Text Mining and Natural Language Processing for the Early Stages of Space Mission Design

Audrey Berquand

Submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

Intelligent Computational Engineering Laboratory
Department of Mechanical & Aerospace Engineering
University of Strathclyde, Glasgow

December 2021

Acknowledgements

I would first like to thank my supervisor Dr. Annalisa Riccardi for her continuous support, precious guidance, and strong leadership. Thank you for trusting me with this project, it was a pleasure working with you and I look forward to our future collaborations.

I am grateful for the funding provided by the European Space Agency through the Networking Partnering Initiative (NPI) program. I am especially thankful for the valuable collaboration with the Concurrent Design Facility team, their support in organising the expert survey was key in shaping the objectives of this thesis. I warmly thank Tiago Soares who supervised the project on ESA side. I am also grateful for the support provided by the NPI industrial partners represented by Sam Gerené (RHEA Group), Kartik Kumar (satsearch) and Norbert Brauer (Airbus).

Additionally, I thank the Space Generation Advisory Council and the Women In Aerospace Europe networks for supporting my participation in conferences. Through these networks, I made trailblazing friends who have been a constant source of inspiration, including among others, my co-National Point of Contact Dr. Florian Marmuse.

I warmly thank Sam Gerené, Martin Fijneman, Arne Matthyssen, Miranda Labella, and Alex Vorobiev for welcoming me at their RHEA office in Leiden for a four-month internship.

I am grateful to Dr. Yashar Moshfeghi for his valuable contributions on the Topic Modelling work, which led to my first journal publication. I would also like to thank Paul Darm, it was a real pleasure tuning the SpaceTransformers models with you.

I thank Prof. Massimiliano Vasile for the opportunity to lead two internal concurrent design studies at the University of Strathclyde. I had a great time organising those with Dr. Andrew Wilson. I warmly thank the students from the Mechanical and Aerospace Engineering department and from the StrathAIS society who participated in the design challenges.

I wish to acknowledge Sabrina Mirtcheva from ESA with whom I had constructive discussions on Natural Language Processing and text mining. I warmly thank Serge Valera and Luis Mansilla, also from ESA, for sharing their knowledge on information modelling and for their guidance with the Engineering Models Knowledge Graph work.

Throughout my PhD within the ICE lab, I encountered many passionate and wonderful people. I would like to thank Dr. Edmondo Minisci for his leadership of the ICE lab, and for the occasional limoncello tasting. I am especially grateful to Dr. Mateusz Polnik who often advised me, your high work ethic and academic approach were inspiring. I thank Robin Kamenicky for our numerous discussions, and for organising with his partner, Liubov Subbotina, an unforgettable expedition in Glen Etive. I would also like to acknowledge Francesco Marchetti, Callum Wilson, Deep Bandivadekar, Luis Sanchez Fernandez-Mellado, Beatriz Casares, Catarina Gomes, Cheyenne Powell, Cristian Greco, Iain McDonald and fellow students I had the chance to meet.

I would like to thank my parents, Delphine and Jean-Paul, for always supporting my career choices, and all my family for their continuous encouragement. I also lovingly remember my mamie and papi, Nicole and Paul, who after numerous times asking me if I was done with "*compter les étoiles*", have decided to go ahead and perform in situ measurements themselves.

Although she will never read this, I would like to acknowledge Nessie (my cat, not the Loch Ness monster) who has been acting as an unofficial emotional support animal throughout this thesis writing.

Finally, I am forever grateful to my fiancé, Ciarán Toner, for supporting me during a few PhD mental breakdowns, for exploring Scotland together by foot and by bike, for being a perfect home-office co-worker during the COVID-19 pandemic, and for fuelling me with delicious curries, cheesecakes and happiness.

“The Answer to the Great Question...

Of Life, the Universe and Everything...

Is... Forty-two,’

said Deep Thought, with infinite majesty and calm.”

Douglas Adams, The Hitchhiker’s Guide to the Galaxy

Abstract

A considerable amount of data related to space mission design has been accumulated since artificial satellites started to venture into space in the 1950s. This data has today become an overwhelming volume of information, triggering a significant knowledge reuse bottleneck at the early stages of space mission design. Meanwhile, virtual assistants, text mining and Natural Language Processing techniques have become pervasive to our daily life.

The work presented in this thesis is one of the first attempts to bridge the gap between the worlds of space systems engineering and text mining. Several novel models are thus developed and implemented here, targeting the structuring of accumulated data through an ontology, but also tasks commonly performed by systems engineers such as requirement management and heritage analysis. A first collection of documents related to space systems is gathered for the training of these methods. Eventually, this work aims to pave the way towards the development of a Design Engineering Assistant (DEA) for the early stages of space mission design. It is also hoped that this work will actively contribute to the integration of text mining and Natural Language Processing methods in the field of space mission design, enhancing current design processes.

Contents

List of Figures

List of Tables

List of Abbreviations

AGI	Artificial General Intelligence
AHP	Analytic Hierarchical Process
AI	Artificial Intelligence
AIT	Assembly, Integration and Test
AIV	Assembly, Integration and Verification
ANN	Artificial Neural Network
AOCS	Attitude and Orbit Control System
ASR	Advance in Space Research
AST	Aerospace Science and Technology
BERT	Bidirectional Encoder Representations from Transformers
BNC	British National Corpus
BOW	Bag Of Words
BPE	Byte-Pair Encoding
CBOW	Continuous Bag-Of-Words
CCDS	Concurrent and Collaborative Design Studio
CDF	Concurrent Design Facility
CDP4	Concurrent Design Platform-4
CE	Concurrent Engineering
CNN	Convolutional Neural Network
ConCORDE	Concurrent Concepts, Options, Requirements and Design Editor
COSPAR	Committee on Space Research
COTS	Commercial Off-The-Shelf
CR	Concept Recognition

DARPA	Defense Advanced Research Projects Agency
DBOW	Distributed Bag Of Words
DEA	Design Engineering Assistant
DL	Description Logics
DLR	Deutsches Zentrum für Luft- und Raumfahrt (German Aerospace Center)
DM	Distributed Memory
DSM	Distributed Space Mission
ECSS	European Cooperation for Space Standardization
EEE	Electronic & Electrical Engineering
EKG	Entreprise Knowledge Graph
ELMo	Embeddings from Language Models
EM	Engineering Model
EO	Earth Observation
ER	Entity-Relationship
ESA	European Space Agency
ESOC	European Space Operations Centre
ESTEC	European Space Research and Technology Centre
FNNLM	Feed-forward Neural Network Language Model
FOL	First-Order Logic
GLUE	General Language Understanding Evaluation
GPT	Generative Pre-Trained
HAL	Hyperspace Analogue to Language
HMI	Human-Machine Interaction
IAA	International Academy of Astronautics
IAC	International Astronautical Congress
IAF	International Astronautical Federation
IDM	Integrate Design Model
IE	Information Extraction
IR	Information Retrieval
ISS	International Space Station

JPL	Jet Propulsion Laboratory
JS	Jensen-Shannon
KG	Knowledge Graph
KM	Knowledge Management
KR	Knowledge Representation
LDA	Latent Dirichlet Allocation
LL	Lesson Learned
LSA	Latent Semantic Analysis
LSI	Latent Semantic Indexing
LEO	Low Earth Orbit
LIDAR	Light Detection and Ranging
LSTM	Long-Short Term Memory
MAE	Mechanical & Aerospace Engineering
MBSE	Model-Based System Engineering
MER	Mars Exploration Rover
ML	Machine Learning
MLM	Masked Language Model
MRR	Mean Reciprocal Ranking
MWE	Multiword Expression
NASA	National Aeronautics and Space Administration
NDA	Non-Disclosure Agreement
NEA	Near-Earth Asteroid
NEO	Near-Earth Object
NER	Named-Entity Recognition
NLG	Natural Language Generation
NLP	Natural Language Processing
NLTK	Natural Language Tool Kit
NLU	Natural Language Understanding
NMT	Neural Machine Translation
NNLM	Neural network language model

NPI	Networking Partnering Initiative
NSP	Next Sentence Prediction
OBDH	On-Board Data Handling
OCDT	Open Concurrent Design Tool
OHE	One-Hot Encoding
OL	Ontology Learning
OMG	Object Management Group
OO	Object-Oriented
OOV	Out-Of-Vocabulary
OSIP	Open Space Innovation Platform
OSMoSE	Overall Semantic Modelling for System Engineering
OWL	Web Ontology Language
PCA	Principal Component Analysis
PDC	Project Design Center
PDF	Portable Document Format
POS	Part-Of-Speech
PTB	Penn Treebank
RDF	Resource Description Framework
regex	Regular Expression
RNN	Recurrent Neural Network
RoBERTa	Robustly optimized BERT pretraining Approach
SECESA	Systems & Concurrent Engineering for Space Applications
SERDL	Space Engineering Reference Data Library
SEIM	System Engineering Information Model
SEVA	Systems Engineer's Virtual Assistant
SMAD	Space Mission Analysis and Design
SOTA	State-Of-The-Art
SQL	Structured Query Language
SSL	Self-Supervised Learning
SVD	Singular Value Decomposition

SVM	Support Vector Machine
Tf-idf	Term frequency - inverse document frequency
TM	Topic Modelling
TPE	Tree of Parzen Estimators
TPU	Tensor Processing Unit
TRAC	Technical Requirements Ambiguity Checked
UML	Unified Modelling Language
ULMFit	Universal Language Model Fine-Tuning
W3C	World Wide Web Consortium
WI	Weirdness Index
WS	Weighted Sum
WSP	Web Services Processor

List of Tables

Chapter 1

Introduction

International Space Station, November 2018:

Alexander: "*Help me with the EPO*[1] *Crystallisation procedure*"

Cimon: "*The requirement equipment is shown on the screen, are you ready to start?*"

Alexander: "*Yes.*"

Cimon: "*The EPO Flying Classroom Crystallisation of charged particles demonstrates the charging of nylon and teflon particles after shaking. Now we can start the main experiment.*"

Alexander: "*What equipment do I need?*"

Cimon: "*Sorry I'm just a robot, I don't know everything you mentioned.*"

The above is extracted from a conversation between astronaut Alexander Gerst and CIMON, the Crew Interactive MObile companioN[2]. CIMON is a technology demonstrator for an astronaut assistant developed by the German Aerospace Center (DLR), Airbus, IBM and the Ludwig Maximilians University of Munich as introduced by Schröder et al. in [1]. Designed as a spherical free flying unit of around 32cm diameter, CIMON is meant to support crew activities, displaying experiments procedures, and answering technical queries. As humanity ventures beyond Low Earth Orbit, communication delays will soon prevent live monitoring and support of crews by ground

[1]Education Payload Operations

[2]`https://www.youtube.com/watch?v=HE0LQ2y_-Pk`

control. This is precisely when virtual assistants, or expert systems, such as CIMON will become crucial team members. Expert systems, as defined by Lucas and van der Gaag [2], store large amounts of technical knowledge and can mimic experts' reasoning. These types of systems were popular in the 90s but failed to meet expectations due to limited computational power and the limitation of rule-based inferring [3]. "Big Data", defined as *the Information asset characterized by such a High Volume, Velocity and Variety to require specific Technology and Analytical Methods for its transformation into Value* by De Mauro et al. in [4], as well as a significant increase in available computational power gave them a second chance. Virtual assistants, such as Apple's Siri, Amazon's Alexa, or Google Assistant, have become pervasive to our daily life. Although these assistants can perfectly handle simple daily tasks, they would be of lesser help to design a complex system. Domain-specific assistants are on the rise in several technical fields, notably for medical and legal applications. For instance, the expert system HIPPOCRAT-EES, presented by Marakakis et al. in [5], simulates the reasoning of neuroscientists to diagnose epilepsy. There are more than 50 epilepsy types and their classification is highly complex as each type is a combination of 28 diagnosis criteria. HIPPOCRAT-EES achieves 83.3% of successful diagnosis. In the legal field, ROSS Intelligence supports research tasks. Lawyers have to conduct lengthy legal searches through jurisprudence and law books for each case they accept. As underlined by ROSS's creators in [6], with high hourly rate, this research often represents a cost some clients simply cannot afford. ROSS was found to decrease by 30% the research time, thus lowering the access barrier to legal representation.

This thesis stems from the vision of a virtual assistant for space mission design, a Design Engineering Assistant (DEA). Although computer vision and attitude control enhanced with Artificial Intelligence (AI) are now common in the space field, other AI branches are yet to infiltrate the full spacecraft life cycle. Meanwhile, searching through accumulated design data resulting from decades of spacecraft development slows knowledge management and reuse. This thesis lays the first building blocks for the DEA, exploring the latest NLP and text mining methods, to adapt them to space systems and contribute to improving current design processes.

1.1 Towards a Design Engineering Assistant

This section presents a potential architecture for a future Design Engineering Assistant (DEA). As defined by Lucas and van der Gaag [2], a virtual assistant usually combines three core elements: (i) a Knowledge Base containing all the data, (ii) a reasoner or inference engine, and (iii) and interface. As seen on Figure 1.1, the suggested architecture of the DEA is based on similar elements: (i) a Knowledge Base and additional libraries containing domain-specific models and rules, (ii) a query manager matching the user query with its relevant application, and (iii) an interface.

A Knowledge Graph (KG) is suggested to store the collected data. As will be discussed in Chapter 6, graph databases are more adapted than relational solutions for relationship-heavy use cases, when seeking hidden connection and new insights from data. A prerequisite for the KG is an ontology, defining the allowed nodes and edges of the graph. With an ontology, the heterogeneous data collected can be restructured according to a single conceptual model, which furthermore allows rule-based reasoning. The reasoner is the "*brain*" of the system, querying the data stored in the KG and inferring knowledge. The library of domain-specific models would include text mining methods tailored to space systems such as the ones that will be presented in Chapters 4 and 5. The library of rules could either contain formulas, or *if-then* rules.

The assistant should be able to handle various types of queries, a sample is displayed on Figure 1.1, in the *Query Manager* block. These queries are divided between continuous and dotted boxes, illustrating two distinct development stages of the DEA. The continuous line elements correspond to queries to be handled by a passive assistant, a DEA which would only manipulate information stored in the Knowledge Base. On the other hand, the dotted line elements are associated with an active assistant able to assess a new design, run in the background of a study, flagging design conflicts or design decisions that would not match the initial requirements. These last three elements named *Design Calculation*, *Design Verification*, and *Design Suggestions* require further developments. Calculations could be handled by building a thorough database of formulas used in the field. The design verification would require a NLP layer to understand the mission requirements and compare them to the current design options. The generation of design solution is a complex problem, generative ML methods such as Generative Adversarial Networks (GANs) could be explored.

Finally the interface block is the theater of the Human-Machine Interaction (HMI). It could either be integrated, as a plug-in, into an existing design environment such as the CDP4, or it could be a stand-alone interface. Off-the-shelf tools such as open-source RASA[3], Google's Dialogflow or VisionSpace's Karel.ai [4], provide today the building blocks for creating conversational agents.

The suggested consecutive development steps for the DEA are the following:

1. **Start with a passive assistant:** A tool to retrieve information from the Knowledge Graph, and perform text mining tasks (classification, heritage analysis, budget inference) on the knowledge previously gathered.

2. **Evolve to an active assistant:** A tool capable of assessing a new design, flag its flaws with respect to initial requirements, and suggest design improvements.

3. **Evolve to a conversational agent:** A tool capable of interacting with its user in natural language, and continuously learn from their feedback.

[3]`https://rasa.com/`
[4]`https://www.karel.ai/`

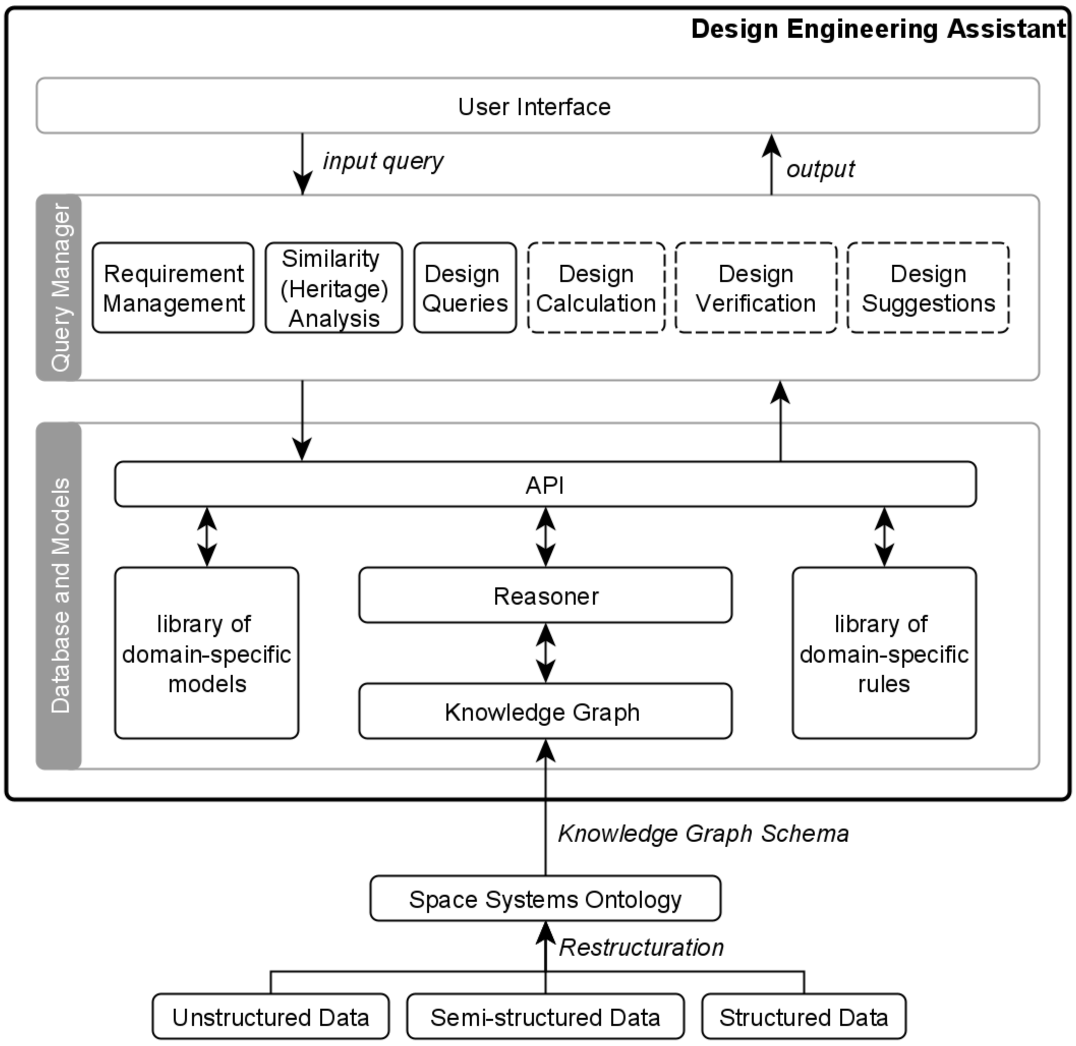


Figure 1.1: Suggested architecture for a Design Engineering Assistant

1.2 Research Aims & Objectives

The aim of this thesis is to lay the foundations for a DEA, a virtual assistant supporting decision-making at the early stages of space mission design. To do so, building blocks for the DEA are developed based on novel Natural Language Processing (NLP), text mining and Machine Learning (ML) methods. Each domain-specific methods developed in this thesis can be used as a stand-alone tool to support knowledge management and reuse. The work presented here also aspires to alleviate the knowledge reuse bottleneck often encountered by experts in the early design phases, and contribute to improving design processes in the space field. This thesis focuses on the unstructured data and semi-structured data used or generated during feasibility studies applying the concurrent engineering approach. To achieve the main project goal, the following core objectives will be addressed:

1. Review the latest design methodologies in Concurrent Engineering, both in theory and in practice by surveying experts involved in concurrent engineering design sessions. Understand the obstacles met by experts regarding knowledge management and reuse.

