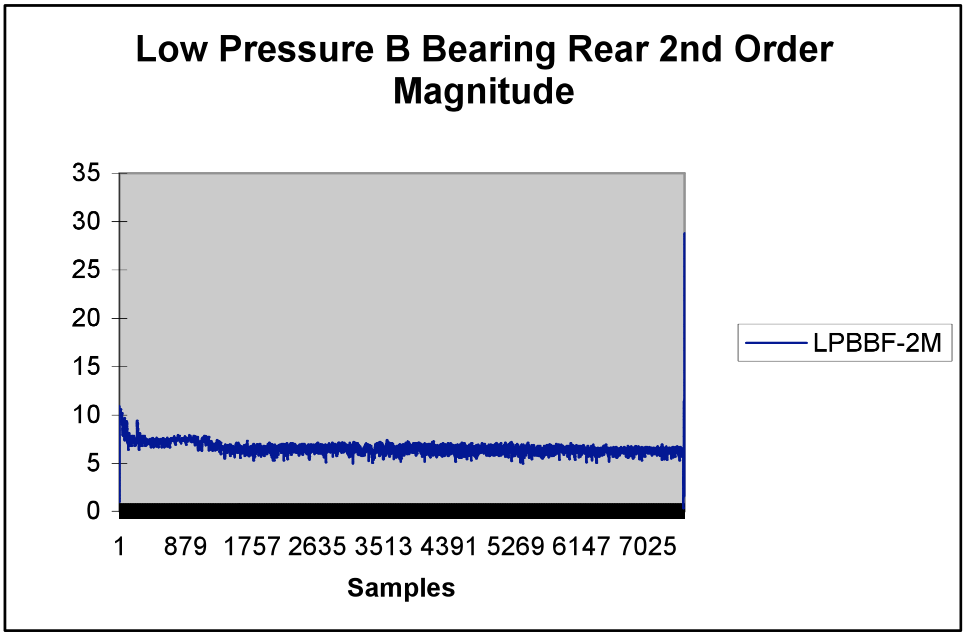


Figure D.13: Turbine 2, channel 7, 15/06/2006 – 02/08/2006, low pressure B bearing front 2nd order magnitude vibration raw data

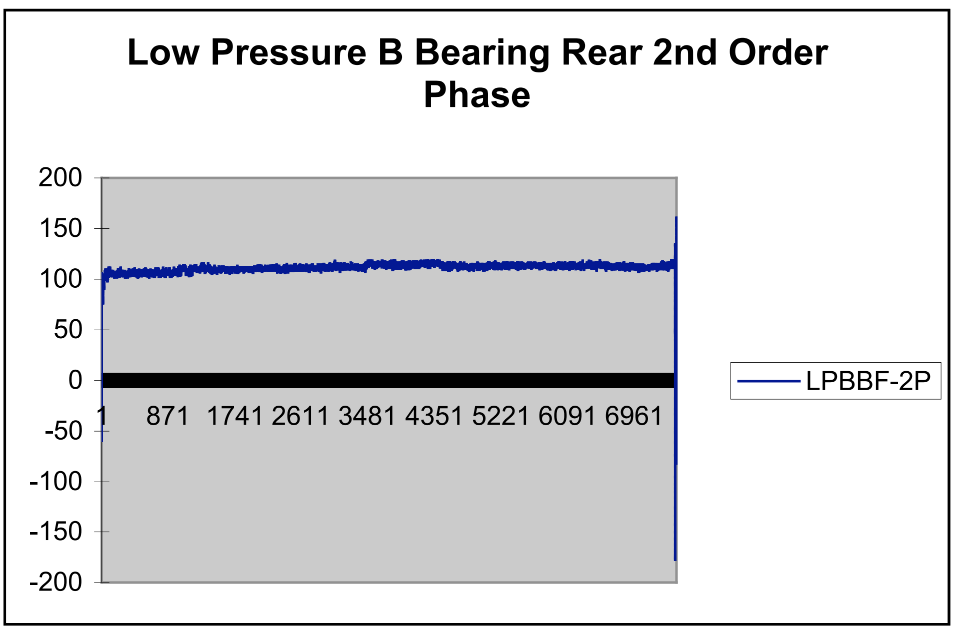


Figure D.14: Turbine 2, channel 7, 15/06/2006 – 02/08/2006, low pressure B bearing front 2nd order phase vibration raw data

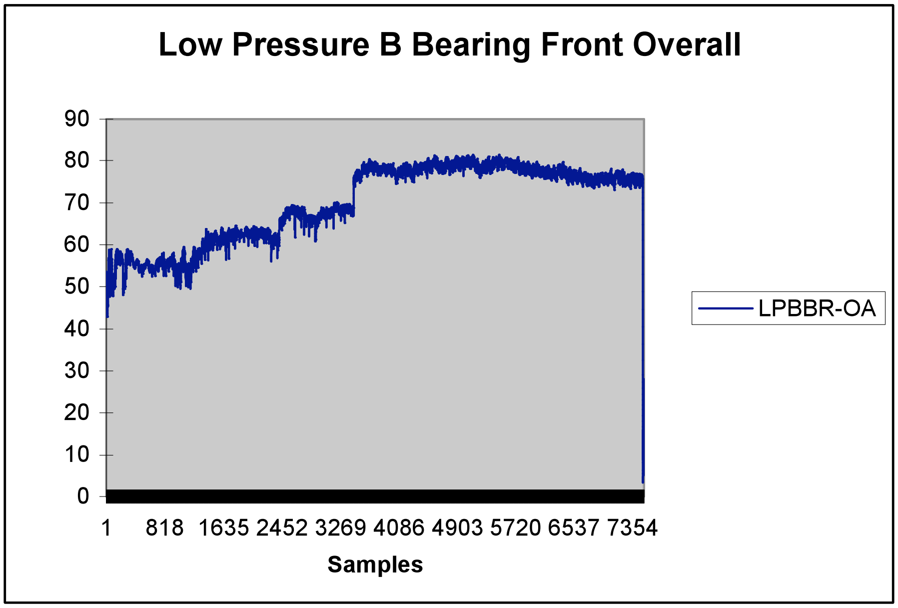


Figure D.15: Turbine 2, channel 8, 15/06/2006 – 02/08/2006, low pressure B bearing rear overall vibration raw data

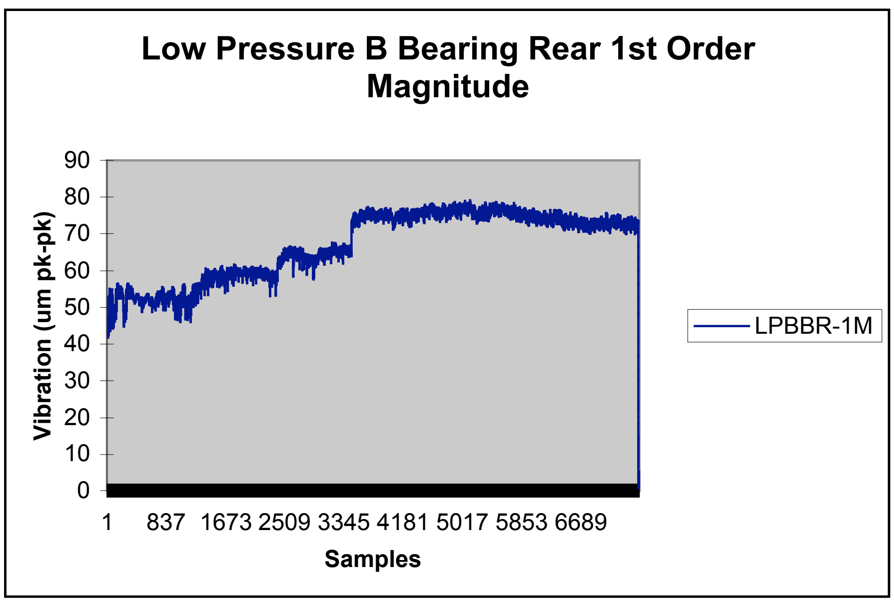


Figure D.16: Turbine 2, channel 8, 15/06/2006 – 02/08/2006, low pressure B bearing rear 1st order magnitude vibration raw data

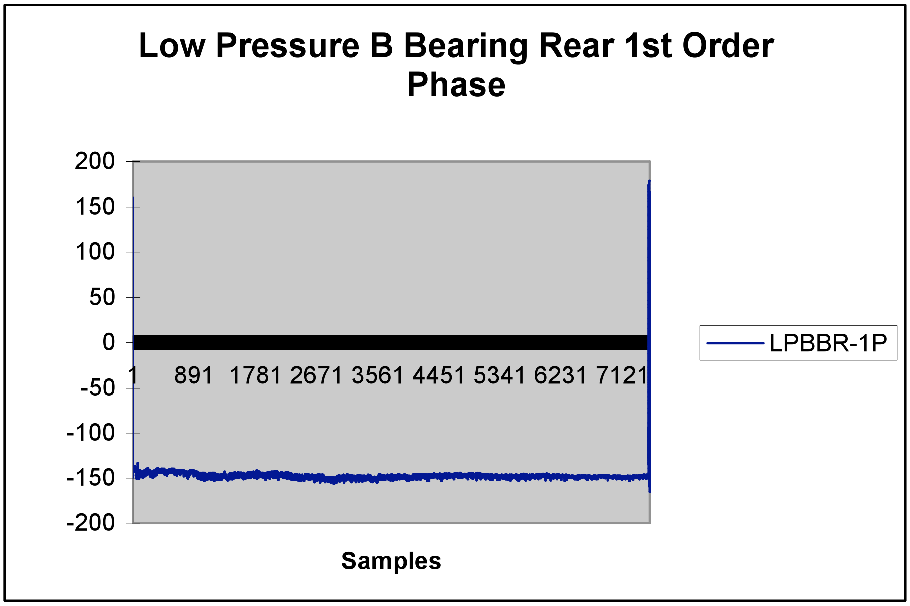


Figure D.17: Turbine 2, channel 8, 15/06/2006 – 02/08/2006, low pressure B bearing rear 1st order phase vibration raw data

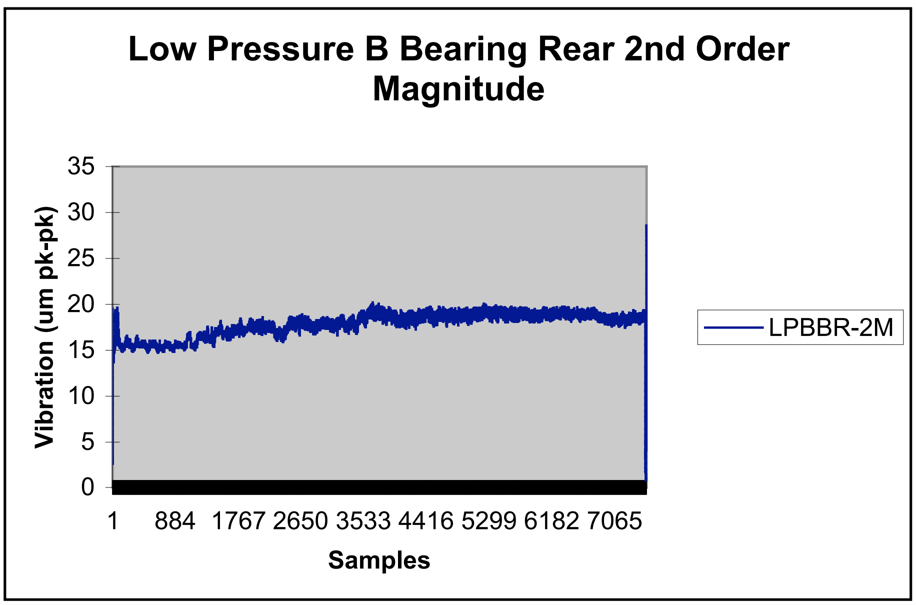


Figure D.18: Turbine 2, channel 8, 15/06/2006 – 02/08/2006, low pressure B bearing rear 2nd order magnitude vibration raw data

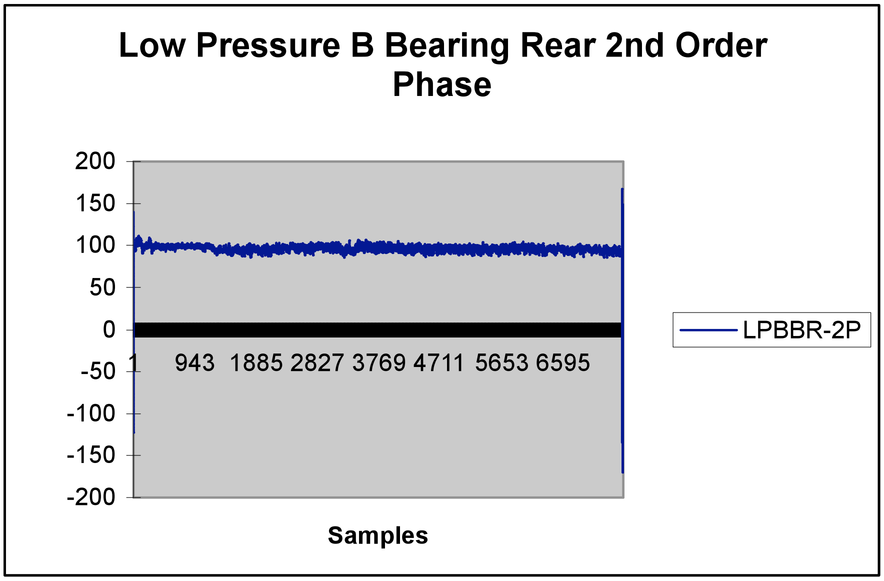


Figure D.19: Turbine 2, channel 8, 15/06/2006 – 02/08/2006, low pressure B bearing rear 2nd order phase vibration raw data

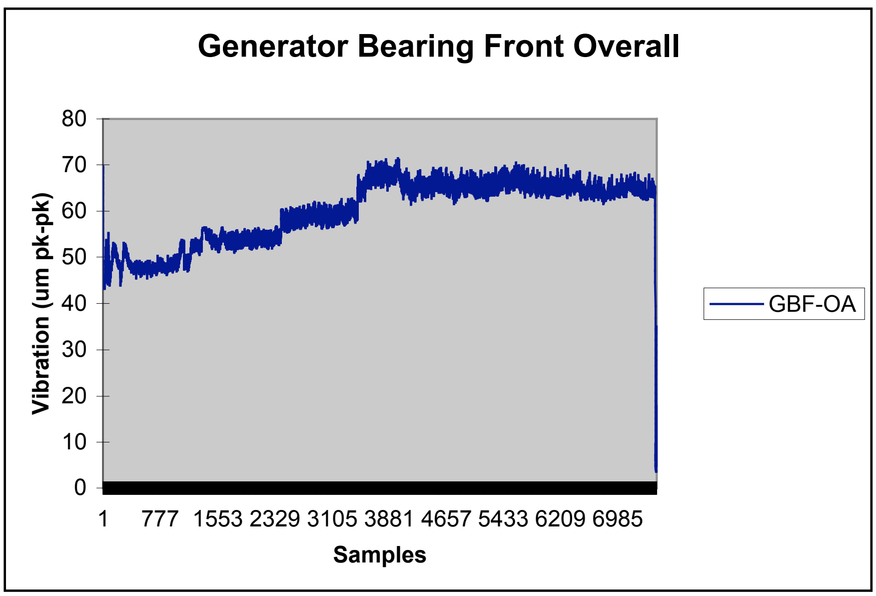


Figure D.20: Turbine 2, channel 9, 15/06/2006 – 02/08/2006, generator bearing front overall vibration raw data