2. Build a first collection of unstructured and semi-structured data dedicated to space systems engineering that will serve as a base to train and evaluate the developed methods.

3. Develop statistical, embedding and contextualised methods adapted to space systems to semi-automatically generate the initials layers of a space systems ontology.

4. Develop probabilistic and embedding methods adapted to space systems, and understand their scope of applicability within the Concurrent Engineering design process. Train these models with the unstructured data collected.

5. Propose a method to structure the knowledge within semi-structured data generated during Concurrent Engineering studies to improve accessibility and extract new insights.

1.3 Research Outputs

A first library of documents related to space systems is collected and curated. The complete text collection, except for the feasibility reports, property of the European Space Agency (ESA), is openly available from the University of Strathclyde KnowledgeBase at `https://doi.org/10.15129/8e1c3353-ccbe-4835-b4f9-bffd6b5e058b`. The text collection is processed with a NLP pipeline tailored to space systems, developed in the frame of this thesis.

The majority of the research presented here has been published either in conference proceedings or in journal articles:

1. "*Artificial Intelligence for Early Design of Space Missions in Support of Concurrent Engineering Sessions*" presented at the 8th International Systems & Concurrent Engineering for Space Applications Conference (SECESA 2018) [7].

2. "*Towards an Artificial Intelligence based Design Engineering Assistant for the Early Design of Space Missions*" presented at the 69th International Astronautical Congress (IAC 2018) [8].

3. "*Artificial Intelligence for the Early Design Phases of Space Missions*" presented at the peer-reviewed 2019 IEEE Aerospace conference [9].

4. "*The automatic categorisation of space mission requirements for the Design Engineering Assistant*" presented at the 70th International Astronautical Congress (IAC 2019) [10].

5. "*Space mission design ontology: extraction of domain-specific entities and concepts similarity analysis*" presented at the AIAA SciTech 2020 Forum, invited session on Cognitive Assistants [11].

6. "*From Engineering Models to Knowledge Graph: Delivering New Insights Into Models*" presented at the 9th International Systems & Concurrent Engineering for Space Applications Conference (SECESA 2020) [12].

7. "*SpaceLDA: Topic distributions aggregation from a heterogeneous corpus for space systems*" published in the journal of Engineering Applications for Artificial Intelligence [13].

8. "*SpaceTransformers: language modeling for space systems*" published in the journal of IEEE Access [14].

The source code repository for [10–14] is publicly available at `https://github.com/strath-ace/smart-nlp`.

1.4 Suggested Background

A background in space mission design and concurrent engineering is not required to understand the work presented here. Chapter 2 provides the necessary background on relevant design processes. Extensive background on Artificial Intelligence, word embedding methods, language models and Topic Modelling is provided in Appendix 8.2. It is however assumed that the reader has a basic knowledge of ML and notably of Neural Networks architecture.

1.5 Research Funding

This research was partly funded by the European Space Agency in the frame of a Networking Partnership Initiative (NPI) under the ESA Contract No. 4000123680/18/NL/MH. This research was also supported by industrial partners: RHEA Group who provided an industrial placement opportunity, satsearch and AIRBUS.

1.6 Thesis Structure

Chapter 2 provides background information on the two key fields, space mission design and AI, explored in this thesis. A brief overview of space mission design, the Concurrent Engineering (CE) design process, and the overlap between knowledge management and design is first addressed. The theoretical approach of CE is then compared to the design process applied in practice through a survey of experts involved in feasibility studies of space missions. The last sections of the chapter introduce NLP and text mining, as well as previous attempts to integrate AI into the design process of space missions.

Text mining, NLP, and ML methods feed on data. Yet there is currently no large open source collection of documents related to space systems. Chapter 3 presents a first curated collection of texts related to space mission design, including journal publications, feasibility reports, books and Wikipedia pages. The raw unstructured data is parsed and then processed through a novel NLP pipeline tailored to space systems. Chapter 3 also introduces a collection of Engineering Models (EMs), blueprint of spacecraft designs. The EMs are here considered as semi-structured data as they include content in natural language. Both corpora are used to train and test the various methods developed in this thesis, and could constitute the basis for the DEA knowledge base.

Chapter 4 explores a wide range of text mining and NLP methods, from statistical to global and contextualised embedding to address the challenge of generating a space systems ontology from unstructured data. As previously mentioned, an ontology is required to build the knowledge base of the DEA and merge heterogeneous data sources. First, a domain-specific lexicon is semi-automatically inferred with statistical methods. Then, similar concepts are clustered with a word2vec model. The chapter ends by tackling contextualised embedding, a revolutionary approach which appeared at the same time as the work on this thesis started. A new family of domain-specific transformer-based models is developed and fine-tuned to identify space systems concepts.

Chapter 5 develops domain-specific methods to address different types of queries a virtual assistant would have to handle. The chapter focuses on two common tasks handled by system engineers: requirement management and heritage analysis. A spaceLDA model is trained to identify the word distributions of topics related to spacecraft subsystems, and thus classify mission requirements per subsystems. Past feasibility reports are embedded with a doc2vec model to identify similar missions. The case studies and applications enabled by these models demonstrate the potential for text mining and NLP methods to enhance knowledge management and reuse at the early stages of space mission design.

Chapter 6 addresses the knowledge management and reuse of stand-alone Engineering Models based on the ECSS-E-TM-10-25A Technical Memorandum. The chapter first provides a detailed comparison of various database types, with an emphasis on graph databases, and a trade-off of Knowledge Graph (KG) tools. The chapter then presents the building blocks to migrate models to a KG. Through two case studies, an automatic mass budget generation and a heritage analysis, the potential for such data structure to support the early design phases and the DEA Knowledge Base is demonstrated.

Chapter 7 evaluates the findings and contributions from this thesis, as well as addresses the current limitation of the methods applied. This final chapter ends with future work recommendations and notes on the suggested DEA architecture.

Chapter 2

Background

2.1 Early Stages of Space Mission Design

This first section introduces the concepts of space mission design and Concurrent Engineering (CE). The latter is a design approach applied at the early stages of space mission design to accelerate feasibility studies. Understanding the dynamics of CE is essential as it is at the heart of the work presented in this thesis. The section ends with a short reflection on the impact of knowledge management and reuse for design.

2.1.1 Space Mission Design

Space mission design includes all the steps necessary for the development of a spacecraft, from initial mission requirements and scientific objectives to a working system. The architecture of a space mission is typically divided between the space, ground and launch segments [15], as shown in Figure 2.1. The space segment corresponds to the spacecraft itself, and is usually broken down into two major functional blocks, the payload and the platform (also called bus). The space segment may be a unique spacecraft or a constellation of spacecraft. The payload includes all on-board instrument(s) generating the space-based measurements. The platform sustains the payload, providing it, for instance, with power, operational temperatures ranges and shielding from the harsh space environment. The platform is further divided between subsystems corresponding to various engineering disciplines. Additional disciplines such as mission analysis, programmatics, cost and risk may not be represented as hardware or software blocks but they also play a critical role in mission design. The ground segment includes all on-ground facilities enabling the communication with the spacecraft, such as ground stations, and intermediaries transferring the data to the various stakeholders and users. Finally the launch segment carries the space segment to its orbit.

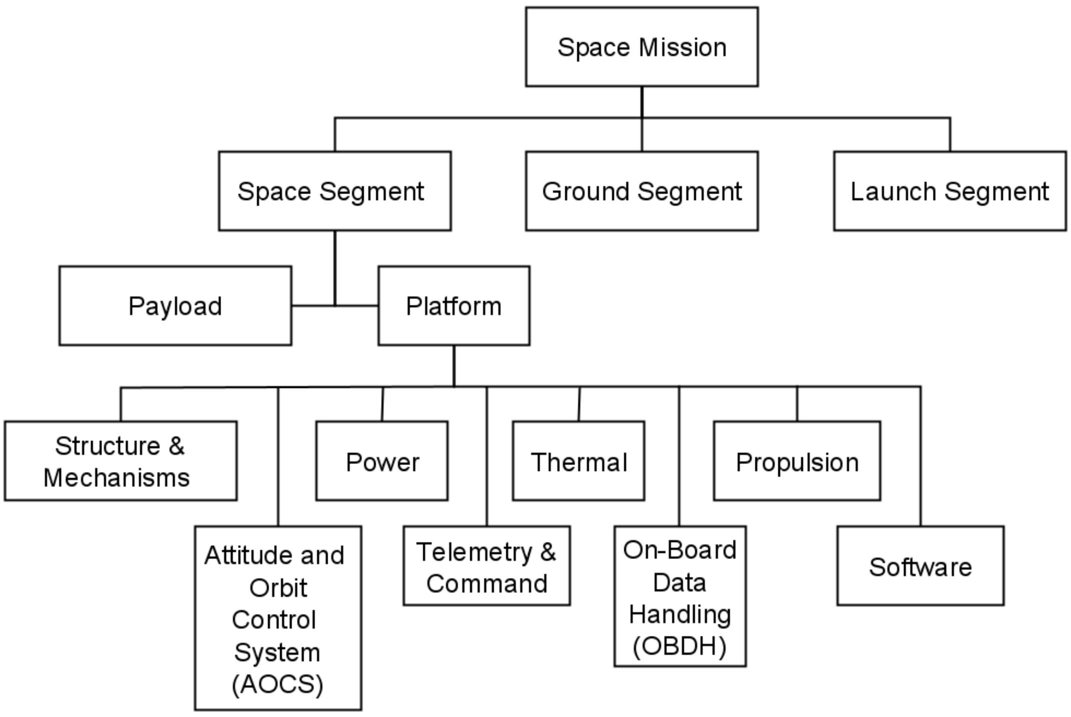


Figure 2.1: Unformalised Element tree of a space mission

The design of space missions is organised in consecutive phases separated by milestones reviews. It can take up to 10 years from the initial conception of a mission to its launch and operation [16]. Although for smaller platform such as CubeSats, the development process has been significantly reduced to 18-24 months [17]. The European Space Agency (ESA) divides its projects into seven phases detailed in the European Cooperation for Space Standardization (ECSS) ECSS-E-ST-10C standard on system engineering general requirements [18]. The ECSS is an initiative launched by ESA in 1993 to define a coherent and single set of standards for all European space activities [19]. The National Aeronautics and Space Administration (NASA) uses a slightly different project planning and milestones introduced in its Space Flight Program and Project Management Handbook [20]. Both project life cycles are highly similar as shown in Figure 2.2.

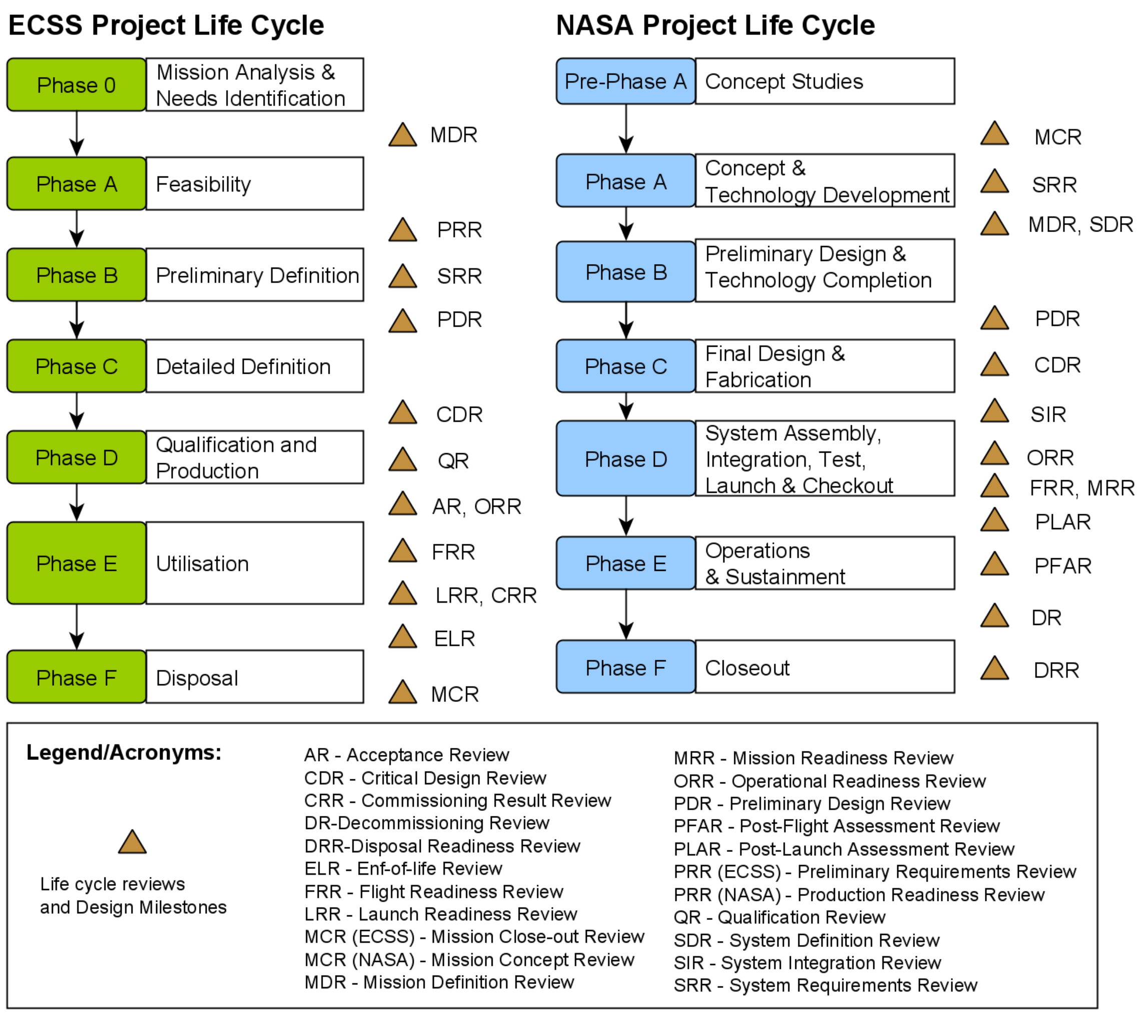

ECSS Project Life Cycle
Phase 0
Mission Analysis & Needs Identification
MDR
Phase A
Feasibility
PRR
Phase B
Preliminary Definition
SRR
PDR
Phase C
Detailed Definition
CDR
Phase D
Qualification and Production
QR
AR, ORR
Phase E
Utilisation
FRR
LRR, CRR
ELR
Phase F
Disposal
MCR
NASA Project Life Cycle
Pre-Phase A
Concept Studies
MCR
Phase A
Concept & Technology Development
SRR
MDR, SDR
Phase B
Preliminary Design & Technology Completion
PDR
Phase C
Final Design & Fabrication
CDR
SIR
Phase D
System Assembly, Integration, Test, Launch & Checkout
ORR
FRR, MRR
PLAR
Phase E
Operations & Sustainment
PFAR
DR
Phase F
Closeout
DRR
Legend/Acronyms:
Life cycle reviews and Design Milestones
AR - Acceptance Review
CDR - Critical Design Review
CRR - Commissioning Result Review
DR-Decommissioning Review
DRR-Disposal Readiness Review
ELR - Enf-of-life Review
FRR - Flight Readiness Review
LRR - Launch Readiness Review
MCR (ECSS) - Mission Close-out Review
MCR (NASA) - Mission Concept Review
MDR - Mission Definition Review
MRR - Mission Readiness Review
ORR - Operational Readiness Review
PDR - Preliminary Design Review
PFAR - Post-Flight Assessment Review
PLAR - Post-Launch Assessment Review
PRR (ECSS) - Preliminary Requirements Review
PRR (NASA) - Production Readiness Review
QR - Qualification Review
SDR - System Definition Review
SIR - System Integration Review
SRR - System Requirements Review

Figure 2.2: Project Life Cycles implemented by ESA [21] and NASA [20]

The work presented in this thesis adopts European standards. During Phase 0, the mission's scientific goals, expected performance and preliminary technical requirements are identified. Mission concepts are drafted but not yet detailed. The Mission Definition Review (MDR) is held at the end of Phase 0 and assesses whether or not the mission should move to phase A. There, the technical, programmatic and financial feasibility of the mission is assessed. Critical technologies that will require further development, and mission design drivers are identified. Initial possible system and operations concepts are elaborated. At the end of Phase A, the Preliminary Requirements Review (PRR) releases the technical requirements specification. A final assessment of the mission concept(s) is given. To avoid wasting resources, it is essential to quickly discard unrealistic missions. The space and ground elements of the mission are designed in phases B and C. In Phase D, the spacecraft is built and tested. Finally it is launched and becomes operational in Phase E. At the end of the mission lifetime, the spacecraft must be disposed off. With the recent efforts to mitigate space debris in densely populated orbits, spacecraft in Low Earth Orbit (LEO) must reenter the Earth's atmosphere within 25 years of mission completion [22].

Phases 0/A are the cradle of space missions. Engineers can let their imagination run free and test innovative solutions provided that they comply with the design requirements, as well as with the cost and programmatics envelope. The decisions taken during these conceptual phases have a considerable impact on the later development phases. About 80% of the system's quality performance is determined during these early phases [23]. Design changes become more and more costly and difficult to implement as a mission progresses through the design phases [24]. The work presented in this thesis focuses on the early stages of space mission design, phase 0 and A, and especially on missions designed with the Concurrent Engineering (CE) approach. The latter method was introduced in the 90s, first at NASA then at ESA, to accelerate the early stages of space mission design.

2.1.2 Concurrent Engineering

The concept

The authors of [25] coined the concept of Concurrent Engineering, following a Defense Advanced Research Projects Agency (DARPA) study, titled *The role of concurrent engineering in weapons system acquisition*, on the improvement of product development process. The authors defined it as: "*A systematic approach to the integrated, concurrent design of products and their related processes, including, manufacture and support.*

This approach is intended to cause the developers, from the outset, to consider all elements of the product life cycle from conception through disposal, including quality, cost, schedule, and user requirements". As the authors of [25] themselves admitted, the term of "Concurrent Engineering" was not new but collected well known practices already applied in product development. Smith, in [26], agrees that CE only formalised design best-practices that were around since the 60s in the manufacturing world.

The keys elements of this design approach are (i) the concurrency of the design process, (ii) the cross-functionality of the team, (iii) a facility with adequate hardware and software infrastructure, and (iv) an Integrated Design Model (IDM) [27]. The designs are iterated on based on inputs and discussions from all the team encouraging communication, team work and information sharing. The CE process fosters communication between the different design actors, involving all stakeholders including the customers. During design sessions, experts from various engineering disciplines work simultaneously on the same system, preferably in the same facility. The facility is at the heart of the CE process. It provides a design environment adapted to the CE process, enabling the experts to access software tools, documentation and easily exchange information stored in the design model. The IDM corresponds to Engineering Models (EMs) which will be introduced in Chapter 3, Section 3.3.

Design methodologies

Classic past design processes include sequential and centralised approaches. Sequential design, as illustrated in [28] corresponds to a linear product development. The concept is transferred from one expert to another, as in a manufacturing line, each expert adding their contribution without interacting with other team members. This process is simple but is slow and often leads to a divergence from the initial requirements. The "design walls" between the different design steps lead to poor communication and can be a source of frustration as explained in [26]. With centralised design, also illustrated in [28], the system engineer is at the center of the design process. The engineer has the heavy task of channeling information to each team member. The CE approach relieves the system engineer's workload by enabling direct interactions between team members, although the system engineer still guides the design process. The differences between these three design approaches are summarised in Figure 2.3.

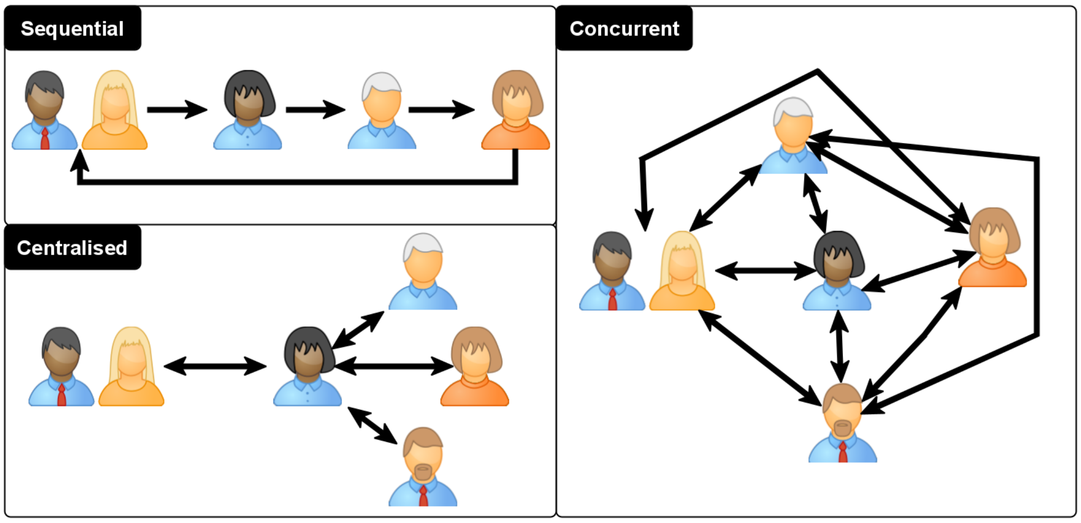


Figure 2.3: Difference between the sequential, centralised and concurrent design approaches derived from [28]

Concurrent Engineering for space systems

Concurrent Engineering made its way to North American and European industries after Japanese competitors successfully implemented it [29]. CE methods are nowadays commonly applied in various industries including the aerospace, automotive and naval fields as covered in [30]. Concurrent Engineering methods were introduced at NASA in the 90s, and were applied by Team X based in the Project Design Center (PDC) of the Jet Propulsion Laboratory (JPL). Team X has since then been using CE methods to accelerate the process of mission definition and preliminary conceptions for new mission proposals with a growing complexity [31]. Inspired by Team X, ESA set up its own Concurrent Design Facility (CDF) at its European Space Research and Technology Centre (ESTEC) in 1998. The goal of the ESA CDF was to establish the initial technical, programmatic and economic feasibility of mission concepts ahead of industrial development. The facility soon proved it could perform studies faster and with better output quality [27]. First an experimental design facility, the CDF evolved into a full operational design facility within 1.5 years of its creation. In the last 20 years, the CDF has performed over 250 studies and has become essential to internal mission assessment. The success of the CDF team has inspired the creation of several concurrent engineering facilities accross Europe. Based on Figure 2.4 provided by ESA, there were 17 concurrent design facilities in Europe in 2017, hosted within space agencies, industries and universities. Figure 2.5 displays two examples of CE facilities, the ESA CDF and the University of Strathclyde's Concurrent and Collaborative Design Studio (CCDS). This latter facility is located on the University of Strathclyde campus in Glasgow (Scotland). The CCDS was opened in October 2015 and is available to both academic and industrial projects [32].

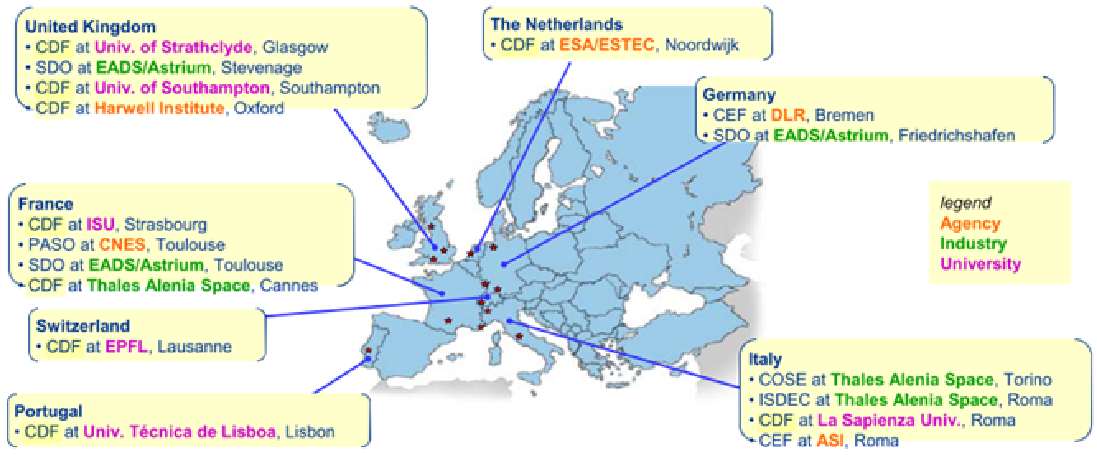

United Kingdom
• CDF at Univ. of Strathclyde, Glasgow
• SDO at EADS/Astrium, Stevenage
• CDF at Univ. of Southampton, Southampton
• CDF at Harwell Institute, Oxford
The Netherlands
• CDF at ESA/ESTEC, Noordwijk
Germany
• CEF at DLR, Bremen
• SDO at EADS/Astrium, Friedrichshafen
France
• CDF at ISU, Strasbourg
• PASO at CNES, Toulouse
• SDO at EADS/Astrium, Toulouse
• CDF at Thales Alenia Space, Cannes
legend
Agency
Industry
University
Switzerland
• CDF at EPFL, Lausanne
Italy
• COSE at Thales Alenia Space, Torino
• ISDEC at Thales Alenia Space, Roma
• CDF at La Sapienza Univ., Roma
• CEF at ASI, Roma
Portugal
• CDF at Univ. Técnica de Lisboa, Lisbon

Figure 2.4: European concurrent engineering facilities in 2016 as seen in [28]

(a) ESA CDF (source:esa.int)

(b) CCDS facility

Figure 2.5: Examples of Concurrent Engineering facilities

Structure of a Concurrent Engineering Study

A concurrent engineering study is usually broken down in three consecutive phases: the preparation, the study and the post-study. As described by [33], the preparation phase usually occurs weeks before the study, with a restricted team including the study lead, the system engineers and the clients. During this phase, the mission heritage, the scientific objectives, and the payload and mission requirements are discussed [16]. The mission requirements are the top level specifications of a spacecraft and serve as a starting point for the design. These could include the mission lifetime, launch opportunities or the expected frequency and resolution of scientific measurements. These initial requirements may however be modified in light of new findings resulting from the study phase. Design drivers, key systems concepts, emerge at that stage, the payload usually being the main design driver [16]. Preliminary results may be presented to the customer. This phase also aims to size the required number of experts and design sessions that will be necessary to meet the expected results within the limited time frame.

The study phase is run with the full design team. At the ESA CDF, the study may require from 6 to 10 sessions, each session lasting half a day [28]. Studies at the CCDS have been run over one week in the form of student-led design challenges [34, 35]. The process is iterative and follows a so-called Spiral Model shown in Figure 2.6. The spiral is divided into sectors corresponding to technical disciplines. Each line corresponds to the value of a key parameter. A target value may be indicated by a T. The system overall mass is, for instance, a common key parameter. To kick-off the design, initial parameters are estimated based on heritage information and the experts' experience. During the sessions, each expert will present their subsystem's suggested architecture, discussing the implications for other subsystems with the team. The parameters' estimations are refined at each iteration, converging towards a sound baseline solution. If the spiral fails to converge, then the design option is discarded. The quality of the initial parameters plays a crucial role in the rate of convergence. Several configurations or design options, each with its own spiral, can be studied simultaneously.

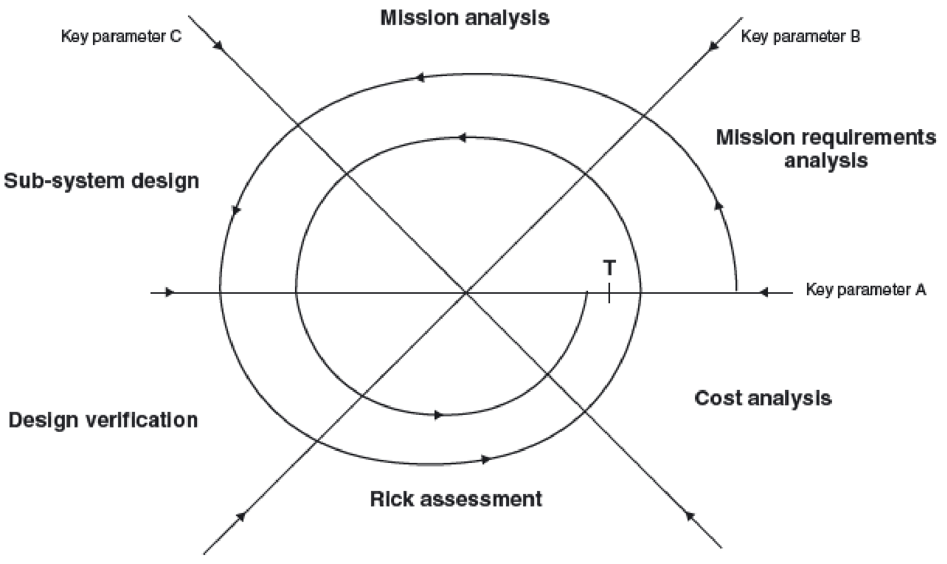


Figure 2.6: Spiral Model illustrating the iterative design process as seen in [16]

A feasibility study has several outputs. One is a study report including a preliminary set of requirements, a first conceptual design, information on the operation concepts, the mass and power system budgets, programmatics, and cost [36]. Information contained in these reports is called unstructured data as it is not stored according to a structured data model. All textual content such as books or scientific publications is in fact considered as unstructured data. If the CE facility has an IDM, then the systems design iterations are saved in an Engineering Model (EM). This type of information is based on a data model and is thus structured. Since the EMs can contain textual (unstructured) information, they will however be considered as semi-structured data. These types of outputs will be furthermore detailed in Chapter 3, Section 3.3. Finally, [37] underlines ESA's past and current efforts to loop-back Lessons Learned (LLs) to the conceptual modeling phase. The authors however mention that there is, currently, no well defined pipeline for the integration of these LLs in the design process, thus the lessons learned will not be considered as a classic output of the design study in this thesis. The various phases and outputs of a concurrent engineering study are summarised in Figure 2.7.

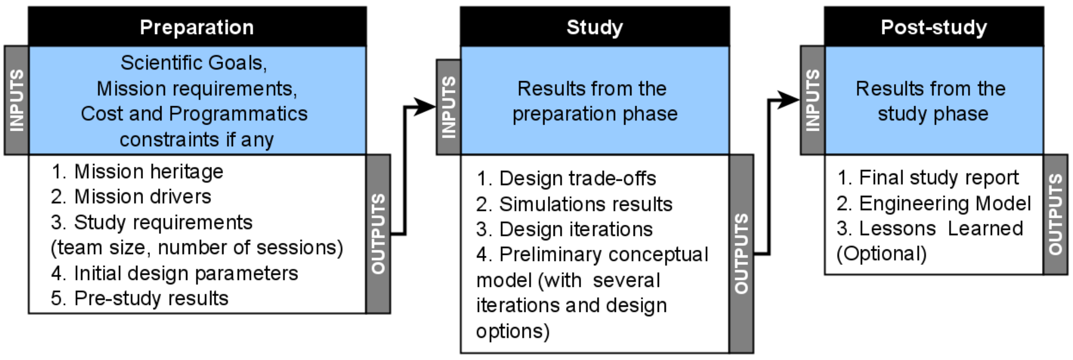

Preparation
INPUTS
Scientific Goals, Mission requirements, Cost and Programmatics constraints if any
1. Mission heritage
2. Mission drivers
3. Study requirements (team size, number of sessions)
4. Initial design parameters
5. Pre-study results
OUTPUTS
Study
INPUTS
Results from the preparation phase
1. Design trade-offs
2. Simulations results
3. Design iterations
4. Preliminary conceptual model (with several iterations and design options)
OUTPUTS
Post-study
INPUTS
Results from the study phase
1. Final study report
2. Engineering Model
3. Lessons Learned (Optional)
OUTPUTS

Figure 2.7: Overview of the Concurrent Engineering process

During the COVID-19 pandemic, several facilities had to adapt their process to further rely on virtual communication means and continue their activities. As presented in [34], virtual exchanges, although not preventing to successfully converge towards system solutions, failed to lift all the additional communication barriers induced by remote work.

Impact of Concurrent Engineering on the Design Process

In [31], team X from NASA JPL reflected on the impact CE had on their design process. They reckoned that it dramatically decreased the costs related to the study as well as the preparation time by 70%. This gain in time allowed them to study around 6 more missions per year. More recent work, [27,38], underlined how CE methods contributed to an increase in study output quality and a decrease in study duration and cost. CE also contributes to a mutual education of inter-disciplinary team members [38]. Tatnall et al., [16], estimate that the introduction of CE at the ESA CDF reduced the study duration from 6-9 months to 3-6 weeks, and the corresponding cost by a factor of two.

While the benefits of the CE method are clear, there still remains some integration challenges [39]. CE is a complex socio-technical process and the success of this approach not only depends on technical aspects but is in great part based on the social component. As mentioned in [27], adopting new processes always takes time and a change of mentality from experts. In lessons learned at the German Aerospace Center

(DLR) concurrent facility, [38] stress several social factors including the need to clearly explain the CE design process to the team members and establish rules and ethics.

2.1.3 Knowledge Management & Design

Knowledge Definition

Dalkir, [40], Kendal and Creen [41], and Misulis and Frisse [42] agree that defining knowledge is a complex task, and that various descriptions exist in the Literature. A traditional definition is that *knowledge is a subset of all true beliefs* [43]. According to the Oxford dictionary, knowledge is defined as (i) the information, understanding and skills that you gain through education or experience, and (ii) the state of knowing about a particular fact or situation. The assimilation between knowledge and information done by the Oxford dictionary might be acceptable for a layperson definition, but is not in the field of information systems or knowledge management. Based on the common data–information–knowledge–wisdom (DIKW) hierarchy [44], often visualised as a pyramid, knowledge can be understood as the extension of information, itself built on data, the pyramid basis. Wisdom is at the tip of the pyramid. The following definitions are thus inferred from the works of Ackoff in [44], Misulis and Frisse [42], Kendal and Creen [41], and Rowley [45]:

1. Data is the product of observations and is fragmented pieces of symbols.
2. Information is inferred, contextualised data.
3. Knowledge is the result of the understanding of information, a structured compilation of information.
4. Wisdom requires judgment.

For instance, a mass value would be data. When associated to a spacecraft's subsystem, it gains context and becomes information. Inferring that the satellite mass budget has been exceeded is knowledge. Suggesting means to reduce the subsystem's mass is wisdom. Knowledge is also often divided between explicit and tacit knowledge [40]. Explicit knowledge includes all content from concrete media such as reports or videos. Tacit knowledge is more complex to articulate as it refers to knowledge acquired by experts through their work experience, including know-hows. Here only explicit knowledge management will be addressed.

Finally, semantics is known as the branch of linguistics focused on the meaning of words. Semantics therefore contributes to the building of understanding, thus knowledge.

Knowledge Management

Knowledge Management (KM) is a multidisciplinary concept, with as many as a hundred different definitions depending on the field of application [40]. In this work, the definition proposed in [46] by the 2018 International Astronautical Federation (IAF) International Programme/Project Management Committee workshop is adopted. The workshop participants surveyed the current KM practice in the space field and defined it as:

> A group of practices ensuring the identification, capture, preservation and sharing of knowledge in order to continuously improve the effectivity and efficiency of a given organization in pursuing its mission.

This definition fits the research interest of this thesis as it stresses how knowledge can actively contribute to improving current processes. The DEA project supports KM as it intends to identify and capture data from heterogeneous sources, restructure it into a Knowledge Graph to facilitate its reuse and therefore contribute to its preservation and sharing.

Knowledge Management & Design

An efficient KM strategy will allow knowledge reuse and avoid wasting energy and time on "*reinventing the wheel*". Dalkir, [40], warns against "*corporate amnesia*" by which an entity risks losing knowledge and eventually efficiency and money due to poor KM.

In the field of engineering design, Verhagen et al, [39], underline that smart application and re-use of accumulated knowledge from previous designs can speed up the study process and improve the output quality. As underlined by [39], the trend in space mission design is towards growing systems complexity. The design of complex systems naturally builds upon prior knowledge as designing from scratch is more demanding [47]. However, Min et al., [47], warns against the automatic and direct reuse of design as it might not always be the most optimal design solution and even compromise innovative new solutions.