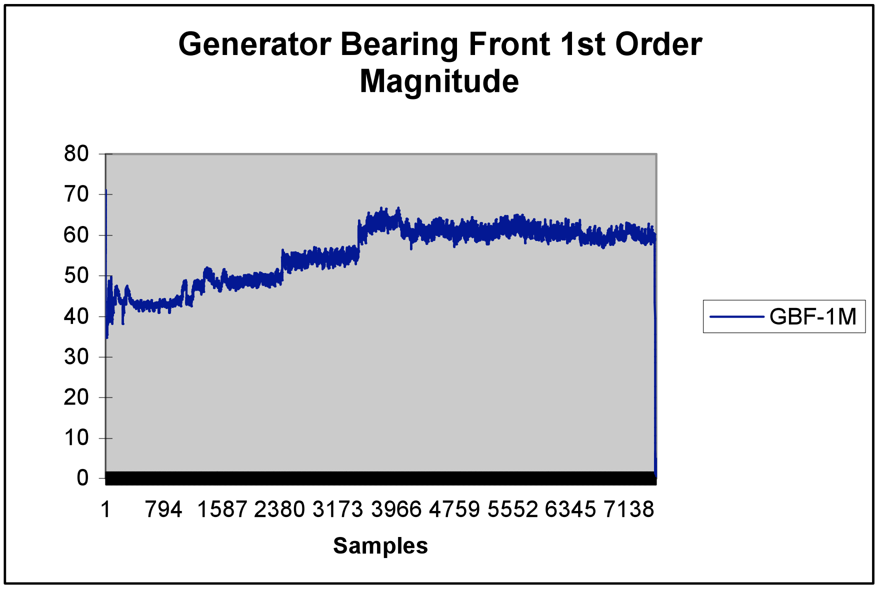


Figure D.21: Turbine 2, channel 9, 15/06/2006 – 02/08/2006, generator bearing front 1st order magnitude vibration raw data

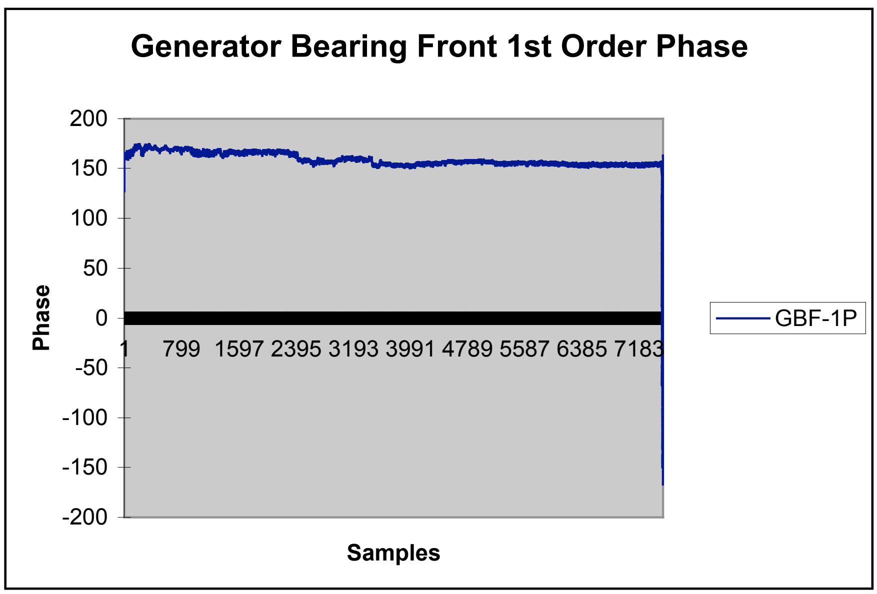


Figure D.22: Turbine 2, channel 9, 15/06/2006 – 02/08/2006, generator bearing front 1st order phase vibration raw data

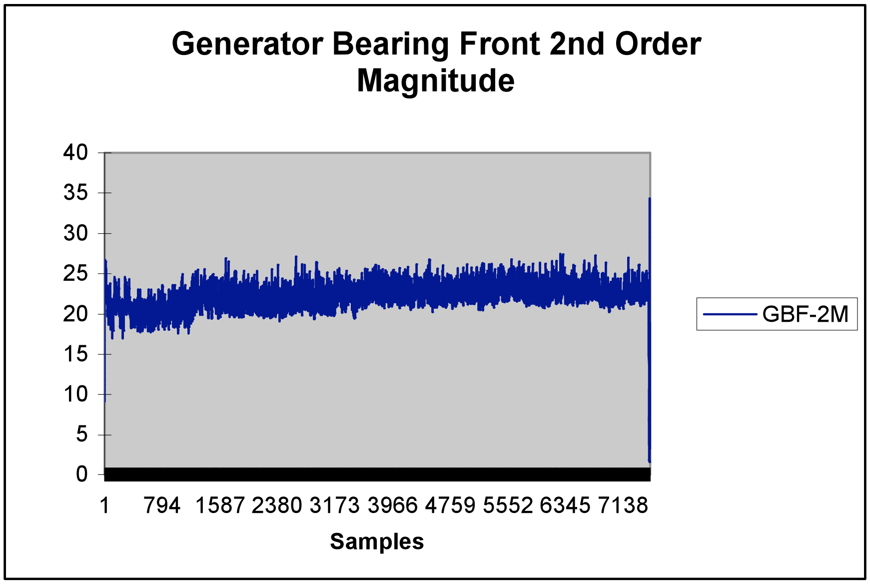


Figure D.23: Turbine 2, channel 9, 15/06/2006 – 02/08/2006, generator bearing front 2nd order magnitude vibration raw data

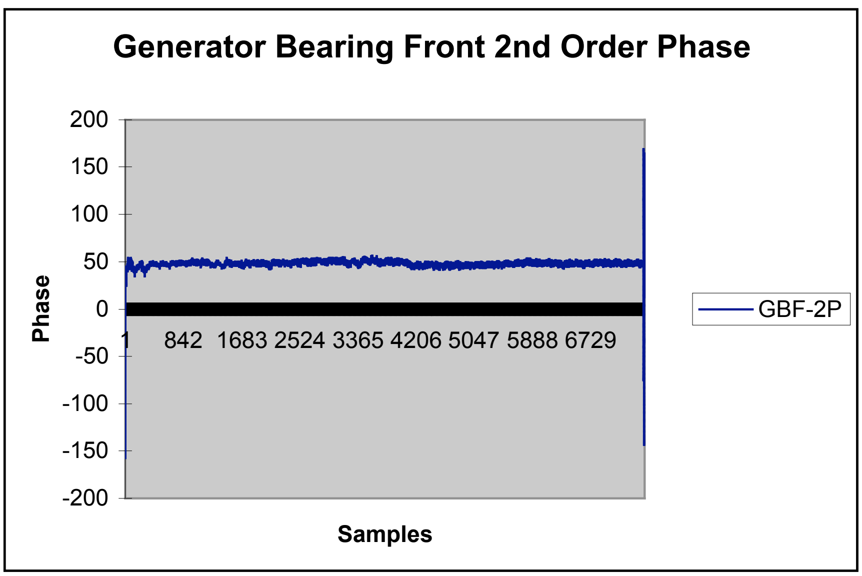


Figure D.24: Turbine 2, channel 9, 15/06/2006 – 02/08/2006, generator bearing front 2nd order phase vibration raw data

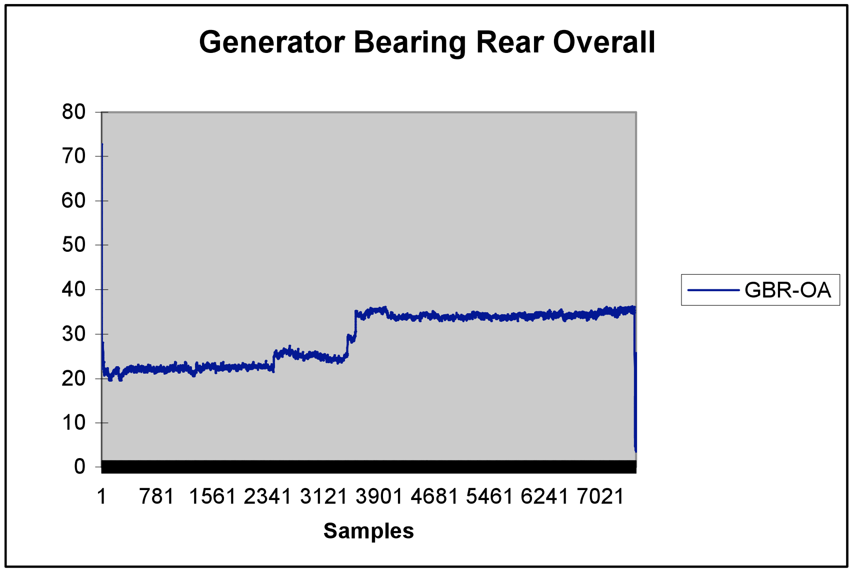


Figure D.25: Turbine 2, channel 10, 15/06/2006 – 02/08/2006, generator bearing rear overall vibration raw data

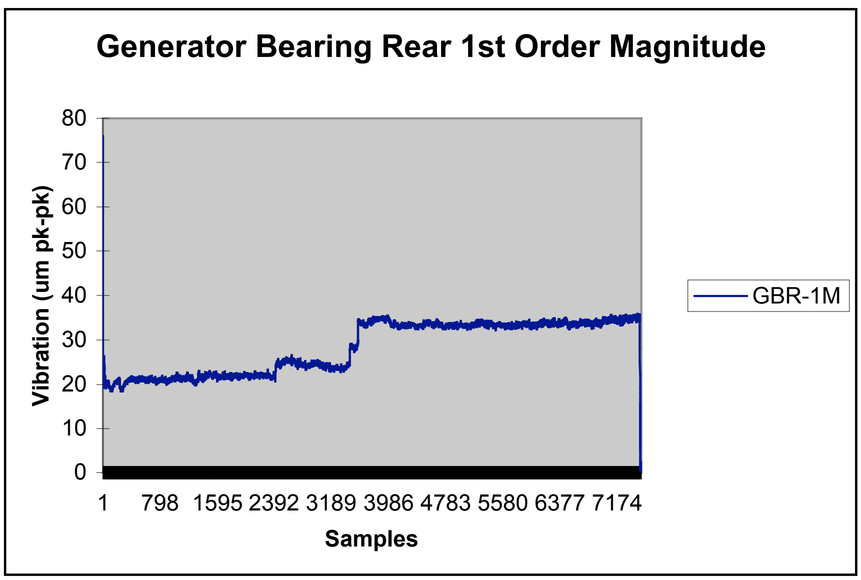


Figure D.26: Turbine 2, channel 10, 15/06/2006 – 02/08/2006, generator bearing rear 1st order magnitude vibration raw data

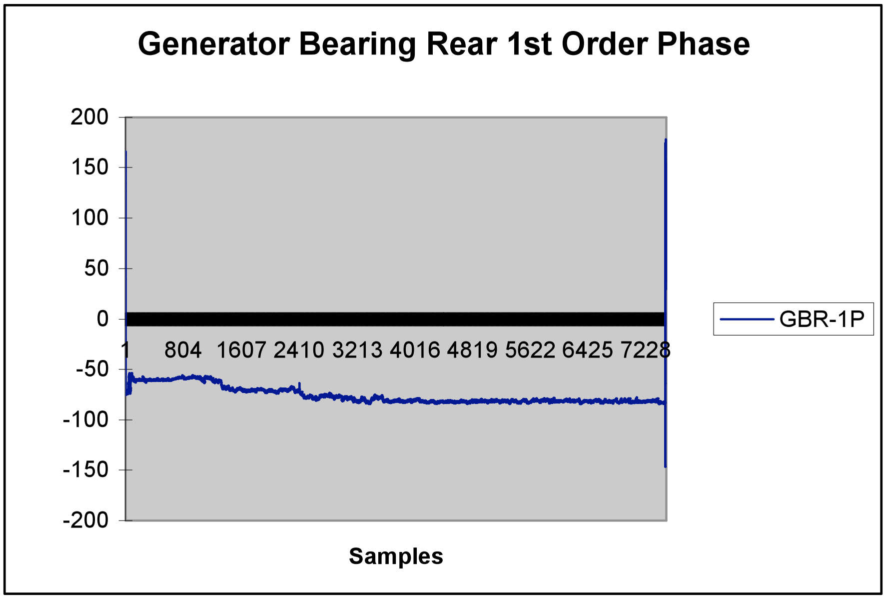


Figure D.27: Turbine 2, channel 10, 15/06/2006 – 02/08/2006, generator bearing rear 1st order phase vibration raw data

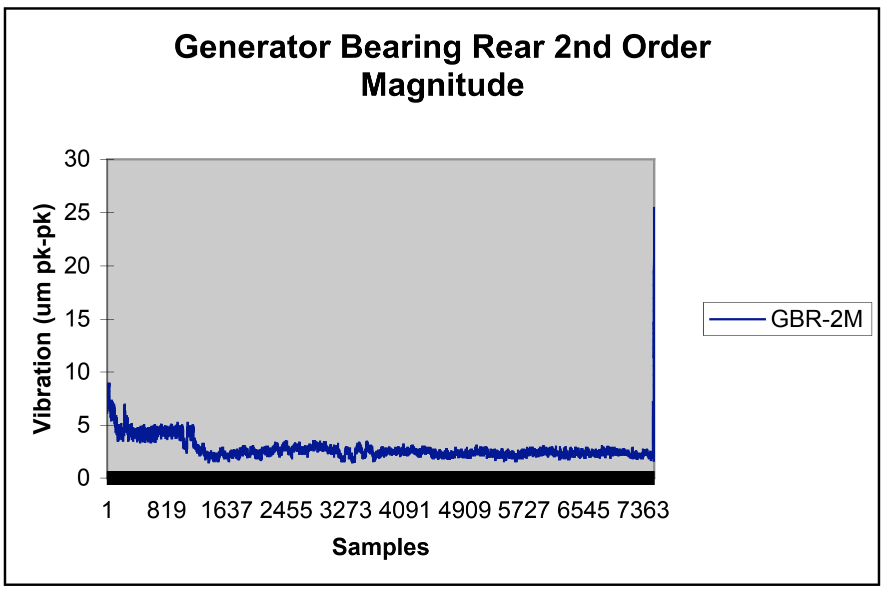


Figure D.28: Turbine 2, channel 10, 15/06/2006 – 02/08/2006, generator bearing rear 2nd order magnitude vibration raw data

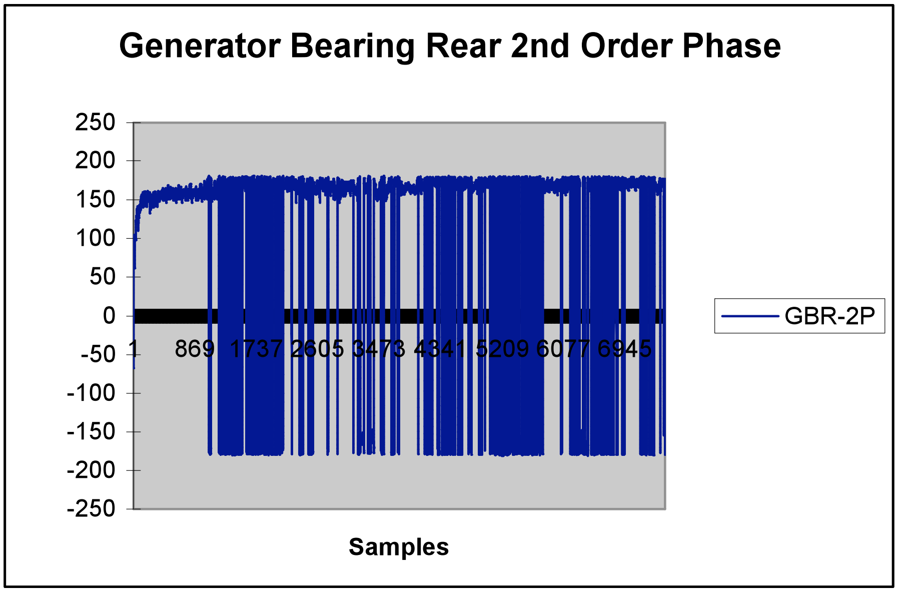


Figure D.29: Turbine 2, channel 10, 15/06/2006 – 02/08/2006, generator bearing rear 2nd order phase vibration raw data

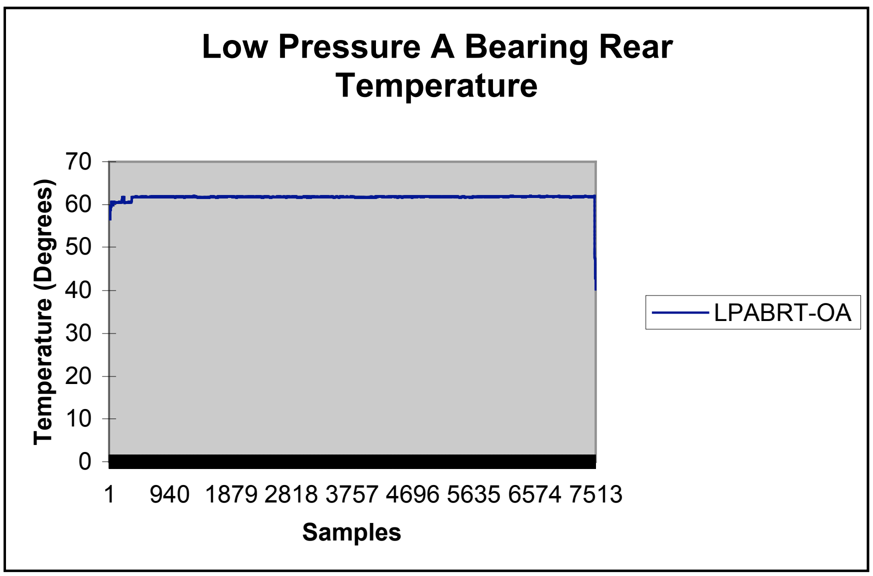


Figure D.30: Turbine 2, channel 6, 15/06/2006 – 02/08/2006, low pressure A bearing rear temperature raw data

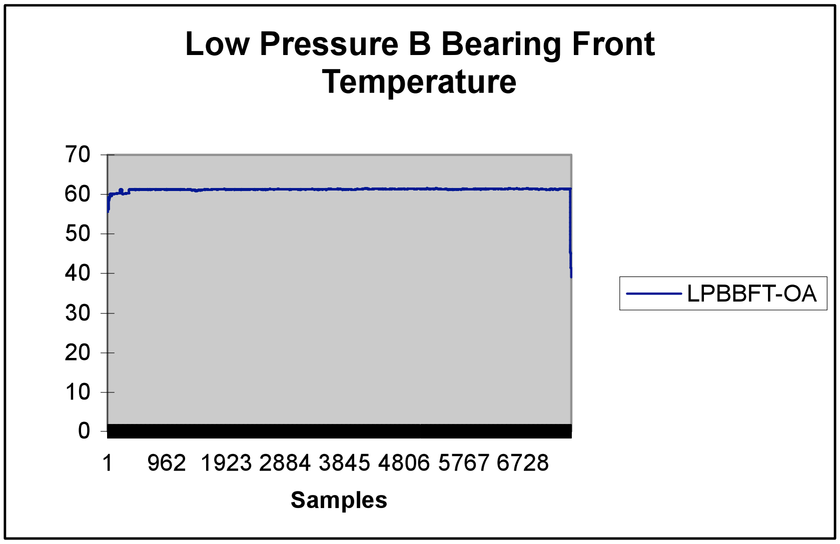


Figure D.31: Turbine 2, channel 7, 15/06/2006 – 02/08/2006, low pressure B bearing front temperature raw data

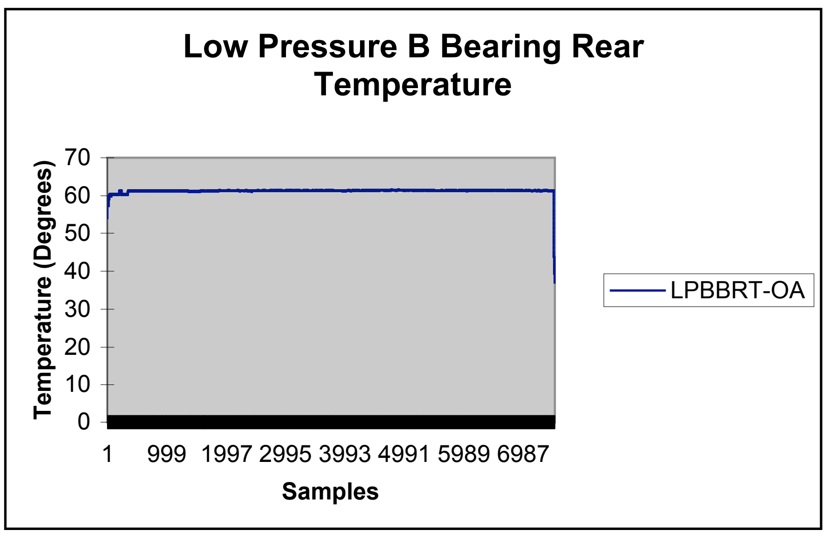


Figure D.32: Turbine 2, channel 8, 15/06/2006 – 02/08/2006, low pressure B bearing rear temperature raw data

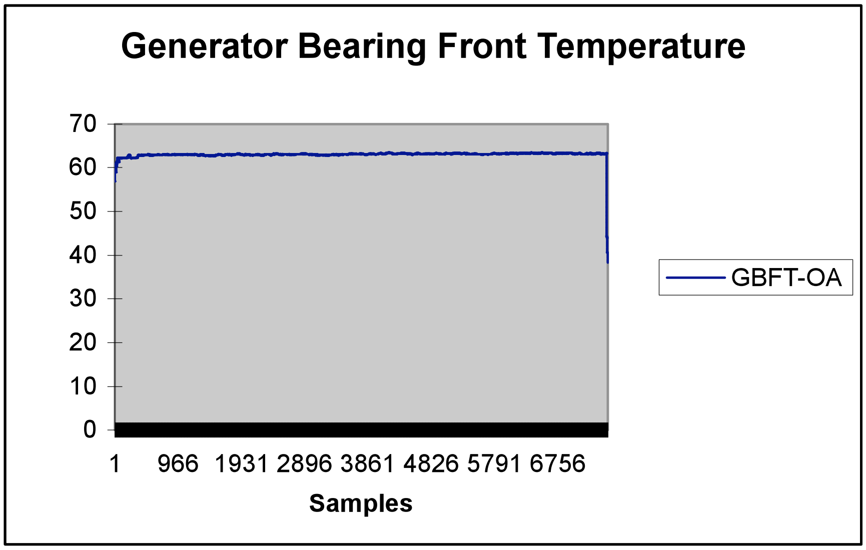


Figure D.33: Turbine 2, channel 9, 15/06/2006 – 02/08/2006, generator bearing front temperature raw data

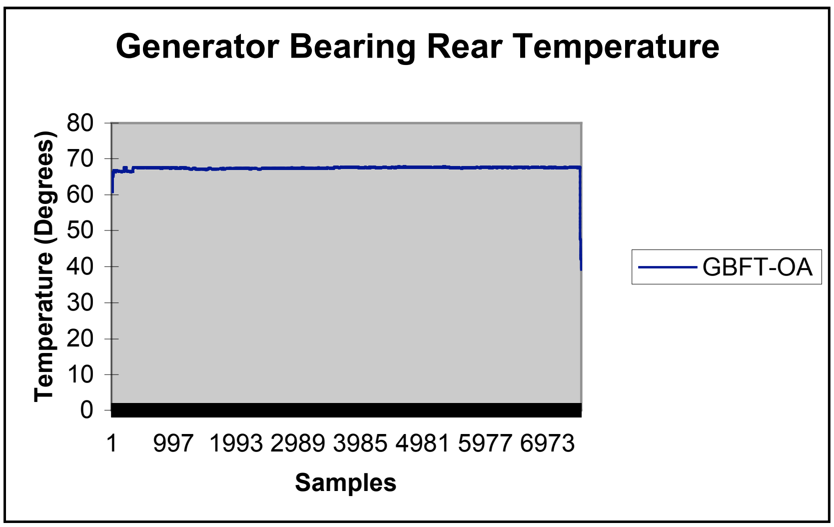


Figure D.34: Turbine 2, channel 10, 15/06/2006 – 02/08/2006, generator bearing rear temperature raw data

Stiction Fault Channel Profile

Stiction,,,
LPABR-OA,Level-St,0,7488
LPABR-OA,Change-St,0,7488
LPABR-1M,Level-St,0,7488
LPABR-1M,Change-St,0,7488
LPABR-1P,Level-St,0,7488
LPABR-1P,Change-St,0,7488
LPABR-2M,Level-St,0,7488
LPABR-2M,Change-Inc,0,144
LPABR-2M,Change-St,144,7344
LPABR-2P,Level-St,0,7488
LPABR-2P,Change-Dec,0,144
LPABR-2P,Change-St,144,7344
LPBBF-OA,Level-St,0,3456
LPBBF-OA,Level-Hi,3456,4032
LPBBF-OA,Change-St,0,7488
LPBBF-1M,Level-St,0,3456
LPBBF-1M,Level-Hi,3456,4032
LPBBF-1M,Change-St,0,7488
LPBBF-1P,Level-St,0,7488
LPBBF-1P,Change-St,0,7488
LPBBF-2M,Level-St,0,7488
LPBBF-2M,Change-Dec,0,144
LPBBF-2M,Change-St,144,7344
LPBBF-2P,Level-St,0,7488
LPBBF-2P,Change-Inc,0,144
LPBBF-2P,Change-St,144,7344
LPBBR-OA,Level-St,0,1440
LPBBR-OA,Level-Hi,1440,6048
LPBBR-OA,Change-Inc,0,144
LPBBR-OA,Change-St,144,7344
LPBBR-1M,Level-St,0,2448
LPBBR-1M,Level-Hi,2448,5040
LPBBR-1M,Change-Inc,0,144
LPBBR-1M,Change-St,144,2160
LPBBR-1M,Change-Inc,2304,144
LPBBR-1M,Change-St,2448,5040
LPBBR-1P,Level-St,0,7488
LPBBR-1P,Change-Dec,0,144
LPBBR-1P,Change-St,144,7344
LPBBR-2M,Level-Hi,0,7488
LPBBR-2M,Change-St,0,1152
LPBBR-2M,Change-Inc,1152,144
LPBBR-2M,Change-St,1296,6192
LPBBR-2P,Level-St,0,7488
LPBBR-2P,Change-St,0,7488
GBF-OA,Level-St,0,3312
GBF-OA,Level-Hi,3312,4176
GBF-OA,Change-Inc,0,144
GBF-OA,Change-St,144,2160
GBF-OA,Change-Inc,2304,144
GBF-OA,Change-St,2448,5040
GBF-1M,Level-St,0,3456
GBF-1M,Level-Hi,3456,3024
GBF-1M,Level-St,6480,576
GBF-1M,Level-Hi,7056,144
GBF-1M,Level-St,7200,288
GBF-1M,Change-Inc,0,144
GBF-1M,Change-St,144,2160
GBF-1M,Change-Inc,2304,144
GBF-1M,Change-St,2448,5040
GBF-1P,Level-St,0,7488
GBF-1P,Change-St,0,7488
GBF-2M,Level-Hi,0,7488
GBF-2M,Change-St,0,7488
GBF-2P,Level-St,0,7488
GBF-2P,Change-St,0,7488
RPM-OA,Level-Low,0,144
RPM-OA,Level-St,144,7344
RPM-OA,Change-Inc,0,144
RPM-OA,Change-St,144,7344
GBR-OA,Level-Low,0,3456
GBR-OA,Level-St,3456,4032
GBR-OA,Change-Dec,0,144
GBR-OA,Change-St,144,3312
GBR-OA,Change-Inc,3456,144
GBR-OA,Change-St,3600,3888
GBR-1M,Level-Low,0,3600
GBR-1M,Level-St,3600,3888
GBR-1M,Change-Dec,0,144
GBR-1M,Change-St,144,3312
GBR-1M,Change-Inc,3456,144
GBR-1M,Change-St,3600,3888
GBR-1P,Level-St,0,7488
GBR-1P,Change-St,0,7488
GBR-2M,Level-St,0,1008
GBR-2M,Level-Low,1008,6480
GBR-2M,Change-Dec,0,144
GBR-2M,Change-St,144,1008
GBR-2M,Change-Dec,1152,144
GBR-2M,Change-St,1296,6192
GBR-2P,Level-St,0,7488
GBR-2P,Change-Inc,0,144
GBR-2P,Change-St,144,1008
GBR-2P,Change-Dec,1152,144
GBR-2P,Change-Inc,1296,288
GBR-2P,Change-Dec,1584,144
GBR-2P,Change-Inc,1728,432
GBR-2P,Change-St,2160,720
GBR-2P,Change-Inc,2880,144