There are expectations expressed by [46,48] that Artificial Intelligence will enhance current KM practices. As underlined in [47], algorithms that will automatically extract and facilitate the knowledge transfer from former related design exercises can profoundly impact future design processes.

2.2 Space Mission Design in Practice: Expert Survey

The previous section introduced the theory of space mission design and the Concurrent Engineering approach. To understand how spacecraft are designed in practice, it was essential to interact with experts involved in feasibility studies. A two-months placement was thus organised with the ESA CDF team. The internship took place during the first year of the PhD at ESTEC, the technical center of ESA, based in The Netherlands. The goals of the internship were:

1. to acquire a deeper and practical understanding of the concurrent engineering process by attending design sessions,
2. to identify the data sources deemed most useful and reliable by the experts,
3. to identify queries of interest for experts,
4. to define the requirements for a future Design Engineering Assistant,
5. to raise awareness on the potential of AI-based virtual assistants to support space mission design.

Eventually, 47 ESA experts were surveyed, either through a round-table (40% of the experts) or during face-to-face interviews during the summer 2018. The following sections summarise the preparation and results of these interactions.

2.2.1 Expert Selection

Concurrent Engineering sessions involve experts with various backgrounds, depending on the system or subsystem they provide expertise on. Each subsystem expert has a different perspective on the CE process. For instance, the configuration team steps in at a later phase of the design sessions. The diversity of background was thus encouraged in the selection of the surveyed experts. The intuitive approach in the expert selection process is to involve as many experts as possible. Reference [49] argues that increasing the experts pool can help reduce errors and bias associated to subjective judgements. An effort was thus made to identify at least two experts from each subsystem. The CDF team contributed to the identification of 90 experts, including systems, subsystems engineers but also CDF users. Due to the limited time frame, indicators such as the years of experience or the number and variety of studies involved in, were used to assign priority weights.

From the initial list, 47 experts were surveyed. 18 experts were surveyed through a round-table while 29 experts were interviewed face-to-face. From the 47 experts interviewed, 36% were systems engineers, 36% were subsystems engineers, 15% represented the Knowledge Management team of ESA, 7% were regular CDF internal clients and 6% of the participants came from the inspector general services and the harmonisation activities. The system and subsystems engineers, the primary target users of the DEA, were equally represented. The system engineers belonged to the CDF team, acting as study team leader, system engineer or assistant system engineer. All subsystems were represented as experts from mission analysis, Attitude and Orbit Control System (AOCS), On-Board Data Handling (OBDH), propulsion, thermal, telemetry and command, structure and mechanism, power, operations, risk, cost and programmatics were surveyed.

2.2.2 Knowledge Elicitation Strategy

Shadbolt and Smart, in [50], define knowledge elicitation as the "*set of techniques and methods that attempt to elicit the knowledge of a domain expert*". There are two main types of elicitation protocol, natural and contrived techniques. Natural techniques includes interviews and observation of experts during a problem solving exercise. With the contrived approach, experts are submitted a task, for instance concept sorting, and are observed while performing the task. In a similar study, presented in [51], astronauts were interviewed to define the requirements for a cognitive assistant to be used on-board a space station or manned spacecraft. As underlined in [50], interviews are the most commonly used elicitation techniques. Natural techniques were also preferred for this survey as they are less formal and easier to organise than the contrived approach.

A few round-tables were first planned to elicit knowledge from experts in groups. However, after the first round-table, it became obvious that the group dynamics could prevent some experts from expressing their opinions. The elicitation process was then reviewed to focus on face-to-face interviews. Eventually, 18 experts attended the expert round-table and 29 were interviewed in face-to-face meetings. The results of the survey are presented in the next section.

The round-table started with a short introduction on the project and design assistants. Then, questions were submitted to the experts via a Mentimeter[1] presentation, an interactive software which collects and stores answers from the audience. The questions aimed to understand the experts' current work habits and assess their open-mindedness towards AI and virtual design assistants. The round-table questions can be found in Appendix 1, Table 8.1. The round-table lasted a couple of hours. The face-to-face interviews were usually around 1h, and followed a similar process as the round-table. Although, the interviews were more time-consuming than a group discussion, they allowed more in-depth conversations with the experts. The interview template can be found in Appendix 1, Figure 8.1.

2.2.3 Survey Results

The figures in this section were generated from the data collected through the Mentimeter presentation and are thus based solely on the round-table answers. However the comments include observations collected from all experts.

Current work habits

The first set of questions aimed to better understand the work habits of the experts, and identify the information sources they relied on. Figure 2.8 is a key image demonstrating the need for improved knowledge management for spacecraft design. As shown in the figure, a majority of experts spend from 25% and 50% of their time searching for information in accumulated knowledge. Further discussions with experts confirmed their needs for a quicker access to reliable and synthesized information. According to the experts, heritage knowledge is essential to the preparation and early study phase. By screening previous similar missions, experts are able to provide better parameters estimations, accelerating the study kick-start. Heritage information also helps a new

[1] `https://www.mentimeter.com/`

team member get up to pace quickly. As was underlined in [46], young professionals have more difficulty accessing data than experimented colleagues.

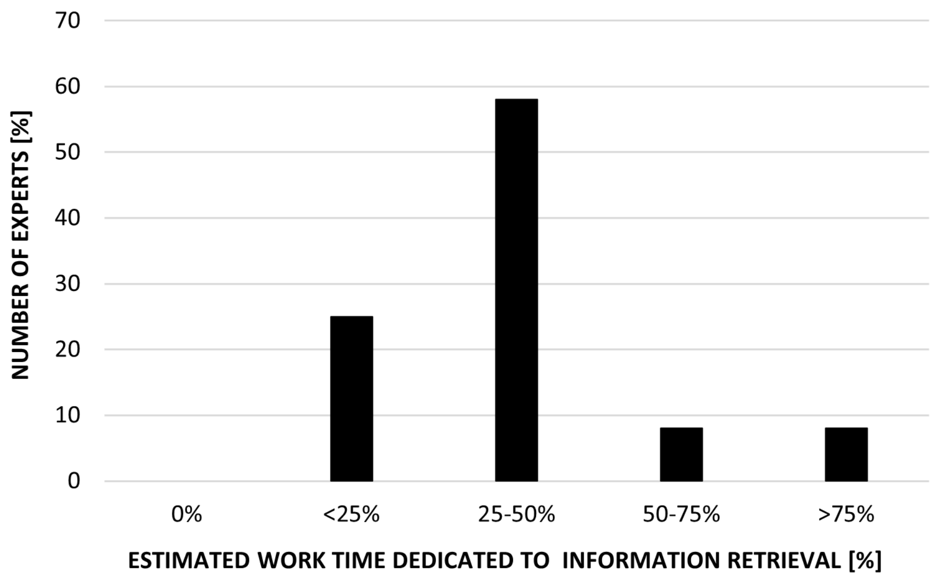


Figure 2.8: Estimated percentage of work time spent by experts searching for information (based on round-table answers)

Figure 2.9 ranks the experts' preferred sources of information. Colleagues ranked the highest as sources of information. Not only do human experts provide a quick answer, they also offer tacit knowledge which is more precise than raw information found in documents. However, human experts can forget or have a biased opinion. The second preferred source of information is past missions reports, textual documents containing valuable heritage information. Internal databases ranked lower than online material. From the discussion with experts, it became clear that the lack of maintainability was often a major impediment to their usage. The level of population and maintainability of these internal databases seemed to vary in function of the teams.

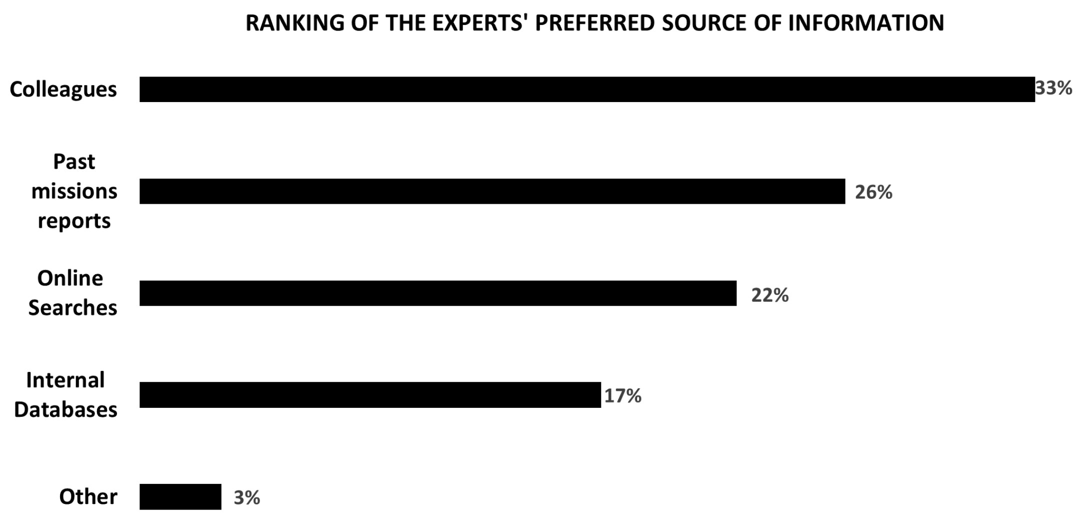


Figure 2.9: Ranking of the experts' preferred sources of information

The integration of a design assistant into the current design process

The second subset of questions anticipates the development of a Design Engineering Assistant that would be actively integrated into the design process.

When asked if they would rely on an AI-based virtual assistants to mine information for them, experts clearly showed their scepticism. 70% of experts were unsure that they would rely on a design assistant while only 20% answered positively and 7% negatively. Although the concept raised some interest among the experts, it was made clear that the tool reliability first needed a thorough investigation before adoption.

The expectations of the experts for a design assistant were assessed. In Figure 2.10, the types of information experts would be interested in are ranked. Experts ranked the understanding of previous design decisions (trade-off outputs and heritage) the highest. The similarity of requirements with previous missions was also a type of information highly requested. Analysing requirement similarity allows to link a study to a relevant design heritage.

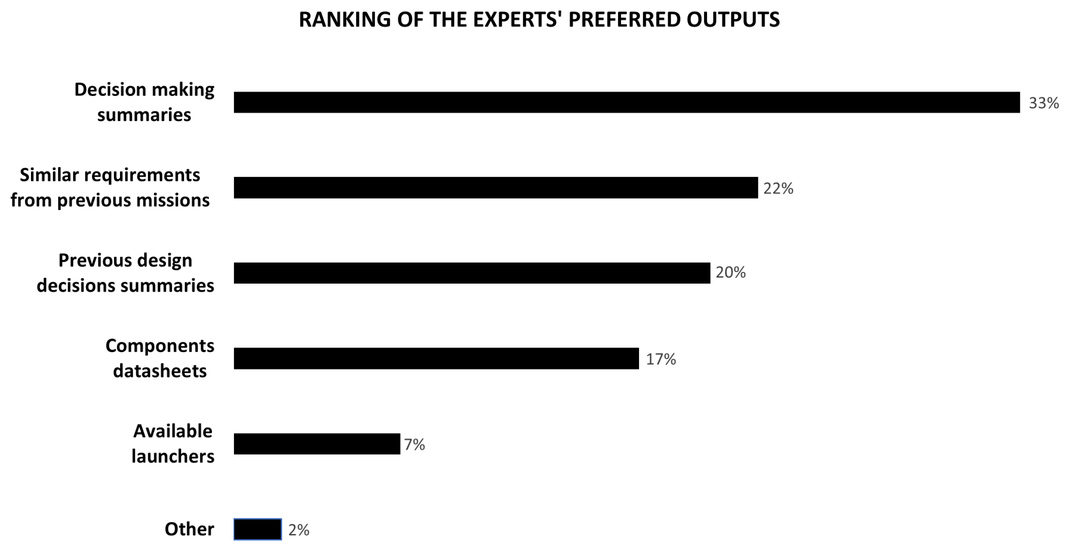


Figure 2.10: Preferred type of information that could be provided by a design assistant, as ranked by the participants of the round-table.

Finally, the experts provided examples of queries they would be interested to submit to a design assistant. The queries sample presented in Table 2.1 are organised per engineering disciplines. They have a broad spectrum of complexity, ranging from a simple extraction of a component value to a complex performances comparison.

Table 2.1: Sample of queries collected during the experts interviews. (AOCS stands for Attitude and Orbit Control System)

Type of Query	**Subsystem**	**Query**
Heritage: Past Design Decisions	AOCS	*Get the AOCS architecture for mission X.* *What is the class of performance for the equipment used in mission X?*
	Mission Analysis	*Find previous missions with similar orbits.*
	System	*Display the heritage of the platform.* *Which launchers could accommodate the spacecraft?*
	Thermal	*Which thermal control hardware was used in mission X?* *How much mass per surface area is assigned to the radiator of mission X?*
Trade-Offs	AOCS	*Does this mission require a gyroscope?* *Are thrusters needed or reaction wheels are sufficient?*
	Propulsion	*Which engine would fit the required range of max. power [min1,max1] or min. thrust [min2,max2]?* *Can the selected thrusters withstand the mission lifetime?*
	Systems	*Provide platform for payload mass X and/or power Y.*
Information Retrieval	AOCS	*What is the TRL level of equipment Y?*
	Propulsion	*How much fuels corresponds to a ΔV of x kms?* *What is the probability of failure of equipment X?*
	Operations	*What are the specification of the ground stations: antenna sizes, frequencies.* *Provide a list of antenna and their specifications.*
	Systems	*Get a platform for payload mass X and/or power Y.*
	Telecom	*Find the frequency type best fitted to the mission type X.* *Find the bandwidth limitation for the mission Y.*

These queries provide indications of the *Design Queries* that could be submitted to the DEA as seen in Figure 1.1.

2.2.4 Discussions

A key finding is that experts spend from 25% to 50% of their work time searching for information. The survey reveals a major knowledge bottleneck at the early stages of space mission design. Knowledge reuse is particularly essential to the preparation phase. When kick-starting a study, experts seek heritage information from previous similar missions, to get a better idea of parameter values and architectures validated by previous studies. The more accumulated data there is on past missions the more time consuming it is to carry this heritage analysis. Observing the CDF team process confirmed that a primary source of information is colleagues. During the placement at the CDF team, it could be observed that one experienced engineer was the prime source for heritage knowledge. This highly knowledgeable colleague has however retired since then.

Experts expressed their interest in accessing various sources of information sources through a knowledge portal, including for instance, design presentations of each iteration, reports from later design phases, technology development updates, LLs or anomaly investigation reports. It would indeed be highly relevant to loop back information from more developed or even flown missions to the early design phases, as was partly attempted by [37]. This could highly contribute to the generation of more feasible and reliable design solutions or avoid repeating similar design errors. Data exchange is however often prevented by confidentiality provisions, isolation of information known as data silos and knowledge hoarding (unwillingness to share information with others). Projects such as the DEA may eventually encourage data sharing by demonstrating the value of reusing knowledge rather than hoarding it.

Although the experts expressed their interest in text mining and AI-based tool, they remain skeptical about their actual integration into the design process. These observations only involved ESA experts, a future survey could attempt to collect observations from different CE facilities, who might follow different design processes or have another approach to the integration of AI in their work environment.

2.3 Introduction to Natural Language Processing and Text Mining

This section provides a broad overview of the Natural Language Processing (NLP) and text mining fields. Detailed background on the methodologies applied in this thesis, Ontology Learning, Topic Modeling, Word Embedding and Contextual Embedding will be provided at the start of each section in Chapters 4 and 5. For readers who are not familiar with Artificial Intelligence (AI), a general introduction can be found in the Appendix 8.2, Section 8.2.1. Past and current projects attempting to integrate AI to the early stages of space mission design are also introduced.

2.3.1 Natural Language Processing

Definition

NLP is a branch of AI studying how machines understand spoken and written natural (human) language. NLP transforms natural language into machine-readable language. It is already pervasive to our everyday life. Spam filters, smart assistant, search engines and machine translation applications all rely on NLP.

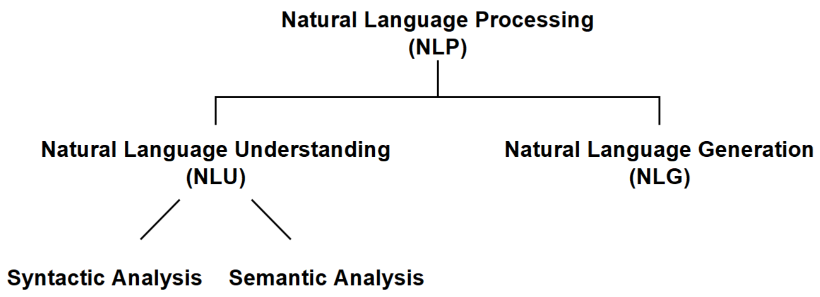

Natural Language Processing (NLP)
Natural Language Understanding (NLU)
Natural Language Generation (NLG)
Syntactic Analysis
Semantic Analysis

Figure 2.11: Classification of Natural Language Processing methods

The two major subtopics of NLP are Natural Language Understanding (NLU) and Natural Language Generation (NLG). NLU focuses on computer reading comprehension while NLG enables computers to generate new data in natural language. The work presented in this thesis is limited to NLU. The latter enables computers to understand natural language based on grammar (syntactic) and context (semantic) [52]. Both approaches are covered in this thesis.

The syntactic approach focuses on the structure of natural languages, basing the analysis on grammatical rules. Part-Of-Speech (POS) tagging identifies, for instance, the grammatical labels of words. A word may be tagged as a noun, a verb, an adjective or an adverb [53]. Rules based on patterns can then be applied to extract relationships between the various grammatical elements following regular expressions pattern such as *subject-verb-object* [53, 54]. On the other hand, the semantic approach focuses on meaning and context. As underlined by [55], the understanding of languages is not just based on a sentence structure, but requires the understanding of context as well. For instance, reading the sentence "*Alice wrote two thesis chapters and ran a marathon this morning, she is on fire today!*", a human would instinctively understand that Alice is being very productive and "on fire" is a figurative image. Without appreciation of the context, a machine would not be able to tell the difference with literally being on fire. Natural language is highly ambiguous, words can have different meaning depending on the context. One of the most difficult NLP task is perhaps sarcasm detection, which is hard to detect even by humans, let alone by machines [56, 57]. Thankfully, in the context of this work based on engineering data, sarcasm should hopefully not be encountered.

NLP Evolution

The evolution of NLP is comparable to the evolution of Artificial Intelligence. The early methods were based on rules (rationalist), assuming that the human language was structured on patterns and strict grammatical rules, notably expressed through the Chomskyan theories of language developed by American linguistic Noam Chomsky [58, 59]. Then, the introduction of data-driven approaches, Machine Learning, led to the adoption of statistical (empirical) approaches, notably introduced by Prof. Christopher Manning, currently professor of Computer Science and Linguistics at Stanford University [60]. The field of NLP has seen a rapid evolution in the last decade, evolving from statistical approaches to probabilistic methods, to word and document embedding, and recently, to contextualised embedding. The background of these methods will be detailed in Appendix 8.2.