GBR-2P,Change-St,3024,144
GBR-2P,Change-Inc,3168,288
GBR-2P,Change-St,3456,432
GBR-2P,Change-Dec,3888,720
GBR-2P,Change-Inc,4608,144
GBR-2P,Change-Dec,4752,144
GBR-2P,Change-Inc,4896,288
GBR-2P,Change-Dec,5184,288
GBR-2P,Change-Inc,5472,432
GBR-2P,Change-St,5904,144
GBR-2P,Change-Dec,6048,288
GBR-2P,Change-Inc,6336,720
GBR-2P,Change-Dec,7056,144
GBR-2P,Change-Inc,7200,288
GRC-OA,Level-Low,0,2160
GRC-OA,Level-St,2160,1008
GRC-OA,Level-Low,3168,288
GRC-OA,Level-St,3456,144
GRC-OA,Level-Low,3600,1440
GRC-OA,Level-St,5040,144
GRC-OA,Level-Low,5184,576
GRC-OA,Level-St,5760,432
GRC-OA,Level-Low,6192,576
GRC-OA,Level-St,6768,144
GRC-OA,Level-Low,6912,576
GRC-OA,Change-Inc,0,144
GRC-OA,Change-St,144,1440
GRC-OA,Change-Dec,1584,144
GRC-OA,Change-St,1728,1008
GRC-OA,Change-Inc,2736,144
GRC-OA,Change-St,2880,432
GRC-OA,Change-Inc,3312,144
GRC-OA,Change-St,3456,144
GRC-OA,Change-Dec,3600,144
GRC-OA,Change-St,3744,3744
GVAR-OA,Level-Low,0,144
GVAR-OA,Level-St,144,1008
GVAR-OA,Level-Low,1152,144
GVAR-OA,Level-St,1296,1872
GVAR-OA,Level-Low,3168,144
GVAR-OA,Level-St,3312,2160
GVAR-OA,Level-Low,5472,144
GVAR-OA,Level-St,5616,1872
GVAR-OA,Change-St,0,7488
GL-OA,Level-Low,0,7488
GL-OA,Change-Inc,0,144
GL-OA,Change-Dec,144,144
GL-OA,Change-St,288,864
GL-OA,Change-Inc,1152,288
GL-OA,Change-St,1440,6048
GL-OA,Step-Inc,1184,0
LPABRT-OA,Level-St,0,7488
LPABRT-OA,Change-St,0,7488
LPBBFT-OA,Level-St,0,7488
LPBBFT-OA,Change-St,0,7488
LPBBRT-OA,Level-St,0,7488
LPBBRT-OA,Change-St,0,7488
GBFT-OA,Level-St,0,7488
GBFT-OA,Change-St,0,7488
GBRT-OA,Level-St,0,7488
GBRT-OA,Change-St,0,74

Appendix D. Looseness Fault Beran Data

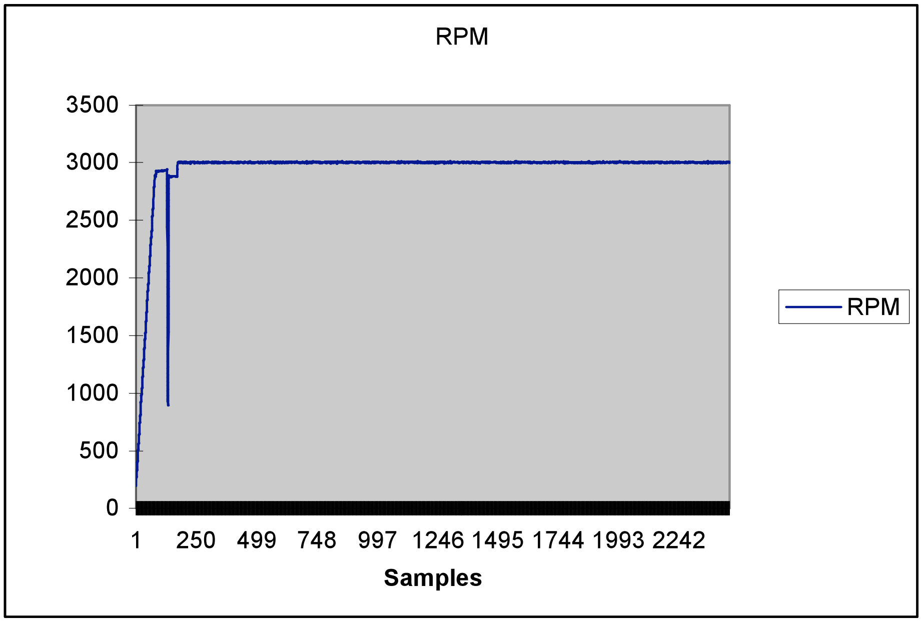


Figure D.35: Turbine 2, 14/08/2006 – 29/08/2006, RPM Data

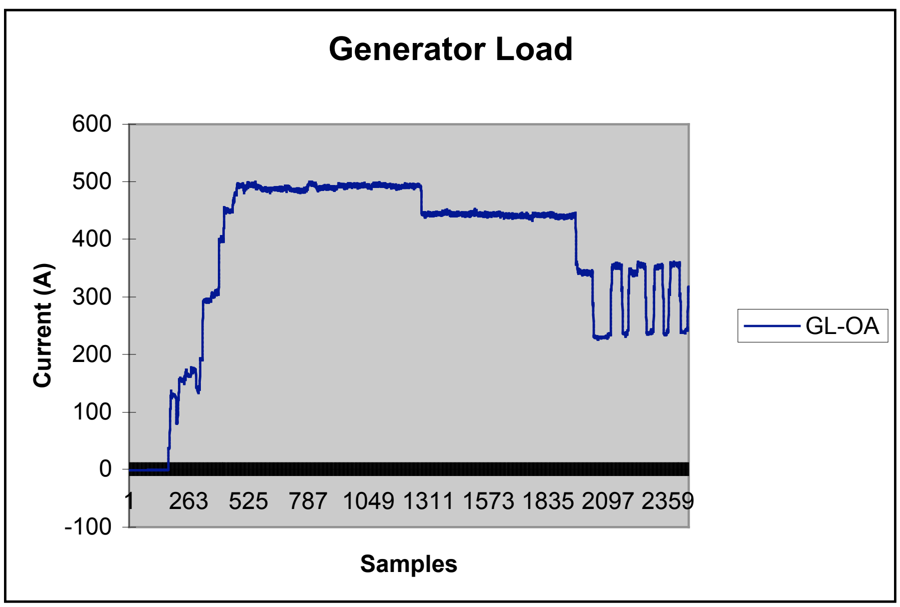


Figure D.36: Turbine 2, 14/08/2006 – 29/08/2006, Generator Load Data

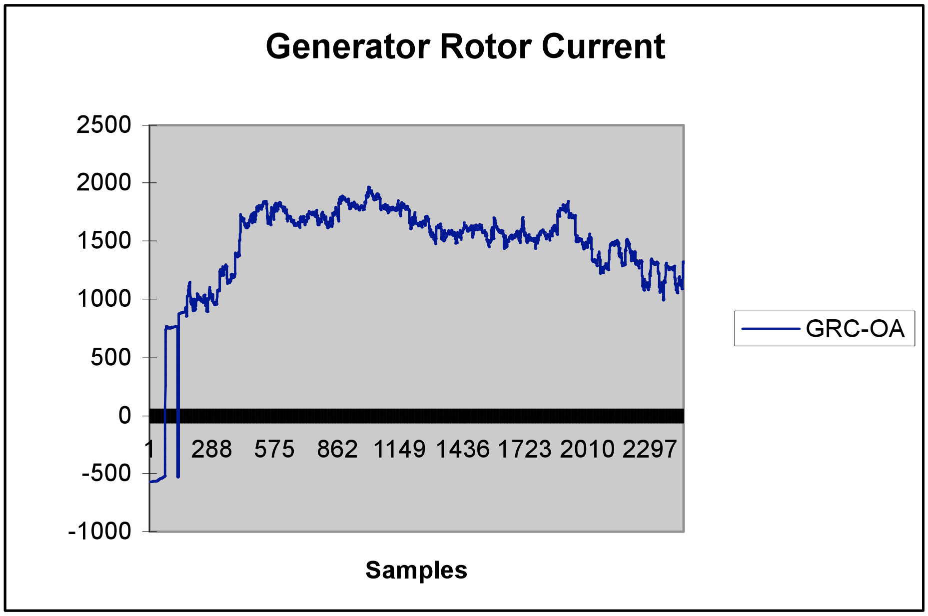


Figure D.37: Turbine 2, 14/08/2006 – 29/08/2006, Generator Rotor Current Data

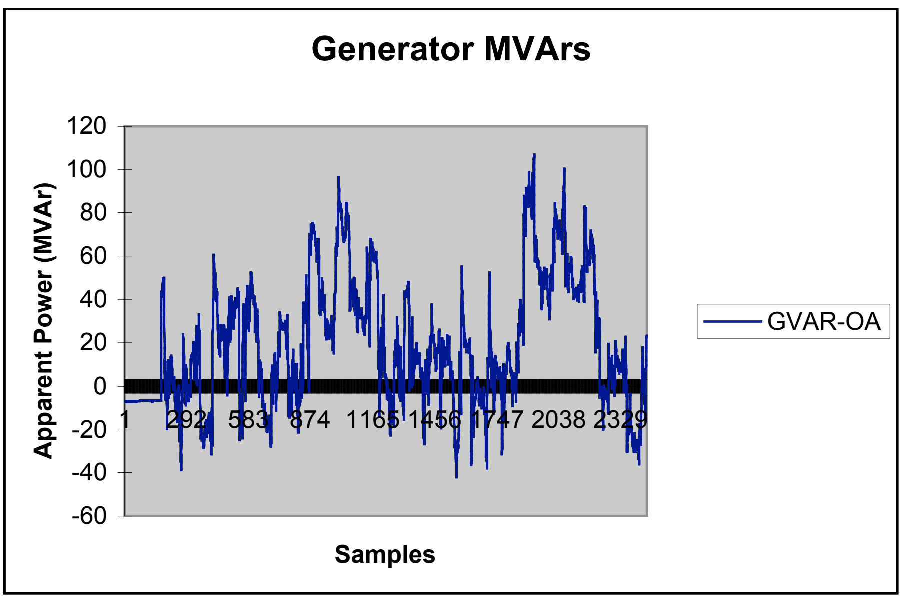


Figure D.38: Turbine 2, 14/08/2006 – 29/08/2006, Generator MVAr Data

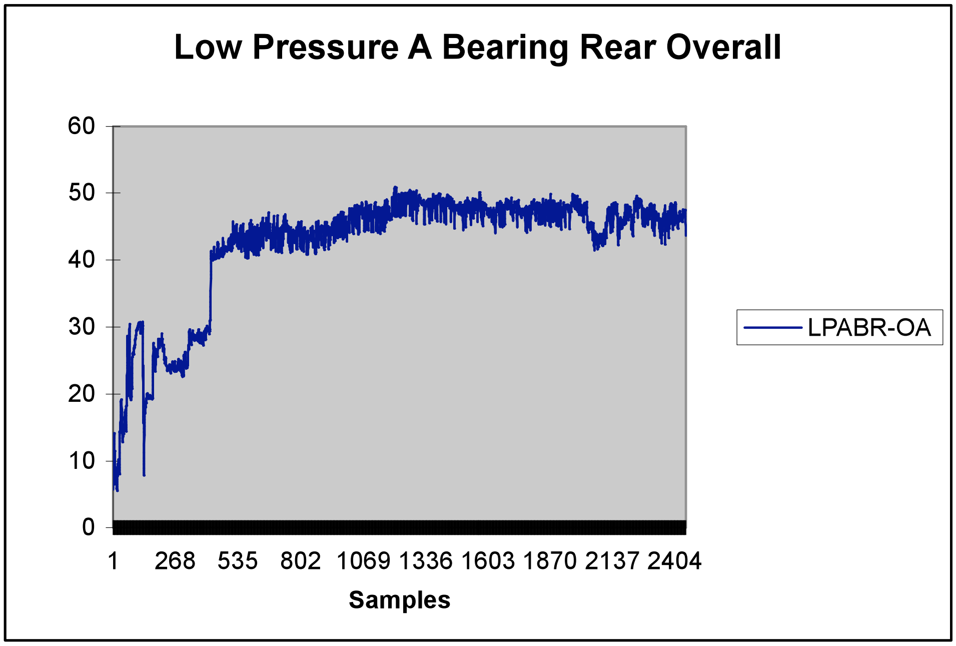


Figure D.39: Turbine 2, channel 6, 14/08/2006 – 29/08/2006, Low Pressure A Bearing Rear Overall Amplitude Data

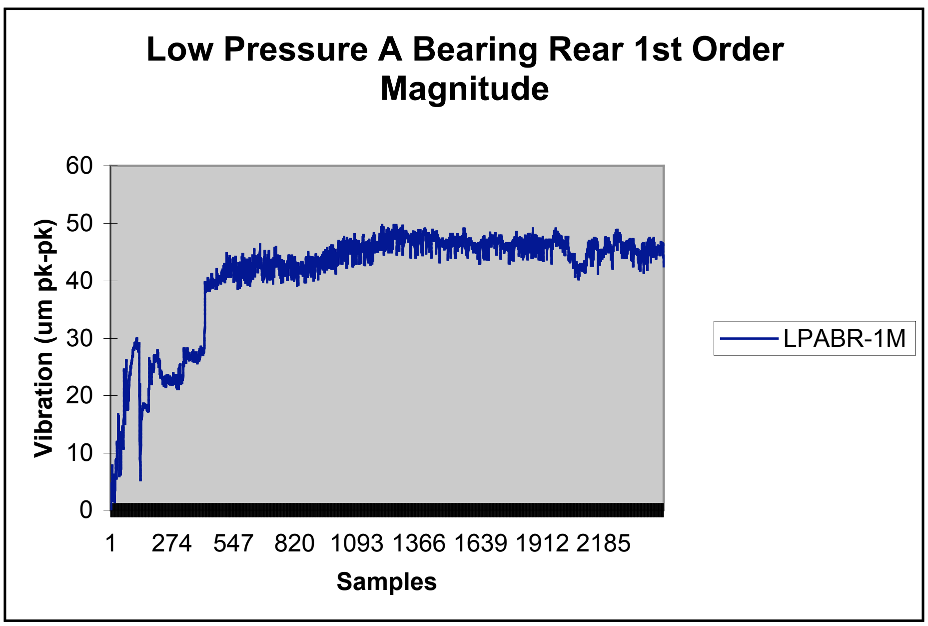


Figure D.40: Turbine 2, channel 6, 14/08/2006 – 29/08/2006, Low Pressure A Bearing Rear 1st Order Magnitude Data