2.3.2 Text Mining

Definition

Text mining leverages NLP and ML to extract valuable insights from unstructured text data, uncovering implicit information that would have remained hidden otherwise [61]. While NLP enables the understanding of natural language, text mining enables its analysis.

Main text mining tasks

Text mining tasks are usually divided between:

- *Information Extraction (IE):* The latter is a technique extracting precised information from text such as keywords, named entities or metadata [62].
- *Information Retrieval (IR):* The IR task consists in returning documents of interest from a collection of textual content [63]. Web search engines will for instance retrieve and rank the most relevant web pages answering a user query.
- *Categorisation:* Text categorisation is the process of assigning categories to unstructured content [61]. Topic analysis and sentiment analysis where subjects and feelings are extrapolated from texts, are applications of text categorisation.
- *Clustering:* Text clustering consists in re-arranging a text group into subgroups with similar features or content [61].
- *Summarisation:* Text summarisation refers to the creation a compressed version or summary of a document [62].

2.4 Past & Current Work on the Integration of AI in Space Mission Design

In 2019 ESA surveyed its internal completed, on-going and future activities involving AI [64]. Around 100 activities were divided between four application areas:

1. *Development*: Targeting activities performed during the phase 0 to D of the spacecraft life cycle, and thus including system engineering methodologies, design activities, operational concepts, reliability, quality assurance and Assembly, Integration and Testing / Verification (AIT/AIV).
2. *Operations*: Targeting activities performed during the utilisation phase of the spacecraft life cycle (phase E) including mission planning support and optimisation, automated operations, intelligent personalised assistant, ground stations management and operations for telerobotics, autonomous, and crewed systems.
3. *Exploitation*: This subcategory also targets activities performed during phase E but focuses on downstream data analysis involving the exploitation of Earth Observation (EO), Science, Navigation, Telecommunication measurements, and data contained in models and simulations.
4. *Others*: The *Others* category includes research activities, education and training, knowledge management, administration and media.

The ESA survey found that a majority of its internal activities focused on the *Development* (44%), then *Operations* (23%), *Exploitation* (18%) and *Others* (15%). The work presented in this thesis would certainly fall in the *Development* category, and was likely taken into account in this survey as this thesis is co-funded by ESA. The authors stressed how AI technologies are becoming key enablers in the design, operations and utilisation of space missions. The authors also highlighted how an increased awareness of AI potential, considering AI since the early stages of mission design, and overcoming data silos are among the key factors for a successful integration. A recent example of a spin-off of an activity initiated at the European Space Operations Centre's (ESOC) is the Technical Requirements Ambiguity Checked (TRAC) developed by Solenix. Their tool relies on a ML classification algorithm to assess the validity of technical requirements with respect to ECSS standards. The automation of requirement management could save a considerable amount of time for engineers.

2.4.1 Past Studies

The ESA CDF has been actively involved in at least two previous attempts to include AI to the CE process. A first study, "The Manager" [65], saw the development of an expert system based on Analytic Hierarchical Process (AHP) and Fuzzy Logic Theory. As introduced in [66], AHP is a field of decision taking in which a complex problem is broken down into pairwise comparisons. By solving the simpler problems first, the complexity is increased step by step (hierarchically) until the initial complex question can be answered. As defined in [55], Fuzzy Logic is a method of reasoning similar to human reasoning, with a degree of truth between 0 and 1. Fuzzy Logic allows the integration of vagueness in reasoning. The goal of "The Manager" was to accelerate the convergence towards a valid system solution. The tool independently ran several design iterations, increasing or decreasing step by step parameters values looking for a convergence point as in the Spiral Model shown in Figure 2.6. The tool could for instance run simulations during the pre-study to support the parameters initialisation, and explore several design options during the study. The tool reasoning was based on Excel workbooks, thus structured data, with a MATLAB interface.

In a second study supported by the CDF, [23], the author developed a decision-support tool based on neural networks to guide the selection of design parameters. The initial parameters of a design study are estimated from the mission's scientific objectives and known technical requirements. The quality of these initial parameters directly impacts the performance of the design process and the quality of the final outcome system solution. The predictive decision support tool presented in [23] targets design decisions prone to be extrapolated from previous data, and based on inductive reasoning. For instance, estimating the most suitable launcher for the mission from a database of existing space launchers. The study showed a significant reduction in the error of design decision, with a potential to accelerate the turnaround and quality of design iterations. As in [65], the tool's inputs were structured data, stored as Excel workbooks. Discussions with ESA experts, during the 2018 survey, revealed that neither tools were currently integrated into the CDF CE process. The reasons suggested were the prohibitive learning curves required to master the tools, as either a basic understanding of neural networks or of MATLAB were required. These two previous attempts to integrate AI to space mission design at the CDF did not rely on text mining nor Natural Language Processing methods. The previous methods also handled structured data whereas unstructured text data is a prime data source for the work presented in this thesis.

2.4.2 On-going Studies

Daphne [67] and the Systems Engineer's Virtual Assistant (SEVA) [54] are more recent work done in this field. The Daphne project is led by a team of researchers from the Texas A&M University, with the participation of experts from NASA JPL. Daphne is a virtual assistant for designing Earth Observation (EO) distributed missions. Distributed Space Missions (DSM) involve multiple satellites working jointly to achieve a common goal. A satellite constellation is a common example of DSM. Daphne relies on three structured data sources: a database of rules to support rule-based reasoning, a design solutions database, and an historical database containing structured data on past EO missions. Daphne is a complex system whose first building block, the expert rules, dates from 2014 [68]. This assistant can explore new design options, assess their validity, identify design weaknesses and mine data from previous missions. Daphne accepts inputs in natural language, its query builder integrates components of NLP to map the queries from natural language to SQL (Structured Query Language). However the integration of NLP remains limited as the tool is based on data mining rather than text mining. On the other hand, the SEVA project started at the George Masson University in collaboration with the NASA Goddard Space Flight Center (Maryland, USA) does focus on natural language inputs. The authors of [54] introduce the requirements for a virtual assistant to support system engineering activities of information retrieval and question answering. The authors presented a first stage of their project, based on

NLP, consisting in extracting concepts and simple relationships from domain-specific unstructured data. The relationships extracted are based on patterns: {*subject is-a object, subject has-property object, subject has-value object*}. The authors however relied on a general model and did not perform domain adaptation. Results from the SEVA project will be further discussed in Section 4.5.

In conclusion, there is a growing interest for integrating AI into the development of space missions and for the domain-adaptation of text mining and NLP methods. Past projects, [65] and [23], focused on structured data while on-going projects, [54] and [67], are integrating natural language either as a data source or as a facilitator for user interactions. The work presented in this thesis focuses on unstructured and semi-structured data while tailoring several text mining and NLP methods, and therefore goes beyond previous works.

Chapter 3

Corpus

3.1 Chapter Overview

A corpus of unstructured data is required for the training of Machine Learning (ML) methods and to establish the content of the DEA's Knowledge Base. However, there is currently no accessible large textual data set related to space systems. Thus a first task for this thesis was to collect and curate a collection of domain-specific documents. Section 3.2 presents the main and additional heterogeneous text sources used to build this collection. The raw unstructured data is parsed and then processed through a NLP pipeline tailored to space systems.

Unstructured data is not the only type of input encountered at the early stages of space mission design. Integration of Model-Based System Engineering (MBSE) into the design process is on-going in the space field [69–71]. In Europe and at ESA, concurrent design facilities rely on Engineering Models (EMs) based on the ECSS-E-TM-10-25A technical memorandum [72]. The models act as a blueprint of the spacecraft's architecture. They store design parameters and iterations, support interactive design, and facilitate design reviews. They are here considered as semi-structured data as, in spite of being structured, they still contain information, for instance, requirements, in natural language. Section 3.3 details furthermore the structure of these EMs and introduces two models resulting from internal feasibility studies.

3.2 Unstructured Data Collection

Textual data is the raw material of text mining and NLP methods. This section presents the four main text corpus used to train the models and generate the results of Chapters 4 and 5. Two additional text corpus used for case studies are also introduced.

The complete text collection, except for the feasibility reports, property of ESA, is openly available from the University of Strathclyde KnowledgeBase at `https://doi.org/10.15129/8e1c3353-ccbe-4835-b4f9-bffd6b5e058b`.

3.2.1 Main Text Sources

The text collection is built on four heterogeneous data sources: (i) ESA feasibility reports generated during CDF studies, (ii) academic publications, (iii) books and (iv) Wikipedia pages. The CDF reports were cited as a primary source of information by experts during the 2018 survey as Figure 2.9. The academic publications and books are peer-reviewed and contain content verified by humans. Wikipedia is a common open-source knowledge base. Each text source is described in details in sections below.

ESA CDF reports

The reports are, with the Engineering Models introduced in 3.3, a main output of a CE study. Each study report summarises the mission objectives, requirements, design trade-offs, as well as the baseline spacecraft architecture at system and subsystem level. These reports are usually written at the end of the study and are reviewed by a member of the CDF team.

The ESA CDF team granted access to 55 of their feasibility studies reports through a Non-Disclosure Agreement (NDA). The reports were published internally between 2000 and 2018 included. The 55 reports represent 25% of all CDF reports as access to all studies could not be granted for confidentiality and proprietary reasons. The CDF team has studied 219 mission concepts over its 20 years of activities. The studies are classified by the CDF team per field of activity, also called target system. A mission labelled as "Earth" would, for instance, encompass Earth Observation (EO) activities, or astronomical payloads placed in Low Earth Orbit. Some studies focus on specific technological developments and are classified as "Space Infrastructure and Technology" (SIT), other studies labelled as "Launchers and Return Vehicles" (LRV) study the development of launcher capabilities and return mission. As shown in Figure 3.1, the reports subset that was provided covers in similar proportions the different fields of activities addressed by the CDF.

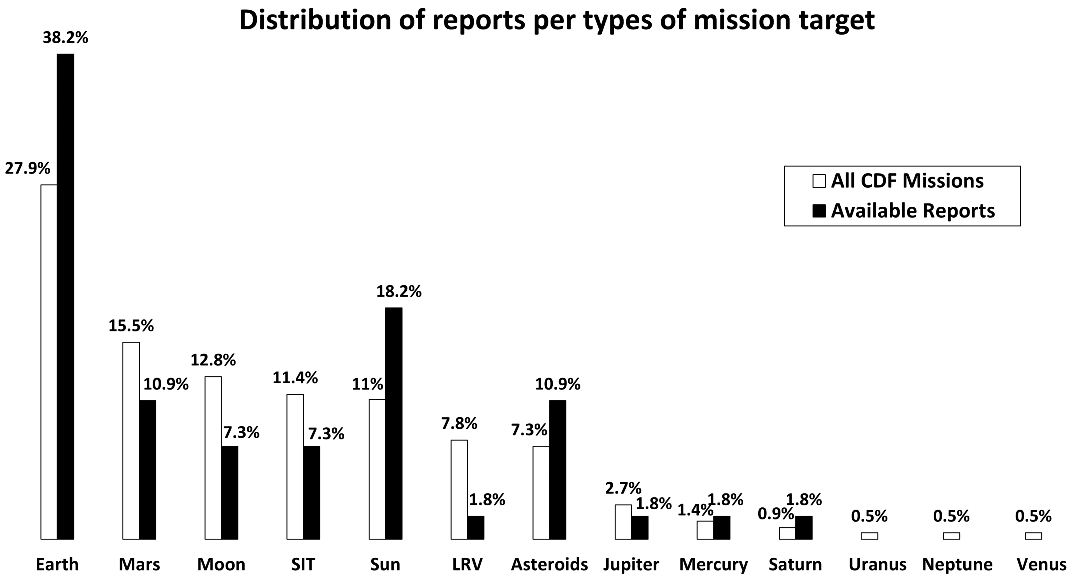


Figure 3.1: Comparison of the topics covered by all CDF reports and the available reports (SIT stands for "Space Infrastructure and Technology", LRV for "Launchers and Return Vehicles").

Academic Publications

The publications corpus includes 4,991 articles published in the *Acta Astronautica* (32.4%), the *Advance in Space Research (ASR)* (30.3%), and the *Aerospace Science and Technology (AST)* (37.4%) journals. These journals were chosen on the following criteria: (i) being peer-reviewed with high citation scores, (ii) having a broad range of topics, and (iii) being accessible either freely or through a University subscription. Only articles published between 2017 and 2019 included were selected to only consider the most recent research findings. The articles were accessed through Science

Direct in partnership with the University of Strathclyde, and downloaded in a Portable Document Format (PDF) format.

The *Acta Astronautica* journal is sponsored by the International Academy of Astronautics (IAA) and addresses topics related to space exploration, and the conception, design, development and operations of both space-borne and Earth-based systems. Special issues of the Acta Astronautica cover selected content presented at the International Astronautical Congress (IAC), including latest developments on satellite and space station technology, space economics, and astrodynamics. The *ASR* is the official journal of the Committee on Space Research (COSPAR) publishing work related to all aspects of space research. Finally, the *AST* journal publishes work related to all fields of the aerospace research, either fundamental or applied.

Books

The books corpus contains 39 books related to space mission design, manually selected, and whose content is publicly available. The selected books cover several fields and subfields of space mission design, including classic textbooks such as the *Space Mission Analysis and Design* (SMAD) [73] and the *NASA Systems Engineering Handbook* [74], as well as more subsystems-specific documents such as *Spacecraft Thermal Control* [75] or *Guidance and Control technology of Spacecraft on Elliptical Orbit* [76]. Figure 3.2 below displays the distribution of the selected books per field of interest. Even if not all spacecraft subsystems are represented, they are covered by system-level books. As shown in Figure 3.3, most of the books selected were published in the past 10 years. Material from the books used in this thesis has been reproduced with permission of SNCSC.

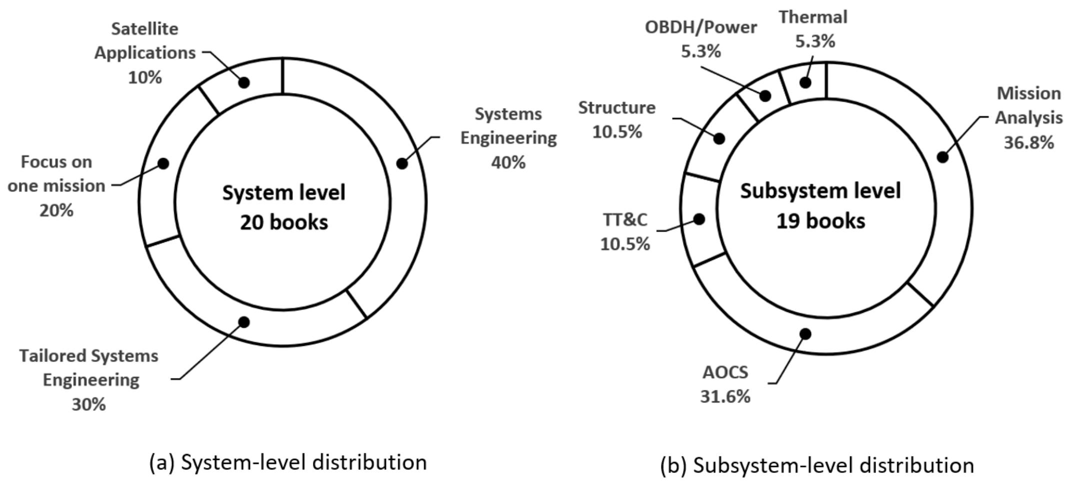


Figure 3.2: Distribution of books topics

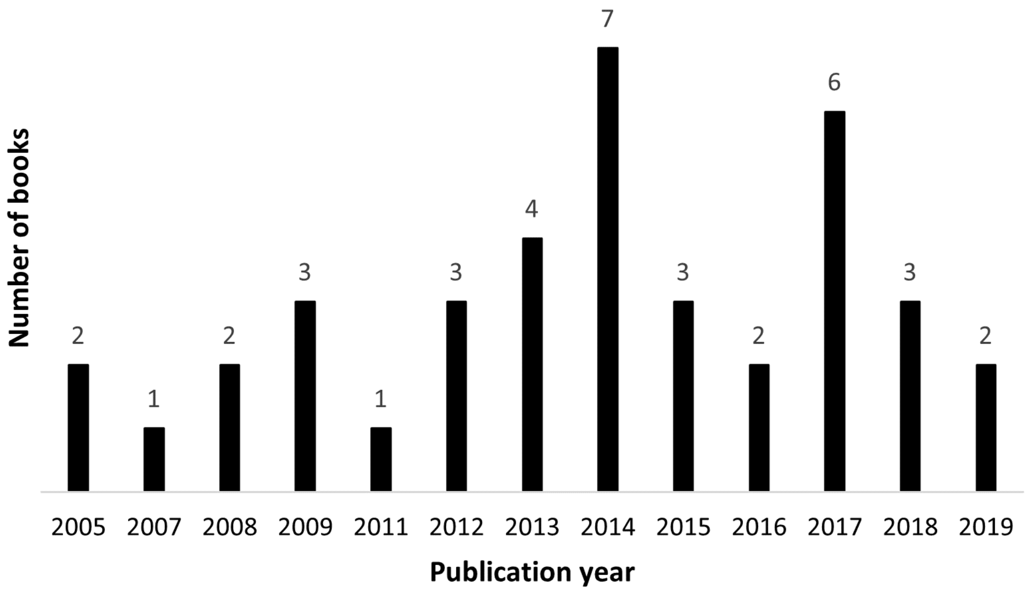


Figure 3.3: Publication years of the selected books

Wikipedia pages

The reliability of Wikipedia content may be questioned but it does represent one of the largest source of freely available texts. Wikipedia data is often used to train Machine Learning algorithms including BERT [77] and GTP-3 [78], or to build NLP benchmark data sets such as the Question-answering Natural Language Inference (QNLI) and the Recognizing Textual Entailment (RTE) sets provided by the General Language Understanding Evaluation (GLUE) benchmark [79]. To collect Wikipedia content related to space systems, the webpage on *Spacecraft Design* is used as a starting point. Additional content is found by exploring the hyperlinks interconnecting the webpages. From the initial page, six hyperlinks, judged as most relevant to a space mission corpus, were manually selected. These webpages were then automatically scrapped using the Python Selenium library[1], leading to the discovery of 1,023 additional non-redundant hyperlinks. The distribution of hyperlinks per webpages, including the main page on Spacecraft design, is shown in Figure 3.4. The list of pages to be included in the corpus was manually filtered, for relevance to the project scope, and eventually yielded a corpus of 242 webpages.

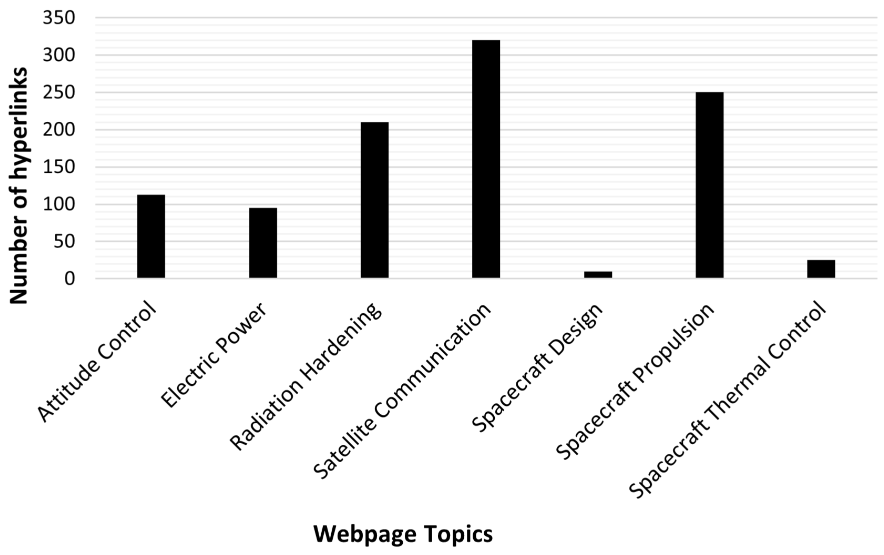


Figure 3.4: Distribution of Wikipedia pages per spacecraft subsystems

[1] `https://github.com/SeleniumHQ/selenium/tree/trunk/py`

3.2.2 Additional Text Sources

Two additional text sources are used as alternative training and case study sets.

ESA Missions requirements

This text collection includes 100 requirements extracted from two ESA documents, publicly available, the SMOS mission System Requirement Document [80] and MarcoPolo-R's Mission Requirement Document [81]. The requirements within these documents are organised per chapter and per subsystem. For instance, all power-related requirements are found under the chapter "*Power requirements*". Design requirements are usually associated with one spacecraft subsystem.

The distribution of requirements per topic is displayed in Figure 3.5. From this corpus, 68 requirements related to the 7 topics of *Attitude and Orbit Control Subsystem* (*AOCS*), *Communication*, *Environment*, *On − Board Data Handling* (*OBDH*), *Power*, *Propulsion* and *Thermal* are used for the case study presented in Section 5.2. Table 3.1 provides samples of requirements extracted from [80].

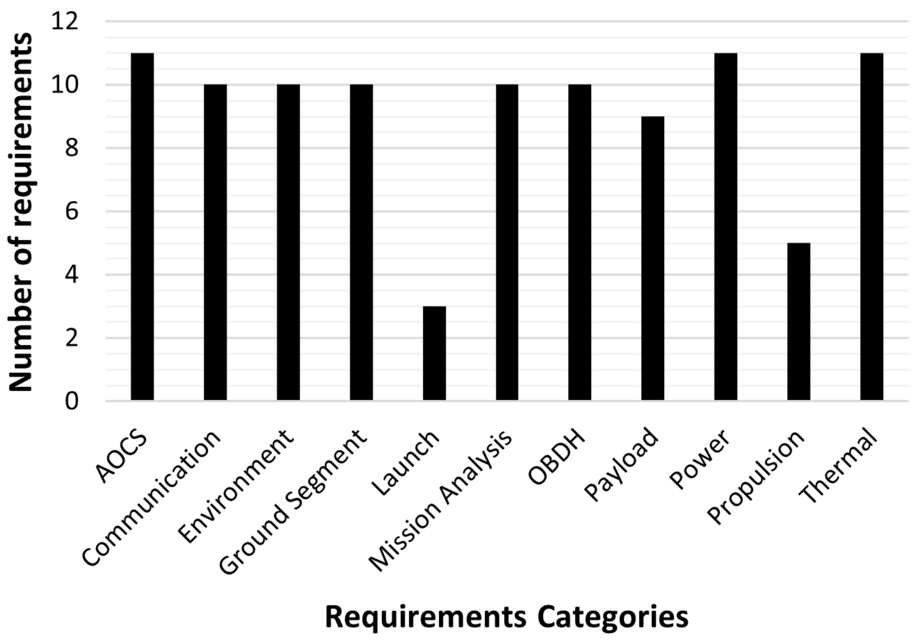


Figure 3.5: Mission Requirements Distribution

Table 3.1: Sample of design requirements extracted from [80]

Subsystem	Requirement
Thermal	*'The thermal control shall be achieved by passive means and by heaters. The use of heat pipes shall be avoided.'*
Attitude and Orbit Control Subsystem	*'In Yaw Steering Mode, the attitude control laws shall be pre-defined law only dependant on one variable: True Latitude'*
Communication	*'The platform communications system shall provide the capabilities to transmit the data stream in S-band.'*

ECSS requirements

The ECSS is an initiative launched by the European Space Agency (ESA) in 1993 to define a coherent and single set of standards for all European space activities [19]. There are currently 129 active ECSS standards, organised in a Document Tree[2] with four branches: *space project management*, *space product assurance*, *space engineering*, and *space sustainability* as shown in Figure 3.6. The branches are further organised per disciplines, covering all topics related to space projects.

Each standard includes several requirements which briefly describe a regulatory provision to be complied within a customer - supplier context as described in [82]. Because of the intent of using them in an obligating contract, the requirements are written in a clear and unambiguous language. The text collection used in this thesis includes 27,016 requirements extracted from 126 active standards (excluding the *Space Sustainability* branch and the Glossary of terms). 66% of these requirements are extracted from the 61 Engineering standards, 32% from the 60 Product Assurance standards, and 2% from the 5 Management standards. These standards were kindly collected and provided by Mirtcheva, S. and Valera, S. (ESA). Table 3.2 provides examples of requirements for each branch.

[2] `https://ecss.nl/standards/ecss-document-tree-and-status/`

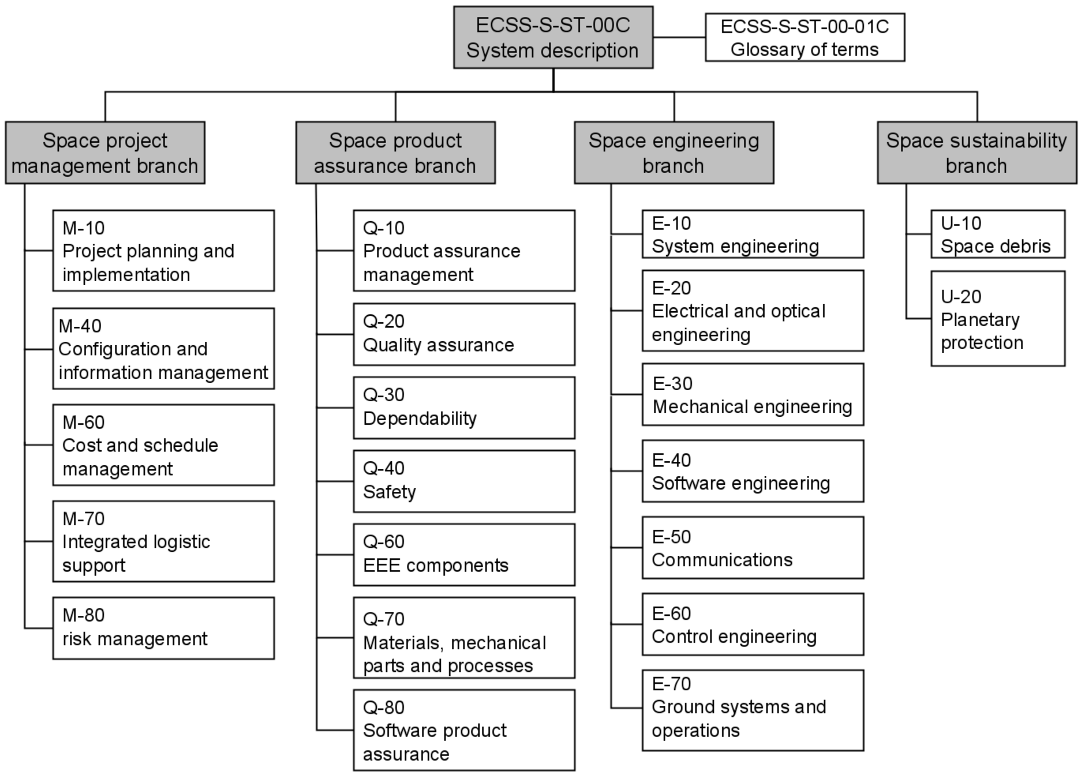

ECSS-S-ST-00C
System description
ECSS-S-ST-00-01C
Glossary of terms
Space project management branch
Space product assurance branch
Space engineering branch
Space sustainability branch
M-10 Project planning and implementation
M-40 Configuration and information management
M-60 Cost and schedule management
M-70 Integrated logistic support
M-80 risk management
Q-10 Product assurance management
Q-20 Quality assurance
Q-30 Dependability
Q-40 Safety
Q-60 EEE components
Q-70 Materials, mechanical parts and processes
Q-80 Software product assurance
E-10 System engineering
E-20 Electrical and optical engineering
E-30 Mechanical engineering
E-40 Software engineering
E-50 Communications
E-60 Control engineering
E-70 Ground systems and operations
U-10 Space debris
U-20 Planetary protection

Figure 3.6: ECSS Document Tree as seen on the ECSS webpage

Table 3.2: Sample of ECSS Requirements

Subsystem	Discipline	Requirement
Management	Project planning & implementation	*'The supplier shall demonstrate that the key personnel have the necessary qualification, skills and experience to perform the task for which they are allocated.'*
	Risk Management	*'Risk management shall be implemented at each level of the customer-supplier network.'*
Product Assurance	Quality Assurance	*'The cleanliness levels shall be specified for molecular contamination in terms of surface contamination'*
	Safety	*'Safety analysis shall evaluate all disposal operations and associated hazards'*
Engineering	System Engineering	*'The system engineering function shall ensure consistency of the requirements at system level, at lower levels, as well as amongst levels.'*
	Electrical & optical engineering	*'All executable commands shall be explicitly acknowledged by telemetry.'*
	Mechanical Engineering	*'Thermal requirements related to the launch and ascent phases shall be specified. '*

3.2.3 Natural Language Processing Pipeline

A NLP pipeline is a chain of analytical tasks to process raw text. The pipeline breaks down, cleans and converts raw text into a format suitable for feature extraction. These steps lay the necessary foundations of more complex language processing tasks [83]. Several open-source toolkits are today available, enabling the quick assembly of these pipelines.

NLP Library selection

Stanford CoreNLP [84], Apache OpenNLP [85], the Natural Language Tool Kit (NLTK) [86], and Explosion's spaCy [87] are the current major open-source NLP tools. These tool kits handle similar syntactical operations.

CoreNLP was developed by Professor Manning and his team from the Stanford University [84]. This toolkit is written in Java. OpenNLP [85] is a ML based toolkit for the processing of natural language text also written in Java. Both CoreNLP and openNLP have Python wrappers. NLTK [86] and spaCy [87] are the key NLP libraries in Python. NLTK is the most well-known tool kit for NLP pipeline. SpaCy is a most recent tool, oriented towards production and fast development.

As highlighted in [88], few published studies tend to justify the use of their NLP libraries, although results obtained are impacted by this choice. As the authors of [88] demonstrated, NLTK performed better than spaCy, and CoreNLP on tokenization but achieved poorer results on Part-Of-Speech (POS) tagging. SpaCy is substantially faster than many other libraries, while NLTK offers a wider range of libraries and modules [89]. NLTK is oriented towards research and thus grants access to a wider range of methods, while spaCy limits the choice of its users and is oriented towards production. The authors of [89] note that OpenNLP obtained poorer results on a NER task.

Since POS is not performed by the NLP pipeline presented here, and Python is the preferred programming language for this thesis, NLTK is chosen to implement the pipeline.

NLP pipeline block elements

An overview of the NLP pipeline developed in the frame of this thesis is displayed in Figure 3.7. The pipeline takes as input raw text parsed from word documents, PDF and text files with the Apache Tika library[3]. The building blocks of a NLP pipeline vary in function of the processing goals. For instance POS tagging is not implemented here but will be applied in later sections focusing on syntactical analyses. To tailor a regular NLP pipeline to a domain-specific corpus, terms, acronyms and stopwords relevant to space systems are integrated to the pipeline. Figure 3.8 displays an example of the NLP processing. The building blocks of the NLP pipeline are the following:

- **Sentence Segmentation:** Blocks of text are first divided in sentences. NLTK uses the Punkt sentence segmenter from [90]. Punkt alleviates splitting ambiguities by identifiying abbreviations, for instance, it will not split a sentence after *Fig.* or *e.g.*
- **Tokenization:** Tokenization is the task of splitting text into simple units called tokens, producing a list of words and punctuations. In NLTK, the method *word_tokenize* divides text by calling the Treebank tokenizer relying on regular expressions (regex) to tokenize text as found in the Penn Treebank (PTB), a very large dataset of annotated words maintained by the University of Pennsylvania [86, 91]. A regex is a sequence of characters specifying a search pattern.
- **Trimming:** Trimming is a simple housekeeping step to removes any blanks before and after a token. When performing frequency analysis, tokens with a blank space introduce redundancy.
- **Special Character and Non English words removal:** Special characters and non-English words add noise and are therefore removed using regex matching.
- **Acronyms Expansion:** Acronyms are common in domain-specific corpora and contain relevant information. An annotated list of abbreviated terms related to space systems is used to expand acronyms found in the corpus. The list and expansion process are further detailed in a below section.

[3] `https://tika.apache.org/`

- **Normalisation:** This step converts all tokens to lower case, putting all words on the same level. This step needs to take place after the acronym expansion as acronyms are usually written in capital letters.
- **Multi-words identification:** N-grams, also called multi-words (MWs) expressions are sequences of n words with a single meaning, for instance, *artificial intelligence* or *jet engine.* Separating MW in different tokens would result in loosing meaning, thus tokens corresponding to a MW are merged. The identification of the few MW with acronyms such as *RF reflector* or *PCB manufacturer* is however done before the acronyms expansion. This process, and the list of domain-specific MW which it is based on are detailed in a later section.
- **Lemmatization:** Lemmatization converts the inflected forms of words to their root form or lemma, for instance, converting *corpora* to *corpus* or *women* to *woman.* The NLTK lemmatizer is based on Wordnet, a large open-source lexical database developed by the Princeton University [92]. Stemming is an alternative approach removing affixes from word, turning them into their stem form. The main difference between lemmatization and stemming is that the former ensures a conversion to root words which are present in the dictionary, thus producing valid lexicon inputs. Since a first application explored in this thesis is the generation of a domain-specific lexicon, lemmatization rather than stemming is implemented in the NLP pipeline.
- **Stopwords Removal:** Stopwords are frequent words with low informative value [86] such as *the* or *a.* The NLTK library offers a pre-set list of 179 English terms including pronouns as *i*, *we*, and common verbs as *have* or *do.* The stopword lists is tailored to the domain-specific corpus with a term frequency-inverse document frequency (tf-idf) approach as presented below.