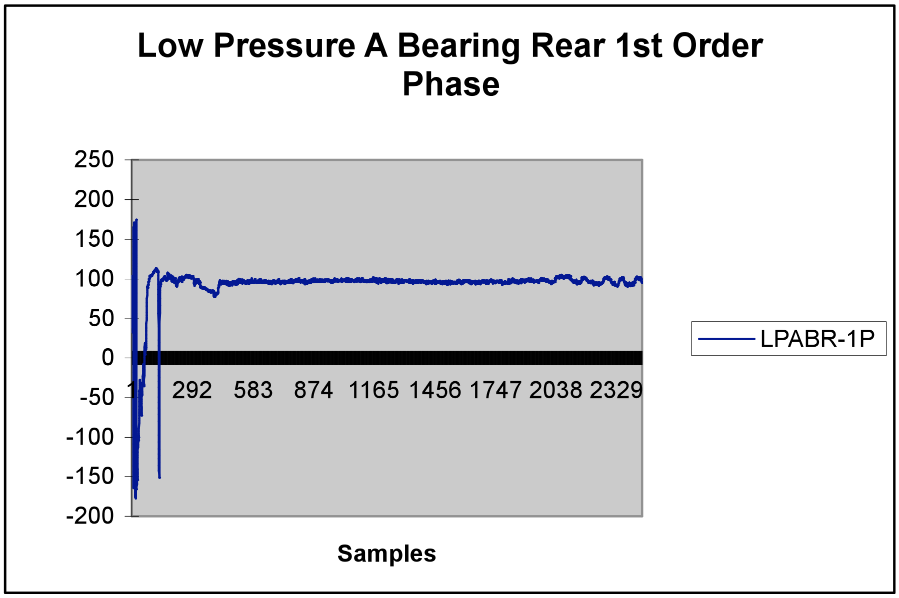


Figure D.41: Turbine 2, channel 6, 14/08/2006 – 29/08/2006, Low Pressure A Bearing Rear 1st Order Phase Data

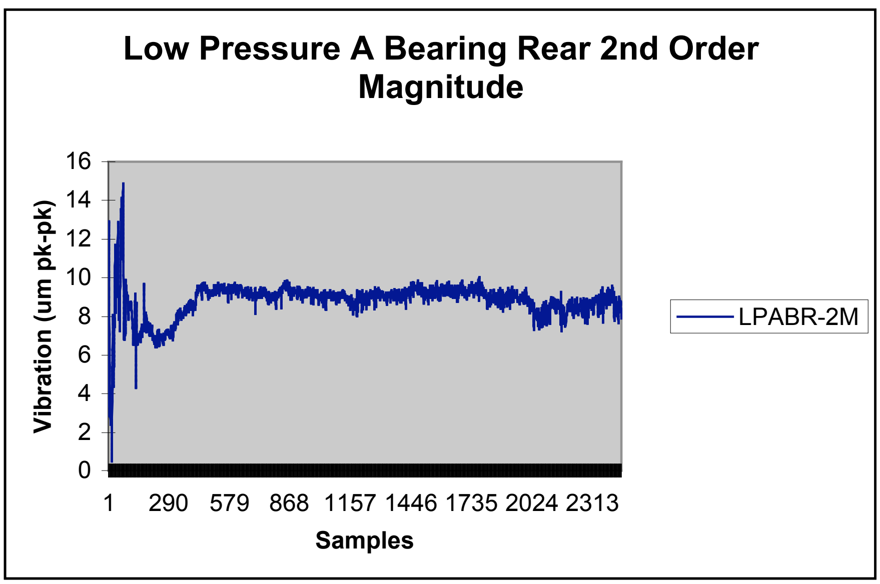


Figure D.42: Turbine 2, channel 6, 14/08/2006 – 29/08/2006, Low Pressure A Bearing Rear 2nd Order Magnitude Data

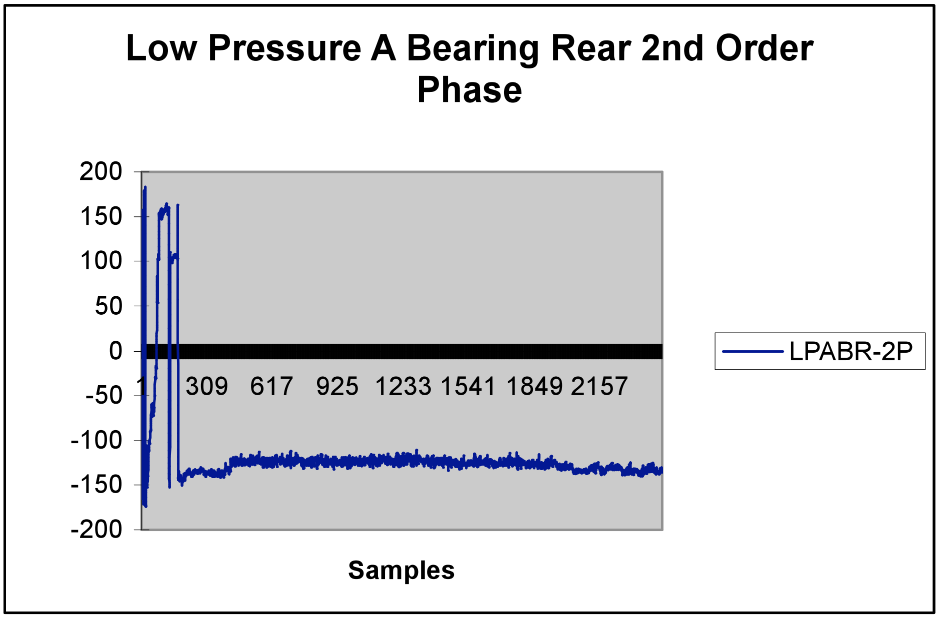


Figure D.43: Turbine 2, channel 6, 14/08/2006 – 29/08/2006, Low Pressure A Bearing Rear 2nd Order Phase Data

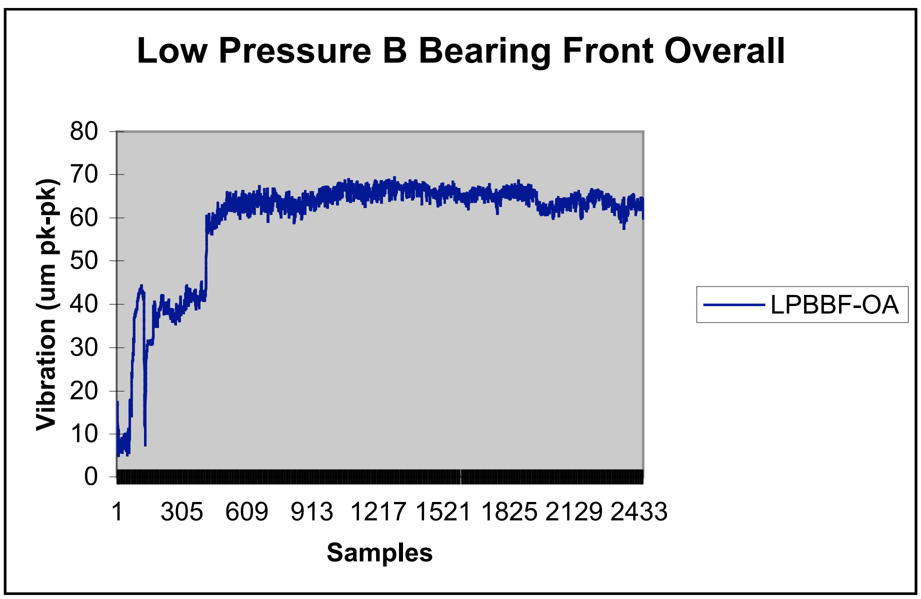


Figure D.44: Turbine 2, channel 7, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Front Overall Amplitude Data

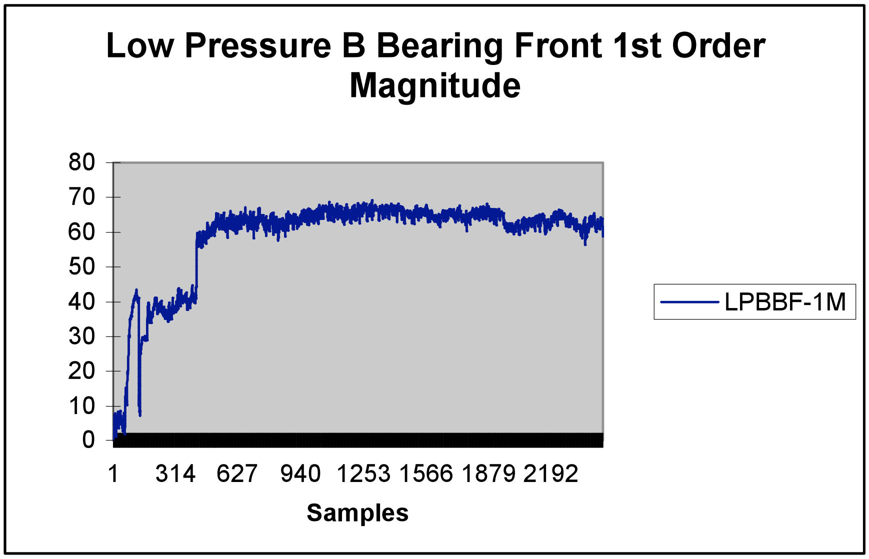


Figure D.45: Turbine 2, channel 7, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Front 1st Order Magnitude Data

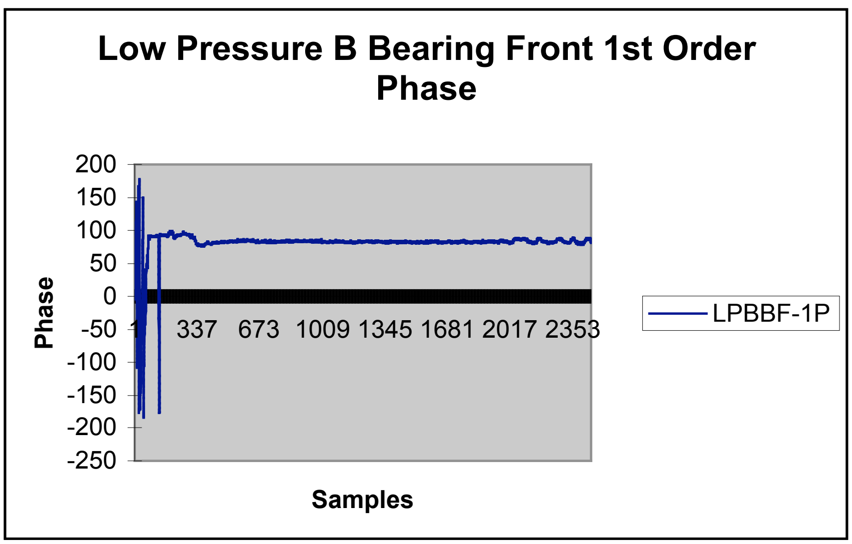


Figure D.46: Turbine 2, channel 7, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Front 1st Order Phase Data

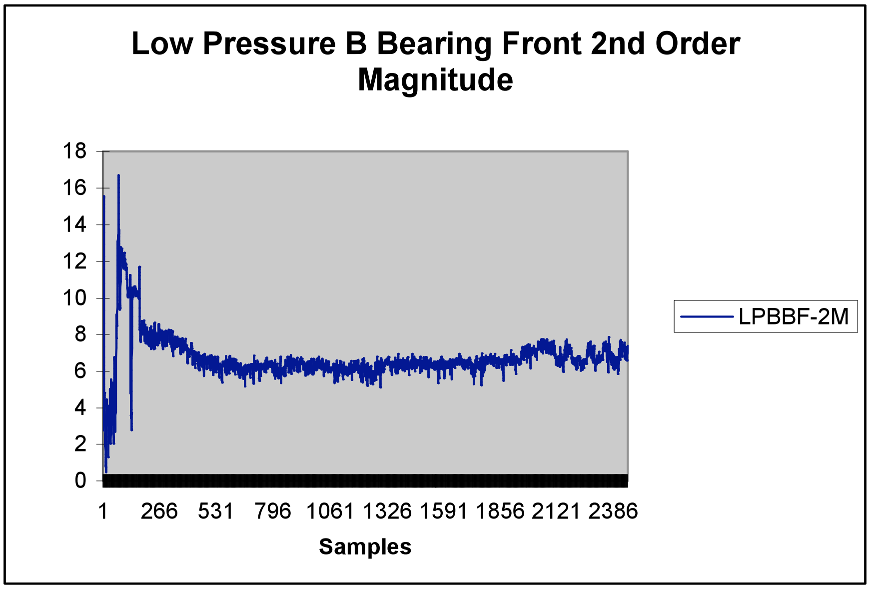


Figure D.47: Turbine 2, channel 7, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Front 2nd Order Magnitude Data

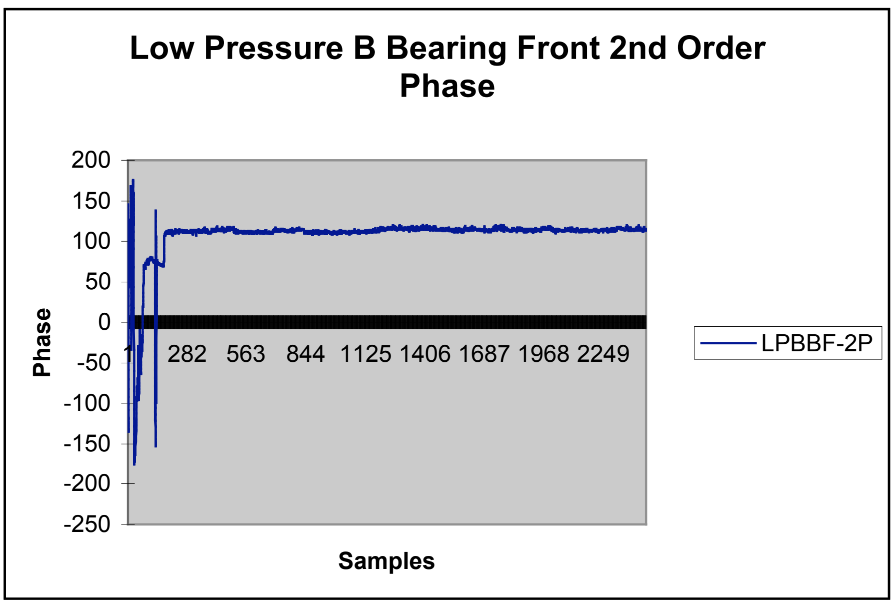


Figure D.48: Turbine 2, channel 7, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Front 2nd Order Phase Data