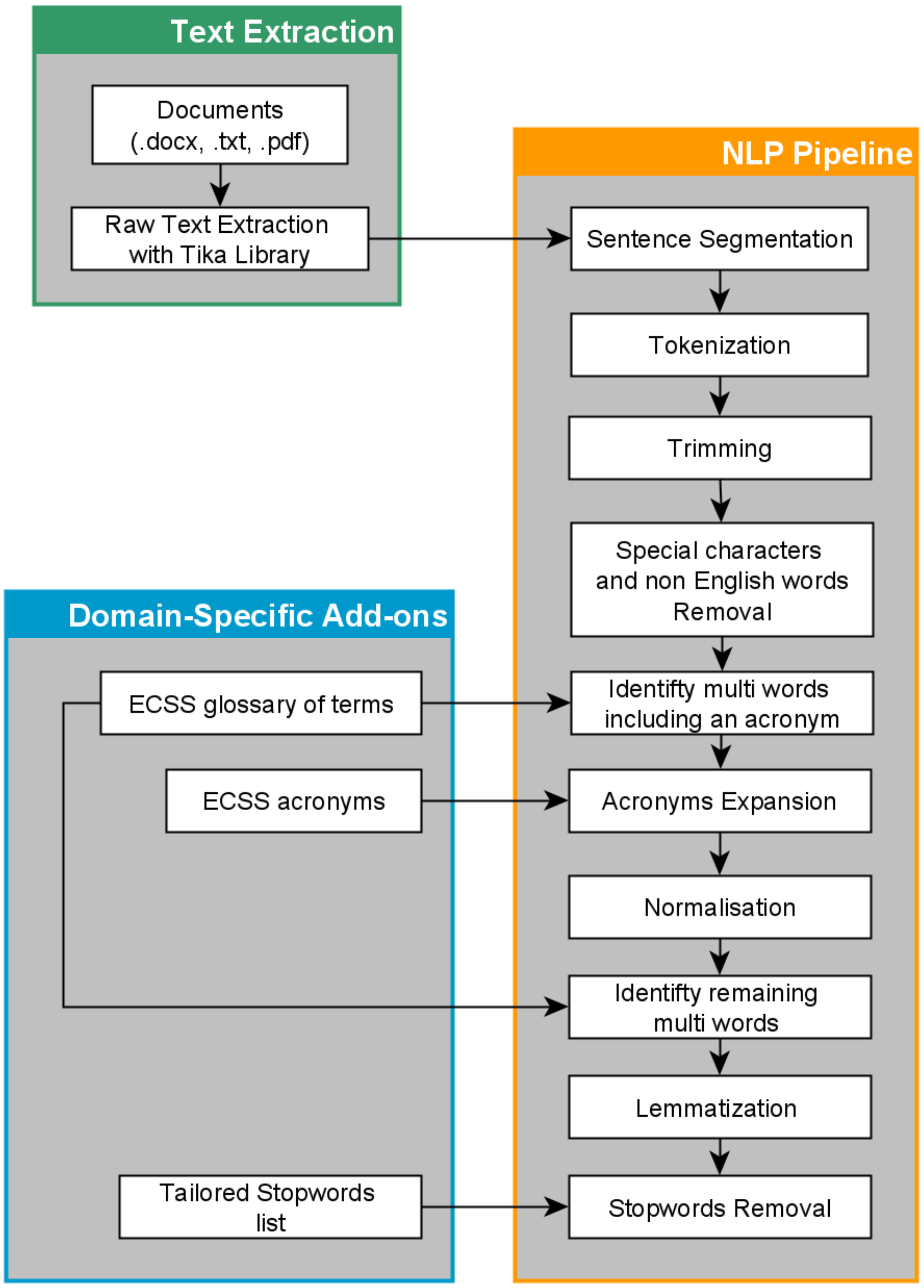
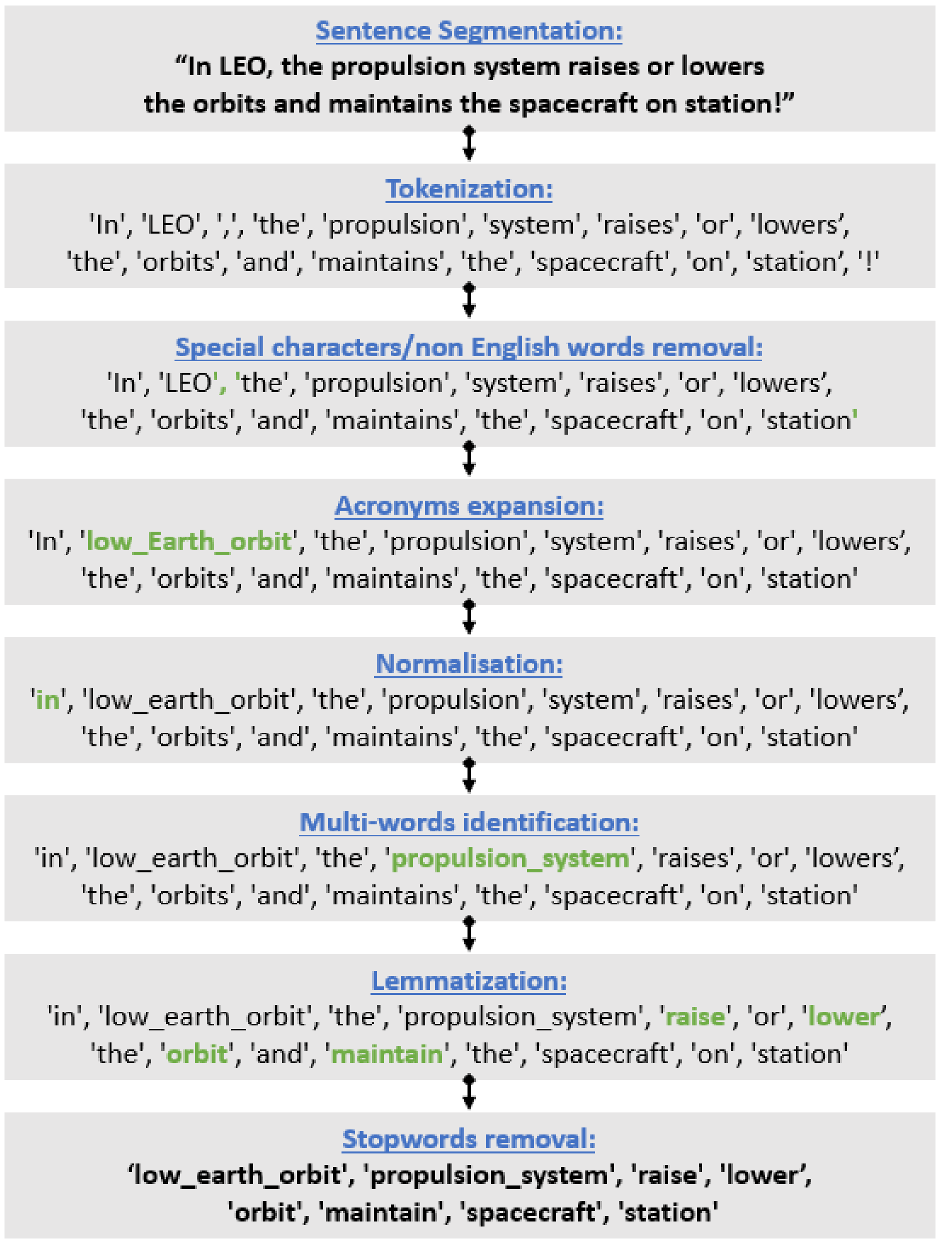


Figure 3.7: Overview of the domain-specific NLP pipeline

Multi-words Extraction

2,704 terms are exported from the 2017 ECSS glossary of terms [18]. Among these terms, 1,591 are non-redundant MWs and 80 include acronyms which are expanded.



Sentence Segmentation:
"In LEO, the propulsion system raises or lowers the orbits and maintains the spacecraft on station!"

Tokenization:
'In', 'LEO', ',', 'the', 'propulsion', 'system', 'raises', 'or', 'lowers', 'the', 'orbits', 'and', 'maintains', 'the', 'spacecraft', 'on', 'station', '!'

Special characters/non English words removal:
'In', 'LEO', 'the', 'propulsion', 'system', 'raises', 'or', 'lowers', 'the', 'orbits', 'and', 'maintains', 'the', 'spacecraft', 'on', 'station'

Acronyms expansion:
'In', 'low_Earth_orbit', 'the', 'propulsion', 'system', 'raises', 'or', 'lowers', 'the', 'orbits', 'and', 'maintains', 'the', 'spacecraft', 'on', 'station'

Normalisation:
'in', 'low_earth_orbit', 'the', 'propulsion', 'system', 'raises', 'or', 'lowers', 'the', 'orbits', 'and', 'maintains', 'the', 'spacecraft', 'on', 'station'

Multi-words identification:
'in', 'low_earth_orbit', 'the', 'propulsion_system', 'raises', 'or', 'lowers', 'the', 'orbits', 'and', 'maintains', 'the', 'spacecraft', 'on', 'station'

Lemmatization:
'in', 'low_earth_orbit', 'the', 'propulsion_system', 'raise', 'or', 'lower', 'the', 'orbit', 'and', 'maintain', 'the', 'spacecraft', 'on', 'station'

Stopwords removal:
'low_earth_orbit', 'propulsion_system', 'raise', 'lower', 'orbit', 'maintain', 'spacecraft', 'station'

Figure 3.8: Example of the NLP Pipeline process and output

Sequence of tokens corresponding to a MW expression are merged into a single token to capture their domain-specific meaning. The distribution of MWs found in the ECSS glossary, along with examples, is displayed in Table 3.3. Longer sequences are significantly less frequent that bigram (2-grams) and trigram (3-grams).

Table 3.3: Distribution of domain-specific multi-words

n-gram	**Number (%)**	**Sample**
2-grams	975 (61.3%)	heat dissipation, requirement traceability, solar array, space mission, verification matrix
3-grams	458 (28.8%)	end of life, critical crack size, signal to noise ratio, single event upset, space segment subsystem
4-grams	102 (6.4%)	mean time to repair, out of band emission, single event gate rupture, solar energetic particle event, total ionizing dose level
5-grams	46 (2.9%)	high efficiency particulate air filter, probability of correct attitude determination, solar cell anti reflection coating, total non ionizing dose level, yield design factor of safety
6-grams	8 (0.5%)	24 hour equivalent noise exposure level, environmental control and life support system, projected peak to peak insulation distance, run on and run off tabs, temporary degradation of space segment function
7-grams	1 (0.1%)	dominant and recessive states of canbus signals
9-grams	1 (0.1%)	oxygen concentration limit during the combustion of polymeric materials
Total	**1,591**	

Additional MWs were extracted from the corpus with the NLTK collocations library [86]. Collocations are sequence of words unusually often found together. The extracted MWs were verified manually. An additional 66 bigrams and 53 trigrams are thus discovered and added to the list of MWs to be identified within the NLP pipeline. A sample of these new terms is displayed in Table 3.4.

Table 3.4: Multi-words discovered through the collocation analysis

bigram	**trigram**
atlantic anomaly	active debris removal
collision avoidance	coronal mass ejection
lessons learned	synthetic aperture radar
optical imaging	technical risk assessment
study team	van allen belts

Acronyms Expansion

1,442 non redundant abbreviated terms are found in the list of ECSS abbreviated terms, released in 2017 [18]. These abbreviated terms, or acronyms, are extracted from the 64 active ECSS standards. These documents cover all disciplines related to space systems, and are distributed in 4 main branches: space project management, space product assurance, space engineering, and space sustainability. Many acronyms are found across several documents, and sometimes have different expansions. For instance, *CTE* is defined as *coefficient of thermal expansion* in the ECSS-E-ST-32-08C standard on space materials, while in another standard on radiation, ECSS-E-ST-10-12C, *CTE* is expanded as *charge transfer efficiency.* Similarly, *MOS* is defined as *margin of safety* in the ECSS-E-ST-32C standard on structural requirements, while it corresponds to *metal oxide semiconductor* in the standard related to radiation hardness assurance for electronic components, ECSS-Q-ST-60-15C.

At this processing level, the pipeline lacks disambiguation, and therefore cannot assess an acronym's context and associate it with the correct expansion. Thus the choice is made to only expand acronyms with a unique possible expansion. Acronyms corresponding to named entities such as “ESA”, “CERN” or “NASA” are also not expanded, and considered as single entities. Eventually, a manually curated list of 1,133 acronyms and their expansion is used to expand acronyms found in the corpus documents. Acronyms are expanded as a single token of several words.

Stopword list

To tailor the stopword list to a corpus, a term frequency-inverse document frequency (tf-idf) analysis of each corpus is run. Tf-idf is a statistical method evaluating the relevance of a word by assigning it a weight based on its frequency in a document but also among the whole corpus. The concept of idf was invented in 1972 by British computer scientist Karen Spärck Jones [93]. Tf-idf is defined by [94] as:

$$tf - idf_{t,d} = tf_{t,d} * idf_t \tag{3.1}$$

with

$$idf_t = \log \frac{N}{df_t} \tag{3.2}$$

where $tf_{t,d}$ is the frequency of term t in document d and idf_t the inverse document frequency of t among a corpus of N documents with a document frequency df_t corresponding to the number of documents where the term t is found. Thus the idf of a rare term is close to 1, whereas the idf of a frequent term tends toward 0. A threshold of 15% is arbitrarily set to only remove the tokens with the lowest tf-idf since these tokens have low informativeness value. A tailored stopword list is thus generated for each corpus. Table 3.5 displays samples of terms added to the tailored stopword lists.

Table 3.5: Sample of words with low tf-idf

Feasibility Reports	**Academic Publications**	**Books**	**Wikipedia Webpages**
tec (department name),	special issue,	springer,	reference,
internal,	international conference,	edition,	language,
noordwijk (location CDF),	workshop,	text,	licence,
content,	mails,	elsevier,	common creative,
appendix	abstract	mediagallery	edited

3.2.4 Corpora Statistics

Table 3.6 displays the corpus statistics after being processed by the NLP pipeline. A total of 2,278,193 sentences, including 22,609,372 tokens are collected. The academic publications represent 81.1% of this corpus, followed by the books (12.5%), the feasiblity reports (4.3%) and the Wikipedia pages (2.1%). Proportionally, the amount of MWs and acronyms found in the feasibility reports is significantly higher than for other corpora. This was to be expected as the ECSS and ESA work closely. Journal publications, books and Wikipedia pages are also written by authors worldwide and don't necessarily abide to European standards. Most of the multi-words identified are bigrams. This was also to be expected as a majority of ECSS MWs are bigrams as shown in Table 3.3. Table 3.7 displays the statistics obtained for full corpus, merging the four main text sources. Finally, Table 3.8 provides the statistics for the additional text sources.

Table 3.6: Main Corpora Statistics

	Feasibility Reports	**Academic Publications**	**Books**	**Wikipedia Webpages**
Nb. of documents	55	4,991	39	242
Corpus Size	542 MB	13.3 GB	1.16 GB	8.95 MB
Nb. of sentences	98,352	1,737,293	274,807	46,544
Nb. of tokens	1,264,948	15,151,242	2,520,398	472,397
Average Nb. of tokens per sentences	12.9	8.7	9.2	10.15
Dictionary Size	28,651	254,482	69,905	28,782
Nb. of MW	22,905 (667 unique)	82,537 (945 unique)	21,300 (896 unique)	3,630 (369 unique)
Top 5 MW	solar array, baseline design, design drivers, ground station, propulsion system	heat flux, space debris, mass flow rate, finite element, control system	launch vehicle, coordinate system, control system, ascending node, solar array	deep space, kinetic energy, electric propulsion, specific impulse, van allen
Nb. of acronyms expanded	37,719 (564 unique)	162,959 (825 unique)	31,850 (671 unique)	4,920 (363 unique)
Top 5 acronyms	S/C (Spacecraft), AOCS (Attitude and Orbit Control System), TRL (Technology Readiness Level), GNC (Guidance Navigation and Control), LEO (Low Earth Orbit)	GPS (Global Positioning System), GNSS (Global Navigation Satellite System), CFD (Computational Fluid Dynamics), GEO (Geostationary Orbit), LEO	GPS, LEO, GEO, OBC (On-board Computer), PCDU (Power Control and Distribution Unit)	GPS, UTC (Universal Time Coordinated), DRAM (Dynamic Random Access Memory), PV (Pressurized pressure Vessel), CPU (Central Processing Unit)

Table 3.7: Combined Main Corpus Statistics

	Combined Corpus
Nb. of documents	5,327
Corpus Size	15 GB
Nb. of sentences	2,157,117
Nb. of tokens	19,452,556
Average Nb. of tokens per sentences	9
Dictionary Size	305,364
Nb. of MW	130,779 (1,168 unique)
Top 5 MW	magnetic field, control system, space debris, remote sensing, heat flux
Nb. of acronyms expanded	237,448 (908 unique)
Top 5 acronyms	GPS (Global Navigation Satellite System), GNSS (Global Navigation Satellite System), CFD (Computational Fluid Dynamics), LEO (Low Earth Orbit) , GEO (Geostationary Orbit),

Table 3.8: Additional Corpora Statistics

	Mission Requirements	**ECSS Requirements**
Corpus Size	17 KB	10 MB
Nb. of sentences	100	27,016
Nb. of tokens	1,297	382,990
Average Nb. of tokens per sentences	13	14
Dictionary Size	477	11,478
Nb. of MW	23 (12 unique)	9,220 (1,020 unique)
Top 5 MW	ground segment, data handling, propulsion system, control system, bit error rate	application process, space segment, propulsion system, ground segment, execution notification
Nb. of acronyms expanded	63 (17 unique)	7,828 (646 unique)
Top 5 acronyms	TBC (To Be Confirmed), OBDH (On-Board Data Handling), AOCS (Attitude and Orbit Control System), TCS (Thermal Control System), S/C (Spacecraft)	PCB (Printed Circuit Board), OBCP (Original Baseline Cost Plan) , DRD (Document Requirements Definition), AOCS, RF (Radio Frequency)

3.3 Semi-structured Data Collection

An Engineering Model (EM) is the equivalent of a spacecraft design blueprint. As the Concurrent Engineering process is iterative, one EM may contain several design iterations of a same spacecraft. Within one iteration, several design options are usually studied to facilitate trade-offs decisions. For instance, one design option might have a propulsion system and a second option not. Each design option is linked to a spacecraft product tree, organised per subsystem and containing the parameters of each equipment as shown in Figure 3.9.

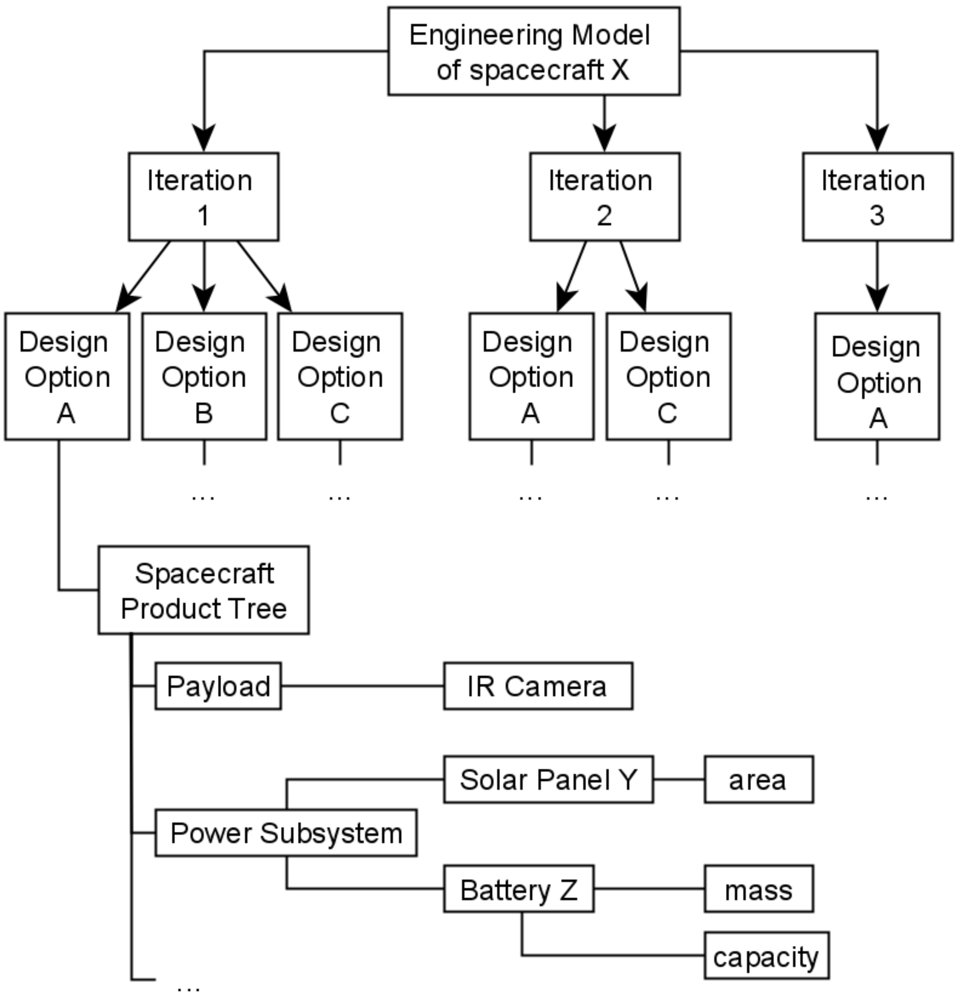


Figure 3.9: Unformalised Schema of a basic Engineering Model structure

3.3.1 The ECSS-E-TM-10-25A Technical Memorandum

EMs generated during feasibility studies by the ESTEC CDF team are based on the data model defined in the ECSS-E-TM-10-25A technical memorandum [72]. This memorandum defines the recommendations for model-based data exchange at the early phases of engineering design. In [95], the International Council on Systems Engineering (INCOSE) defines MBSE as:

> The formalised application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases.

System Markup Language (SysML) is the standard architecture modeling language for MBSE applications [4]. SysML is based on Unified Modeling Language (UML), the language in which the information model of the EMs is written. UML is furthermore introduced in Section 6.3.