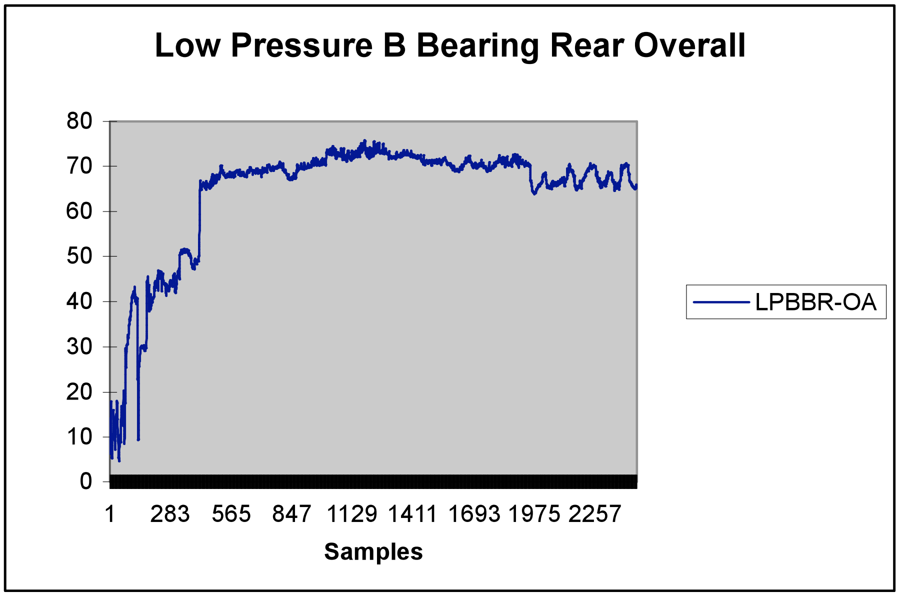


Figure D.49: Turbine 2, channel 8, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Rear Overall Amplitude Data

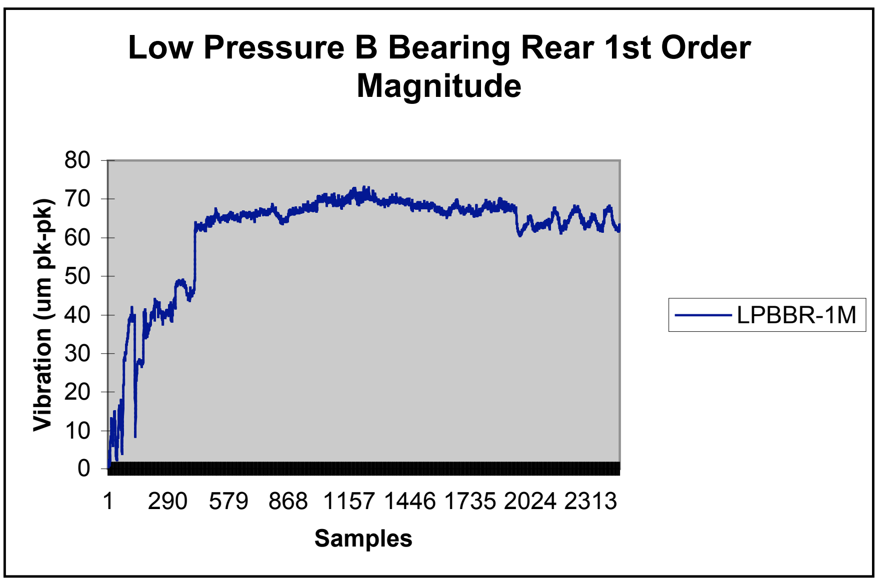


Figure D.50: Turbine 2, channel 8, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Rear 1st Order Magnitude Data

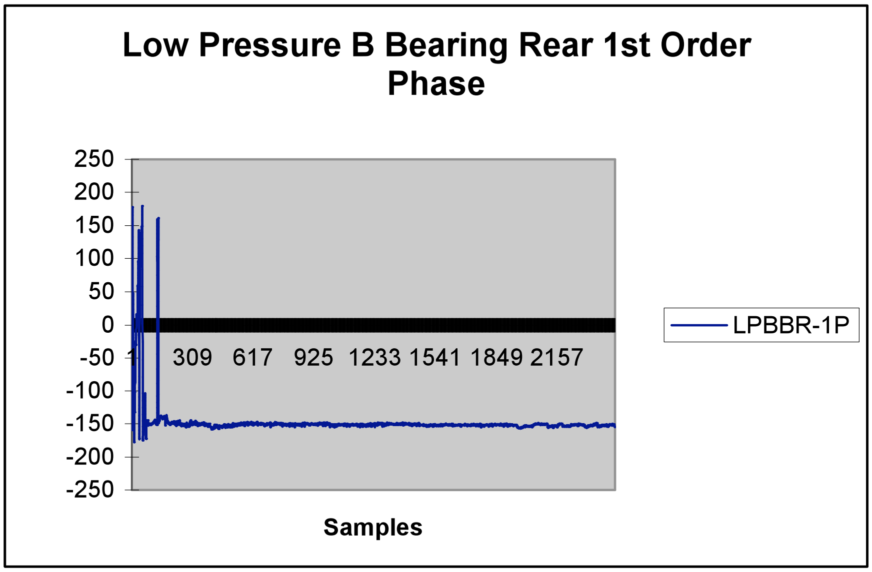


Figure D.51: Turbine 2, channel 8, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Rear 1st Order Phase Data

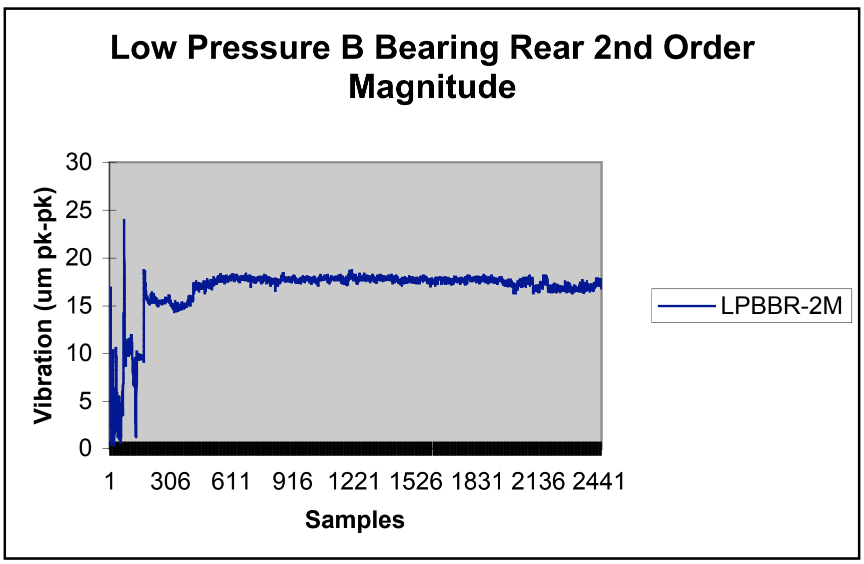


Figure D.52: Turbine 2, channel 8, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Rear 2nd Order Magnitude Data

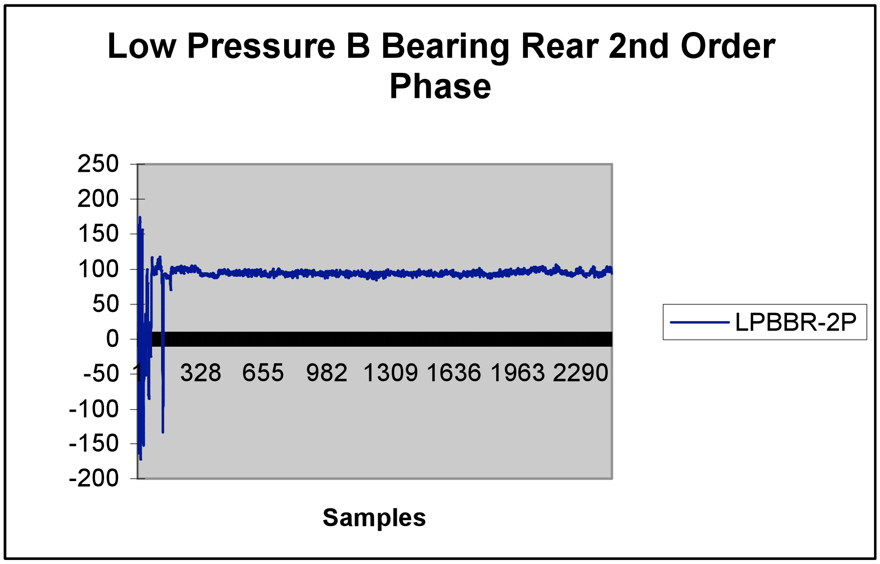


Figure D.53: Turbine 2, channel 8, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Rear 2nd Order Phase Data

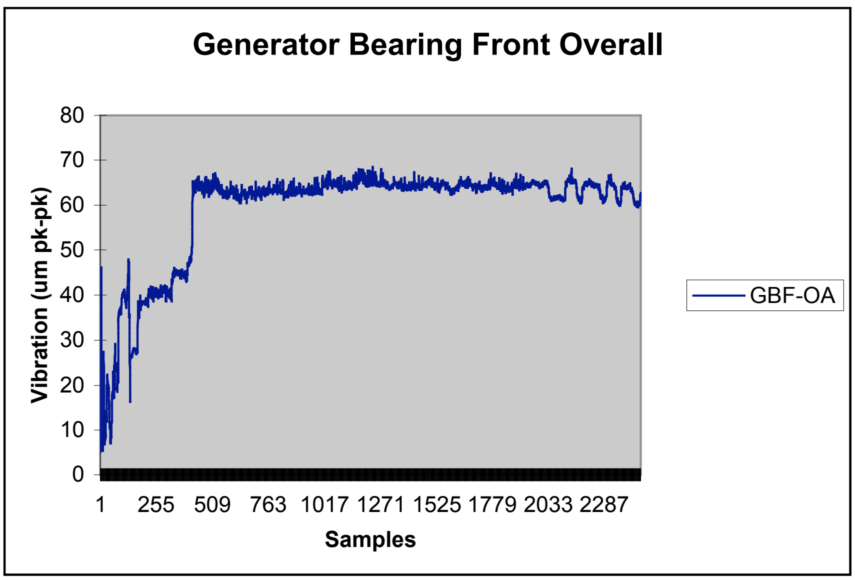


Figure D.54: Turbine 2, channel 9, 14/08/2006 – 29/08/2006, Generator Bearing Front Overall Amplitude Data

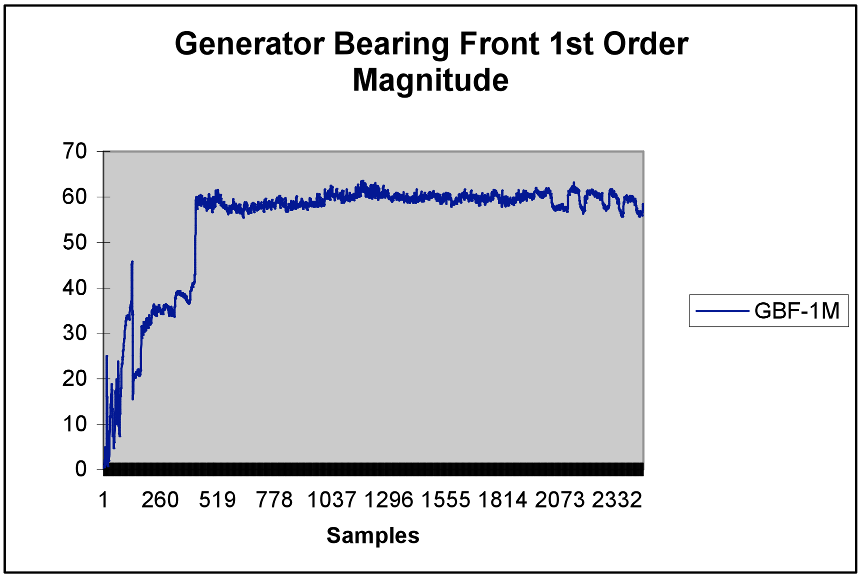


Figure D.55: Turbine 2, channel 9, 14/08/2006 – 29/08/2006, Generator Bearing Front 1st Order Magnitude Data

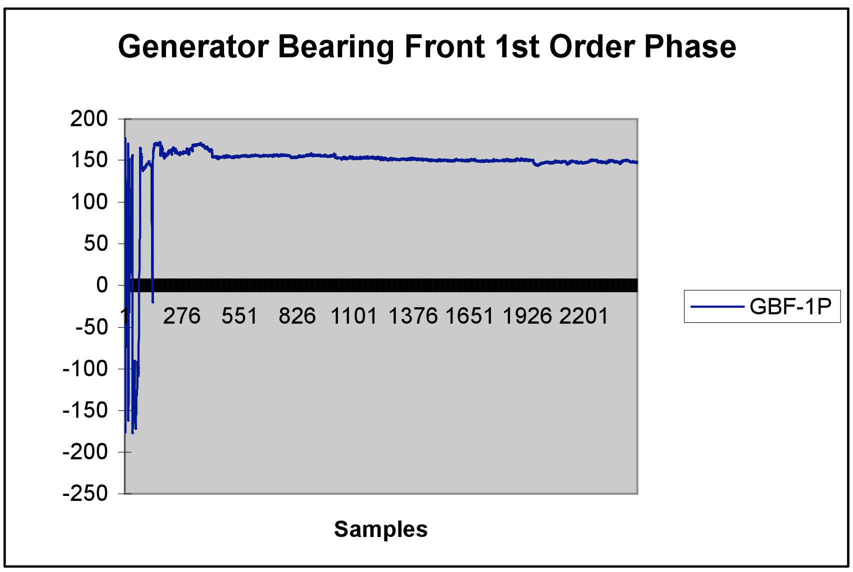


Figure D.56: Turbine 2, channel 9, 14/08/2006 – 29/08/2006, Generator Bearing Front 1st Order Phase Data

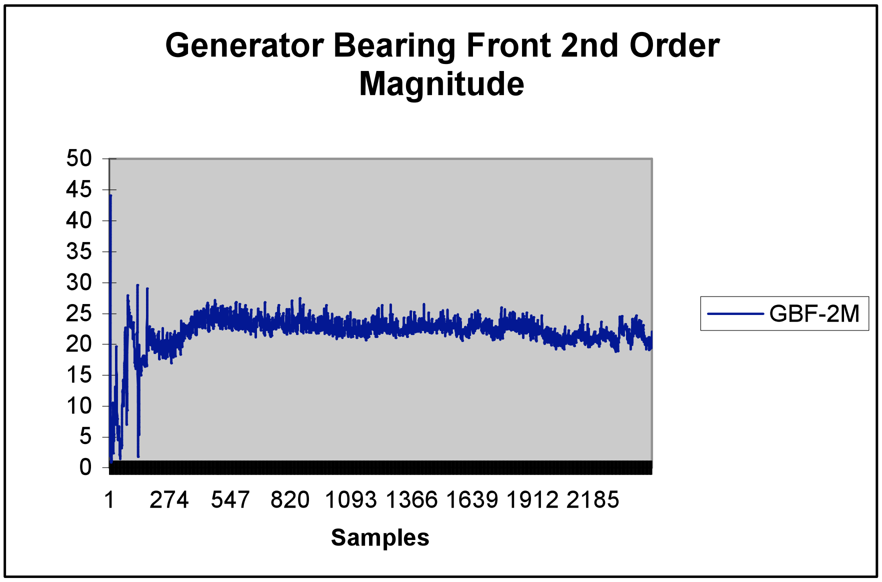


Figure D.57: Turbine 2, channel 9, 14/08/2006 – 29/08/2006, Generator Bearing Front 2nd Order Magnitude Data

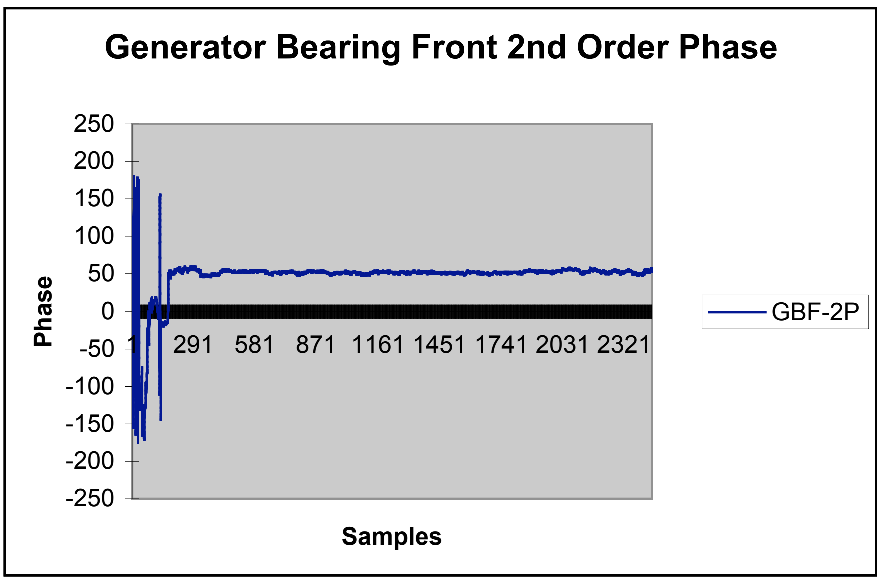


Figure D.58: Turbine 2, channel 9, 14/08/2006 – 29/08/2006, Generator Bearing Front 2nd Order Phase Data

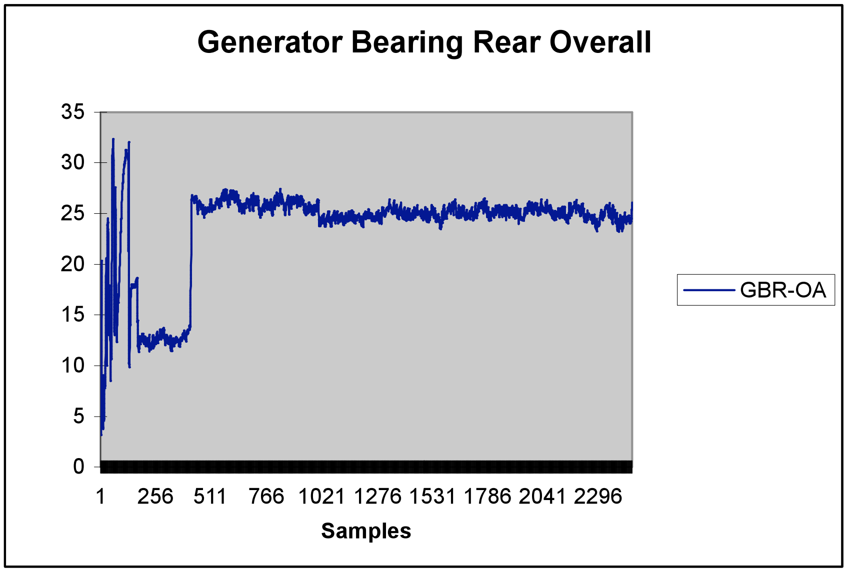


Figure D.59: Turbine 2, channel 10, 14/08/2006 – 29/08/2006, Generator Bearing Rear Overall Amplitude Data

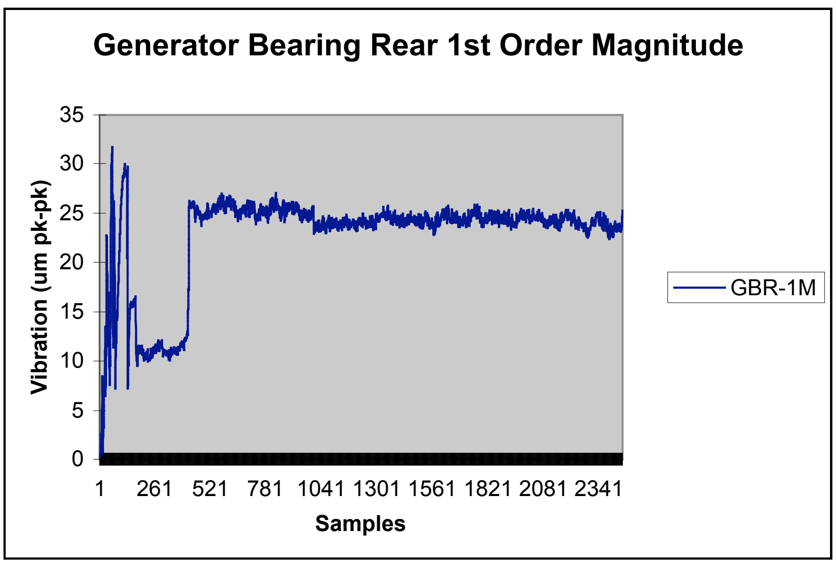


Figure D.60: Turbine 2, channel 10, 14/08/2006 – 29/08/2006, Generator Bearing Rear 1st Order Magnitude Data

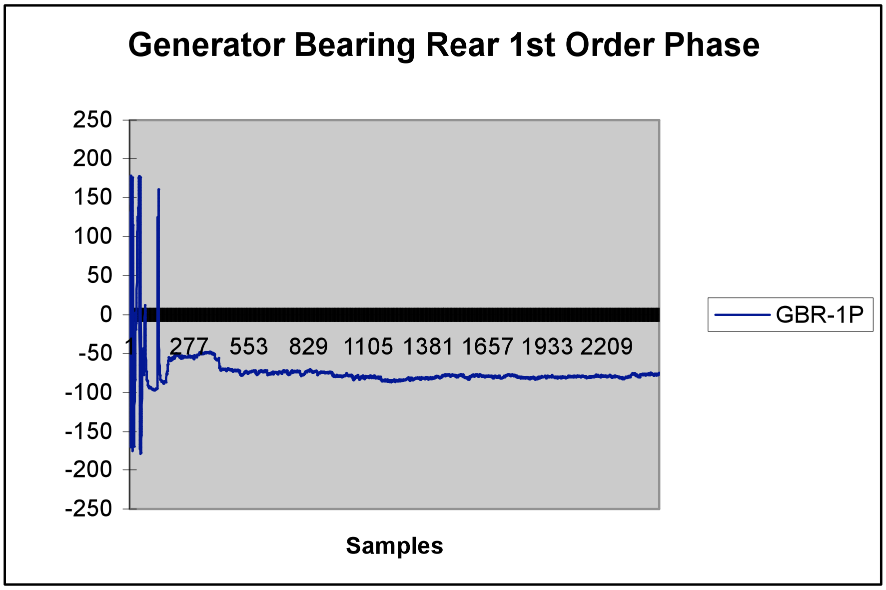


Figure D.61: Turbine 2, channel 10, 14/08/2006 – 29/08/2006, Generator Bearing Rear 1st Order Phase Data

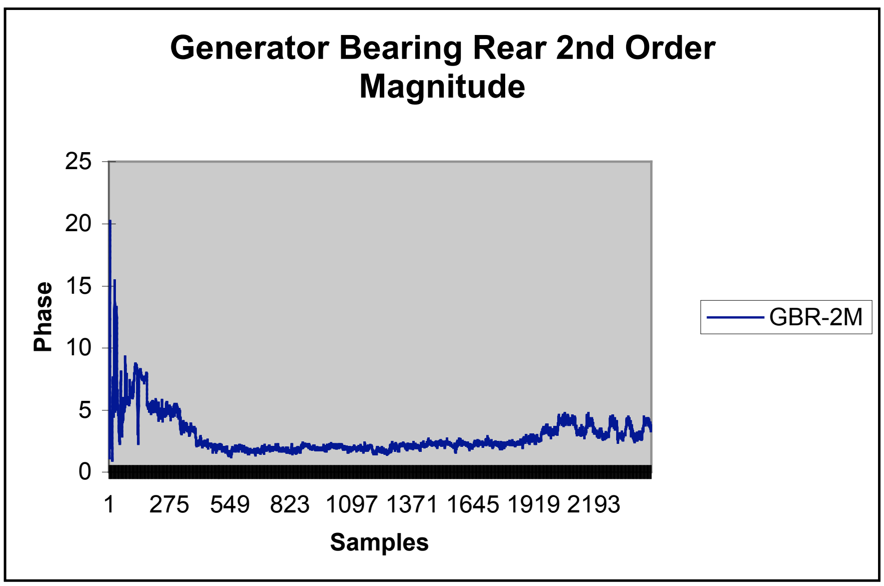


Figure D.62: Turbine 2, channel 10, 14/08/2006 – 29/08/2006, Generator Bearing Rear 2nd Order Magnitude Data

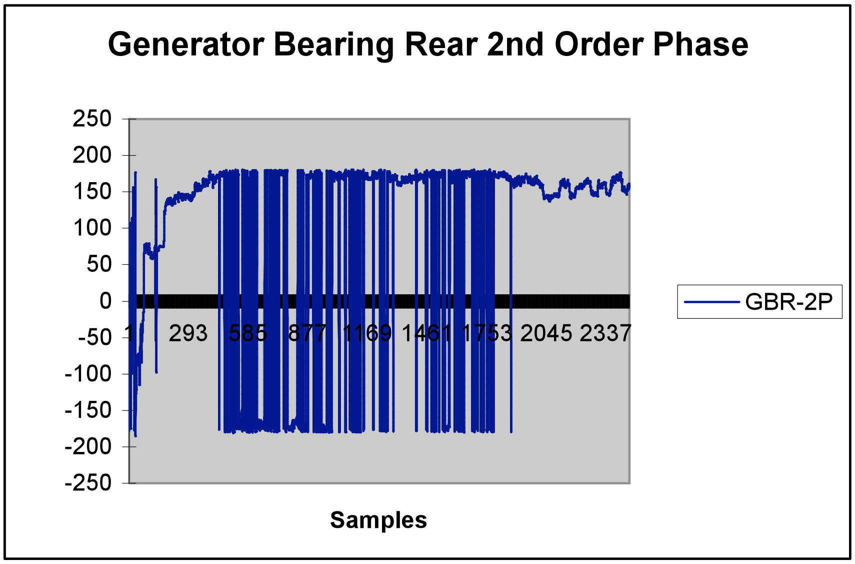


Figure D.63: Turbine 2, channel 10, 14/08/2006 – 29/08/2006, Generator Bearing Rear 2nd Order Phase Data

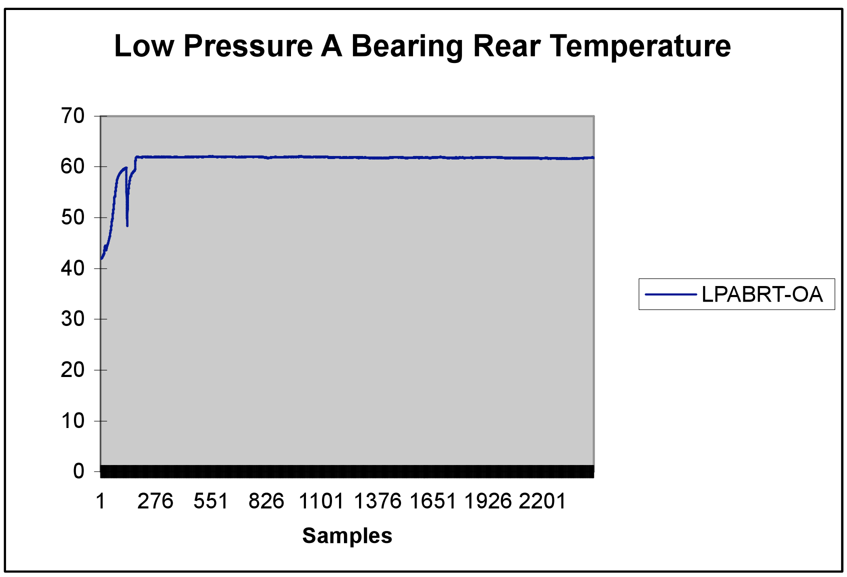


Figure D.64: Turbine 2, channel 6, 14/08/2006 – 29/08/2006, Low Pressure A Bearing Rear Temperature Data

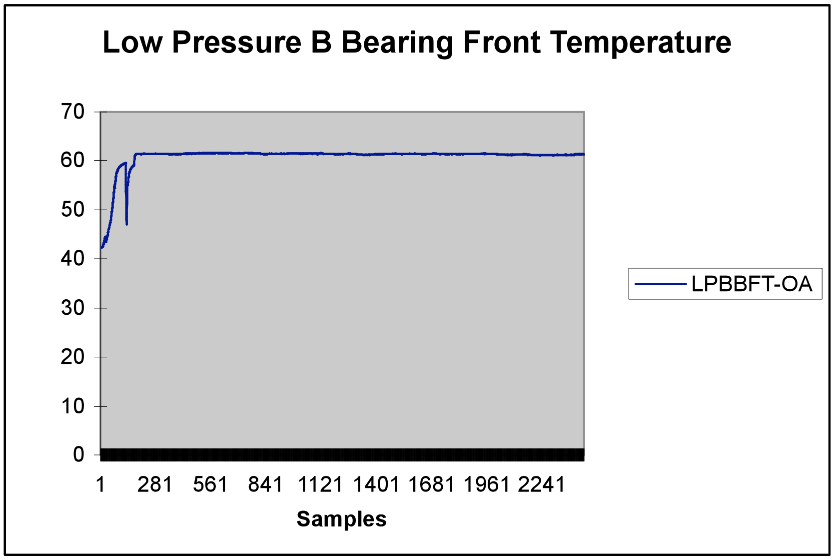


Figure D.65: Turbine 2, channel 7, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Front Temperature Data

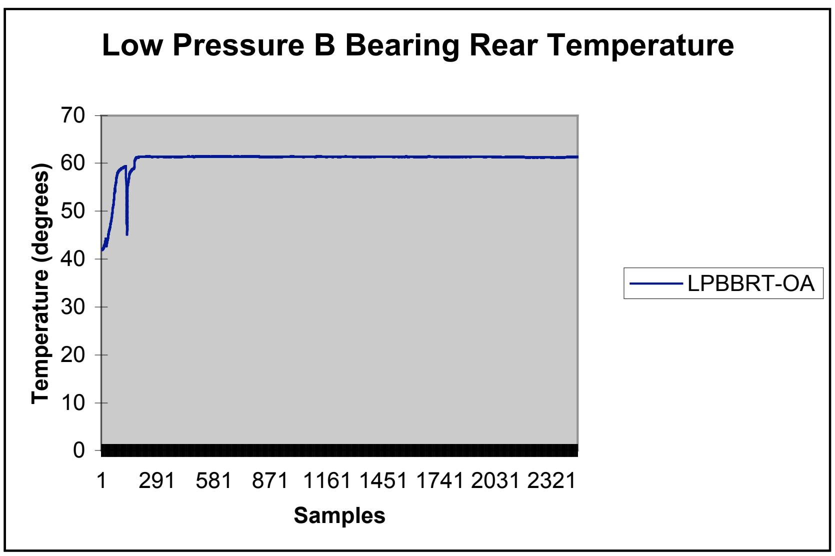


Figure D.66: Turbine 2, channel 8, 14/08/2006 – 29/08/2006, Low Pressure B Bearing Rear Temperature Data

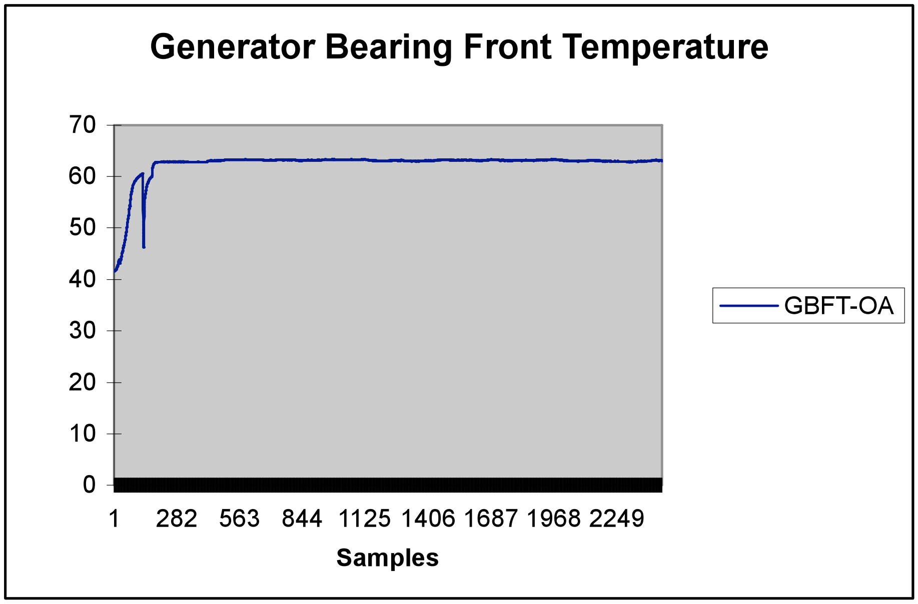


Figure D.67: Turbine 2, channel 9, 14/08/2006 – 29/08/2006, Generator Bearing Front Temperature Data

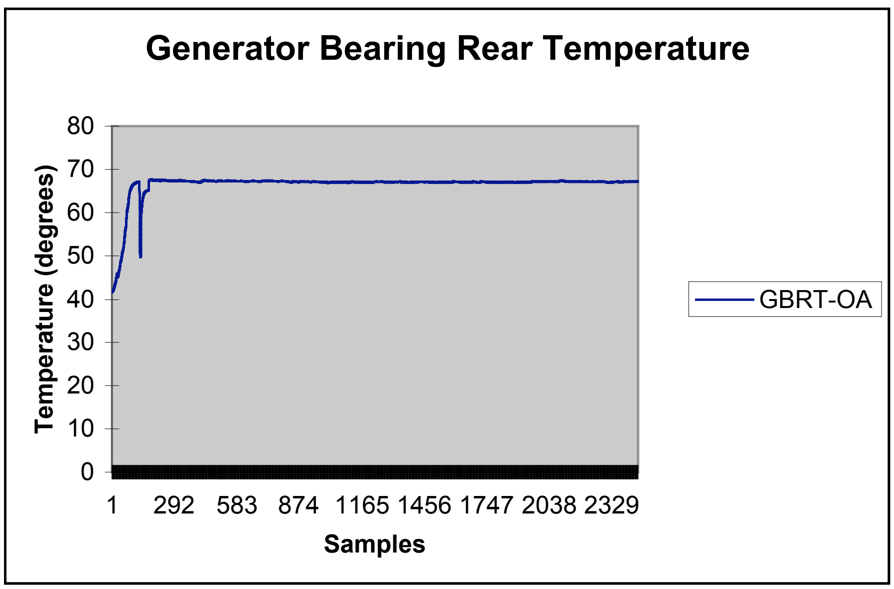


Figure D.68: Turbine 2, channel 10, 14/08/2006 – 29/08/2006, Generator Bearing Rear Temperature Data

Channel Profile of Looseness Fault Data

oosegpf,,,
LPABR-OA,Level-Low,0,432
LPABR-OA,Level-St,432,2016
LPABR-OA,Change-Inc,0,144
LPABR-OA,Change-St,144,144
LPABR-OA,Change-Inc,288,144
LPABR-OA,Change-St,432,2016
LPABR-1M,Level-Low,0,432
LPABR-1M,Level-St,432,2016
LPABR-1M,Change-Inc,0,144
LPABR-1M,Change-St,144,144
LPABR-1M,Change-Inc,288,144
LPABR-1M,Change-St,432,2016
LPABR-1P,Level-St,0,2448
LPABR-1P,Change-Dec,0,144
LPABR-1P,Change-St,144,2304
LPABR-2M,Level-St,0,2448
LPABR-2M,Change-Inc,0,144
LPABR-2M,Change-St,144,144
LPABR-2M,Change-Inc,288,144
LPABR-2M,Change-St,432,2016
LPABR-2P,Level-Hi,0,144
LPABR-2P,Level-St,144,2304
LPABR-2P,Change-Inc,0,144
LPABR-2P,Change-Dec,144,144
LPABR-2P,Change-St,288,2160
LPBBF-OA,Level-Low,0,144
LPBBF-OA,Level-St,144,288
LPBBF-OA,Level-Hi,432,2016
LPBBF-OA,Change-Inc,0,432
LPBBF-OA,Change-St,432,2016
LPBBF-1M,Level-Low,0,144
LPBBF-1M,Level-St,144,288
LPBBF-1M,Level-Hi,432,2016
LPBBF-1M,Change-Inc,0,432
LPBBF-1M,Change-St,432,1440
LPBBF-1M,Change-Dec,1872,144
LPBBF-1M,Change-St,2016,432
LPBBF-1P,Level-St,0,2448
LPBBF-1P,Change-Dec,0,144
LPBBF-1P,Change-St,144,2304
LPBBF-2M,Level-St,0,2448
LPBBF-2M,Change-Inc,0,144
LPBBF-2M,Change-Dec,144,288
LPBBF-2M,Change-St,432,2016
LPBBF-2P,Level-Low,0,144
LPBBF-2P,Level-St,144,2304
LPBBF-2P,Change-St,0,144
LPBBF-2P,Change-Inc,144,144
LPBBF-2P,Change-St,288,2160
LPBBR-OA,Level-Low,0,144
LPBBR-OA,Level-St,144,288
LPBBR-OA,Level-Hi,432,2016
LPBBR-OA,Change-Inc,0,432
LPBBR-OA,Change-St,432,1440
LPBBR-OA,Change-Dec,1872,144
LPBBR-OA,Change-St,2016,432
LPBBR-1M,Level-Low,0,144
LPBBR-1M,Level-St,144,288
LPBBR-1M,Level-Hi,432,2016
LPBBR-1M,Change-Inc,0,432
LPBBR-1M,Change-St,432,1440
LPBBR-1M,Change-Dec,1872,144
LPBBR-1M,Change-St,2016,432
LPBBR-1P,Level-Hi,0,144
LPBBR-1P,Level-St,144,2304
LPBBR-1P,Change-Dec,0,144
LPBBR-1P,Change-St,144,2304
LPBBR-2M,Level-St,0,288
LPBBR-2M,Level-Hi,288,2160
LPBBR-2M,Change-Inc,0,432
LPBBR-2M,Change-St,432,2016
LPBBR-2P,Level-Low,0,144
LPBBR-2P,Level-St,144,2304
LPBBR-2P,Change-St,0,2448
GBF-OA,Level-Low,0,144
GBF-OA,Level-St,144,288
GBF-OA,Level-Hi,432,2016
GBF-OA,Change-Inc,0,432
GBF-OA,Change-St,432,1872
GBF-OA,Change-Dec,2304,144
GBF-1M,Level-Low,0,144
GBF-1M,Level-St,144,864
GBF-1M,Level-Hi,1008,288
GBF-1M,Level-St,1296,576
GBF-1M,Level-Hi,1872,144
GBF-1M,Level-St,2016,432
GBF-1M,Change-Inc,0,432
GBF-1M,Change-St,432,2016
GBF-1P,Level-Low,0,144
GBF-1P,Level-St,144,2304
GBF-1P,Change-Inc,0,144
GBF-1P,Change-St,144,2304
GBF-2M,Level-St,0,144
GBF-2M,Level-Hi,144,2304
GBF-2M,Change-Inc,0,144
GBF-2M,Change-St,144,144
GBF-2M,Change-Inc,288,144
GBF-2M,Change-St,432,2016
GBF-2P,Level-Low,0,144
GBF-2P,Level-St,144,2304
GBF-2P,Change-St,0,144
GBF-2P,Change-Inc,144,144
GBF-2P,Change-St,288,2160

RPM-OA,Level-Low,0,288
RPM-OA,Level-St,288,2160
RPM-OA,Change-Inc,0,288
RPM-OA,Change-St,288,2160
GBR-OA,Level-Low,0,2448
GBR-OA,Change-Inc,0,144
GBR-OA,Change-St,144,144
GBR-OA,Change-Inc,288,144
GBR-OA,Change-St,432,2016
GBR-1M,Level-Low,0,2448
GBR-1M,Change-Inc,0,144
GBR-1M,Change-St,144,144
GBR-1M,Change-Inc,288,144
GBR-1M,Change-St,432,2016
GBR-1P,Level-St,0,2448
GBR-1P,Change-Dec,0,144
GBR-1P,Change-Inc,144,144
GBR-1P,Change-Dec,288,144
GBR-1P,Change-St,432,2016
GBR-2M,Level-St,0,288
GBR-2M,Level-Low,288,2160
GBR-2M,Change-Inc,0,144
GBR-2M,Change-Dec,144,288
GBR-2M,Change-St,432,1872
GBR-2M,Change-Inc,2304,144
GBR-2P,Level-St,0,2448
GBR-2P,Change-Inc,0,432
GBR-2P,Change-Dec,432,144
GBR-2P,Change-Inc,576,144
GBR-2P,Change-Dec,720,144
GBR-2P,Change-Inc,864,144
GBR-2P,Change-Dec,1008,144
GBR-2P,Change-Inc,1152,288
GBR-2P,Change-Dec,1440,144
GBR-2P,Change-Inc,1584,432
GBR-2P,Change-St,2016,432
GRC-OA,Level-Low,0,2448
GRC-OA,Change-Inc,0,144
GRC-OA,Change-St,144,144
GRC-OA,Change-Inc,288,144
GRC-OA,Change-St,432,720
GRC-OA,Change-Dec,1152,144
GRC-OA,Change-St,1296,576
GRC-OA,Change-Dec,1872,144
GRC-OA,Change-St,2016,432
GVAR-OA,Level-Low,0,432
GVAR-OA,Level-St,432,1008
GVAR-OA,Level-Low,1440,432
GVAR-OA,Level-St,1872,432
GVAR-OA,Level-Low,2304,144
GVAR-OA,Change-St,0,2448
GL-OA,Level-Low,0,2448
GL-OA,Change-St,0,144
GL-OA,Change-Inc,144,432
GL-OA,Change-St,576,576
GL-OA,Change-Dec,1152,144
GL-OA,Change-St,1296,576
GL-OA,Change-Dec,1872,144
GL-OA,Change-St,2016,144
GL-OA,Change-Inc,2160,144
GL-OA,Change-Dec,2304,144
GL-OA,Step-Inc,320,0
GL-OA,Step-Dec,1955,0
LPABRT-OA,Level-St,0,2448
LPABRT-OA,Change-Inc,0,144
LPABRT-OA,Change-St,144,2304
LPBBFT-OA,Level-St,0,2448
LPBBFT-OA,Change-Inc,0,144
LPBBFT-OA,Change-St,144,2304
LPBBRT-OA,Level-St,0,2448
LPBBRT-OA,Change-Inc,0,144
LPBBRT-OA,Change-St,144,2304
GBFT-OA,Level-St,0,2448
GBFT-OA,Change-Inc,0,144
GBFT-OA,Change-St,144,2304
GBRT-OA,Level-St,0,2448
GBRT-OA,Change-Inc,0,144
GBRT-OA,Change-St,14