A standardised data model is necessary for CE facilities to cooperate on concurrent design, enabling them to exchange models and definitions. The ECSS-E-TM-10-25A TM addresses the Phase 0 and A, but is not specific to space mission design and can be applied to various engineering fields. Although a technical memorandum is not a normative document, it is a first step towards standardisation. The data exchange protocol is split into two parts:

1. **The System Engineering Information Model**: The information model is called the System Engineering Information Model (SEIM). It is the core of the data model and defines the main object type and the relationships between them as shown on Figure 3.10. The SEIM is defined in the Annex A of the technical memorandum, and is provided in three formats: as a UML model, as an ISO 10303-11 EXPRESS SCHEMA specification, and as an XML Schema (XSD) specification. UML notation will be introduced in Chapter 4.

[4] `https://sysml.org/`

2. **The System Engineering Reference Data Library (SERDL)**: The SERDL is a reference data library storing predefined parameters that are often used in studies. For instance, expert disciplines such as *Thermal* or *System Engineering* are already defined in the SERDL and can easily be assigned to study participants. The SERDL is specified in the Annex B of the technical memorandum, an extract is shown on Figure 3.11.

As shown on Figure 3.10, the object types and relationships are not specific to space systems. The objects such as *Option*, *Equipment*, or *Element* could be applied to any engineering field. This flexibility however comes at the expense of the semantics.

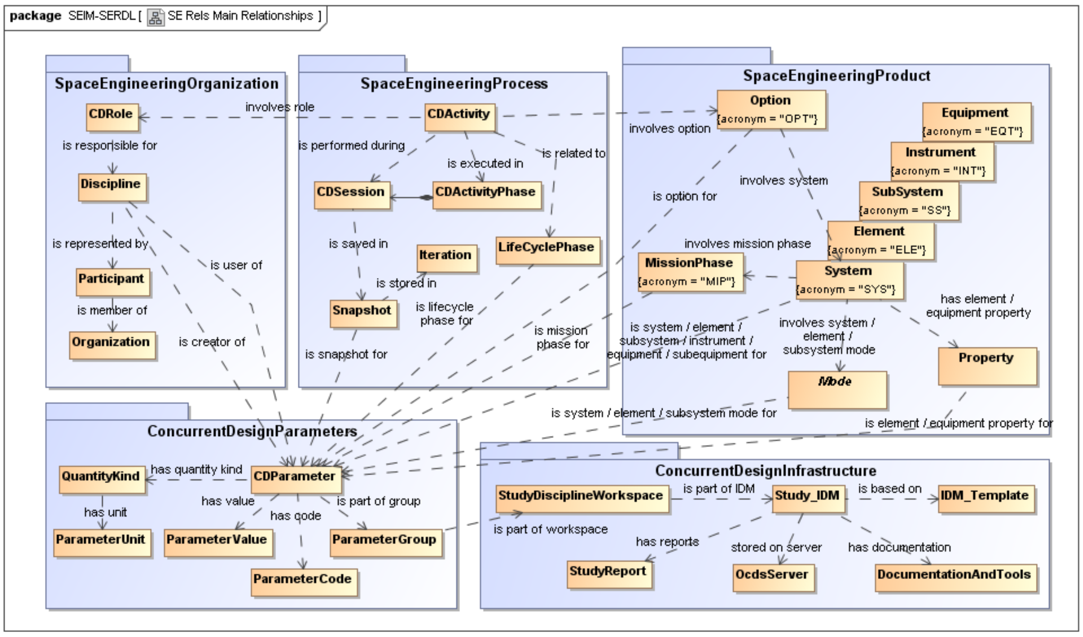


Figure 3.10: SEIM main information object types and relationships (informal UML class diagram) as seen in [72].

The two main software packages currently implementing the ECSS-E-TM-10-25A data model are the Open Concurrent Design Tool (OCDT) used by ESA and the Concurrent Design Platform CDP4 developed by the RHEA Group. Both software allow to visualise and update the product tree of the design options and iterations in near real-time. The software allows the experts to update the spacecraft design during design sessions and share the updates with the rest of the team.

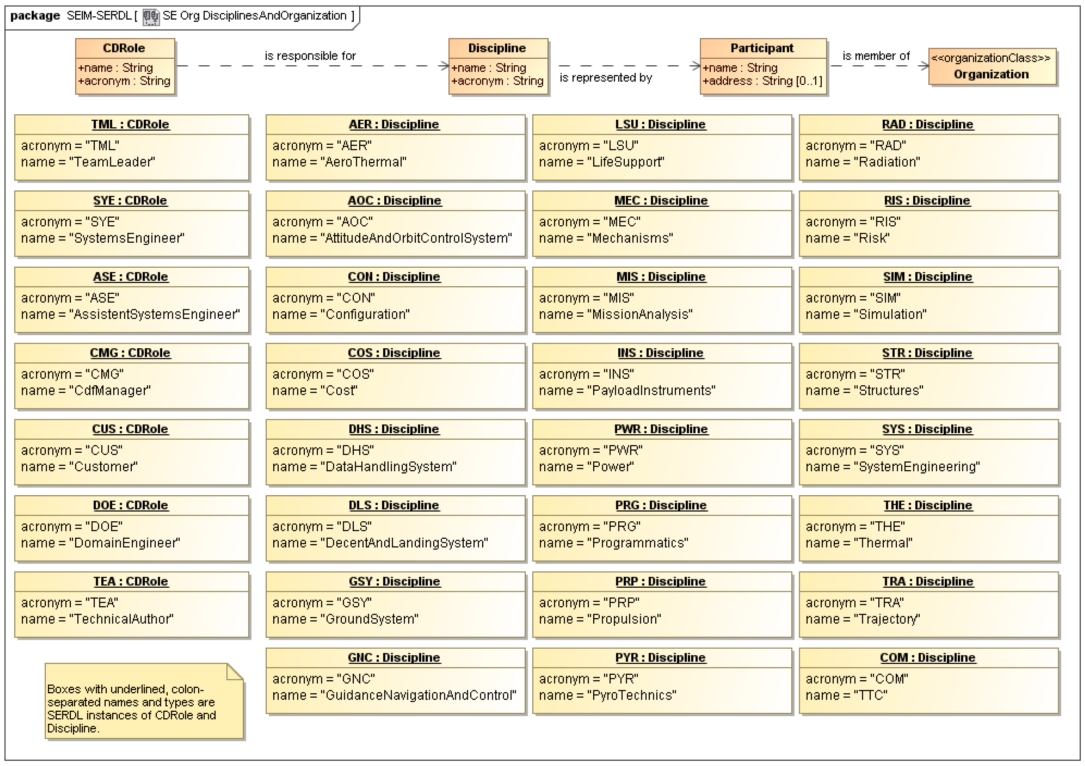


Figure 3.11: Example of SERDL content for roles and disciplines instances (UML class / object diagram) as seen in [72].

The spacecraft design is stored on a server. When a new element is defined, or a parameter value is updated, the modifications are pushed to the server. Then the experts can pull down the updated value and are therefore always up-to-date with the latest parameter values. In March 2021, RHEA and ESA announced the future deployment of a new tool merging OCDT with the CDP4, called COMET [96]. In this thesis, EMs are generated with the CDP4 Community Edition.

The Open Concurrent Design Tool (OCDT)

The OCDT is a client/server software package developed under an ESA contract. It has a three tier architecture based on *clients*, *web services* and *database server* as illustrated in Figure 3.12. The data is stored in a PostgreSQL database for persistent data storage. ConCORDE, standing for Concurrent Concepts, Options, Requirements and Design Editor, is the main end-user client tool. ConCORDE is implemented as a plug-in to Microsoft Excel. The interaction between the database and the clients is handled by the Web Services Processors (WSP). According to [97], the server can support CE teams of more than 20 experts, and the model content can be synchronised at least twice a minute. The OCDT software package is distributed under an ESA community open source software licence, and is only available to users qualifying as members of the OCDT Community. Only organisation that are legally based in an ESA member state or cooperating state can qualify.

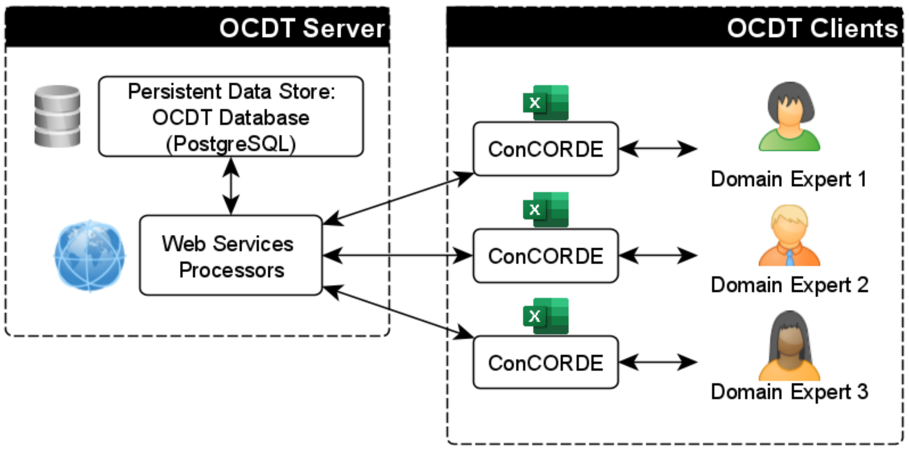
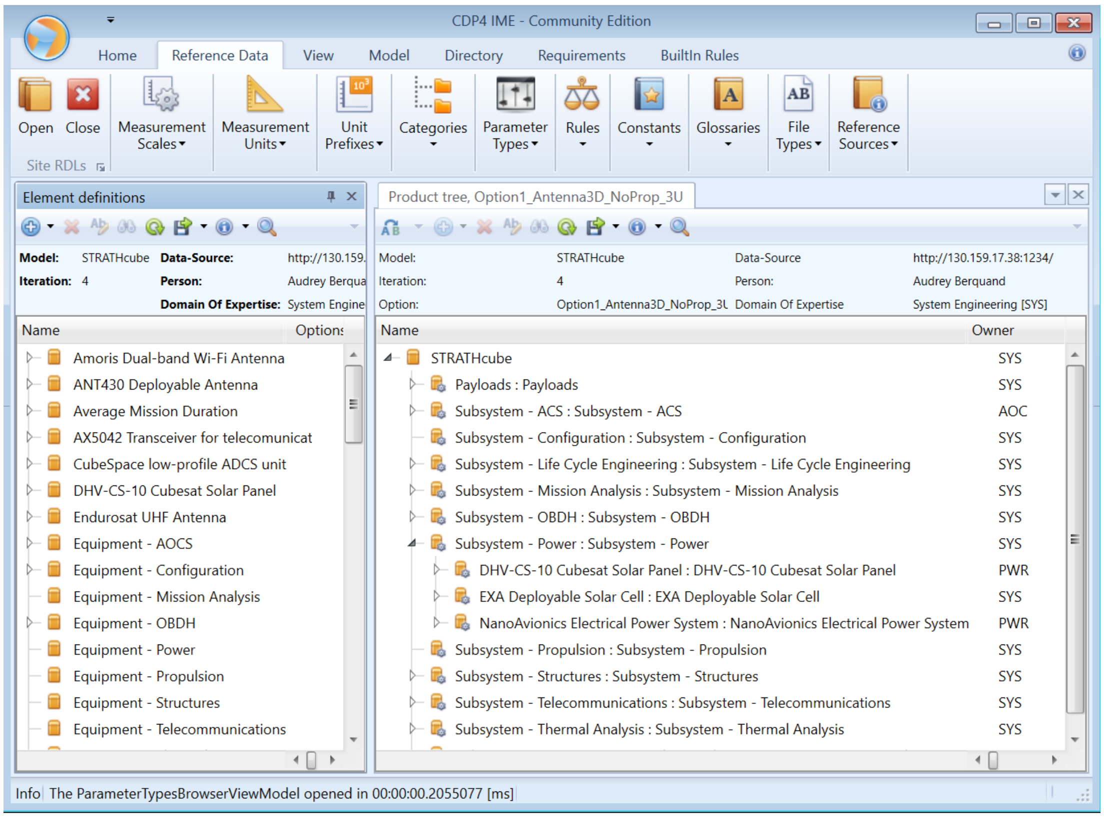


Figure 3.12: OCDT Architecture derived from [97]

CDP4 Community Edition

The CDP4 has a similar architecture as the OCDT, however it is an open-source tool. The CDP4 back-end is a database storing all the Engineering Models, transactions to and from the database are managed by the CDP4 Web Services [98]. The CDP4 client enables the domain engineers to open and edit the different EMs. The CDP4 client is implemented both as an Excel plugin and as a desktop application. The CDP4 client also offers different plug-ins for domain-specific tools, including an integration of the satsearch database[5] allowing a live search of equipment [99]. Finally, CDP4 implements the export of Engineering Models with the JSON protocol, as defined in the Annex C of the ECSS-E-TM-10-25A. Figure 3.13 shows the product tree for the final iteration of the STRATHcube study (presented below) visualised with the CDP4 desktop application.



Model: STRATHcube Data-Source: http://130.159.
Iteration: 4 Person: Audrey Berqua
Domain Of Expertise: System Engine

Name	Options
Amoris Dual-band Wi-Fi Antenna	
ANT430 Deployable Antenna	
Average Mission Duration	
AX5042 Transceiver for telecommunicat	
CubeSpace low-profile ADCS unit	
DHV-CS-10 Cubesat Solar Panel	
Endurosat UHF Antenna	
Equipment - AOCS	
Equipment - Configuration	
Equipment - Mission Analysis	
Equipment - OBDH	
Equipment - Power	
Equipment - Propulsion	
Equipment - Structures	
Equipment - Telecommunications	

Model: STRATHcube Data-Source http://130.159.17.38:1234/
Iteration: 4 Person: Audrey Berquand
Option: Option1_Antenna3D_NoProp_3U Domain Of Expertise System Engineering [SYS]

Name	Owner
STRATHcube	SYS
Payloads : Payloads	SYS
Subsystem - ACS : Subsystem - ACS	AOC
Subsystem - Configuration : Subsystem - Configuration	SYS
Subsystem - Life Cycle Engineering : Subsystem - Life Cycle Engineering	SYS
Subsystem - Mission Analysis : Subsystem - Mission Analysis	SYS
Subsystem - OBDH : Subsystem - OBDH	SYS
Subsystem - Power : Subsystem - Power	SYS
DHV-CS-10 Cubesat Solar Panel : DHV-CS-10 Cubesat Solar Panel	PWR
EXA Deployable Solar Cell : EXA Deployable Solar Cell	SYS
NanoAvionics Electrical Power System : NanoAvionics Electrical Power System	PWR
Subsystem - Propulsion : Subsystem - Propulsion	SYS
Subsystem - Structures : Subsystem - Structures	SYS
Subsystem - Telecommunications : Subsystem - Telecommunications	SYS
Subsystem - Thermal Analysis : Subsystem - Thermal Analysis	SYS

Info The ParameterTypesBrowserViewModel opened in 00:00:00.2055077 [ms]

Figure 3.13: Product Tree of the STRATHcube mission final iteration visualised with the CDP4 desktop application.

3.3.2 Engineering Models Library

The EMs library includes the models of NEACORE and STRATHcube, two feasibility studies led at the CCDS, the Strathclyde's concurrent engineering facility. The CDP4-Community Edition was used for both studies to store each iteration and design options.

[5]`https://satsearch.co/`

NEACORE

NEACORE, standing for Nanospacecraft Exploration of Asteroids by COllision and flyby REconnaissance, is an interplanetary mission concept involving up to six 12U CubeSats [35]. A CubeSat is a small satellite made of several cubic units. 1U corresponds to the basic unit of 10cm^3. A 3U CubeSat consists in 3 basic units stacked on top of each other. The primary scientific goal of the mission is to improve the knowledge of Near-Earth Asteroids (NEAs). To do so, two different sets of payloads were considered, involving a Light Detection and Ranging (LIDAR) device and a camera to measure the NEA's position, velocity and shape, as well as a spectrometer to map the surface composition of the asteroid. The mission is expected to last between 3 and 6 years, with a low-thrust propulsion system.

This study was led at the CCDS, in the frame of a first internal Concurrent Design Challenge meant to familiarise students with the CE process. 17 PhD students from the Departments of Mechanical & Aerospace Engineering (MAE) and Electronic & Electrical Engineering (EEE) participated in the challenge over a week in April 2019. The challenge structure was inspired from the ESA Academy yearly Concurrent Engineering Challenge.

Throughout the challenge, the design was iterated three times. All iterations were recorded in the engineering model. The final design iteration involves two options with different sets of payloads as the LIDAR, spectrometer and camera could not be accommodated on the same spacecraft. A common spacecraft platform was designed to support both design options. The mass budgets of both options for the final iteration are summarised in Table 3.9. The final design for the LIDAR option is displayed on Figure 3.14.

Table 3.9: Mass Budget of the NEACORE mission final design

Subsystem	**Mass [kg] LIDAR Option**	**Mass [kg] Spectrometer Option**
Bus	17.9	17.9
Instrument	2.69	1.44
Total Dry mass	20.59	19.34
Total Dry Mass with 10% margin	22.65	21.27
Total Wet Mass with 10% margin	24.24	22.86

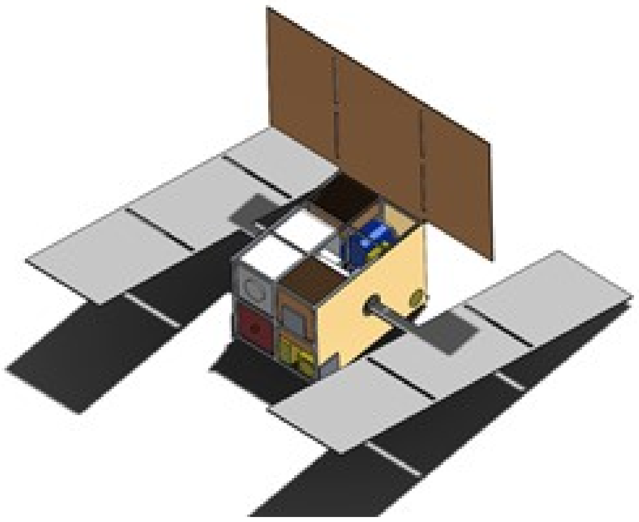


Figure 3.14: NEACORE Configuration with the LIDAR payload as seen in [35]

STRATHcube

The feasibility study for the STRATHcube mission took place remotely during the COVID-19 pandemic lockdown in May 2020. The CE approach was thus adapted to the local social distancing restrictions as presented in [34]. 29 students participated in this challenge. The mission concept was developed in support of an internal student-led application to the ESA Education 'Fly Your Satellite' programme. The latter program enables students to launch a CubeSat in Low Earth Orbit (LEO) from the International Space Station (ISS).

As specified by the ESA program, the volume of the candidate mission is limited to a 3U CubeSat [100]. Its primary scientific objective is to identify space debris with a 3D phased array antenna. A secondary objective is to perform measurements during re-entry using several heat flux, pressure sensors and UV/visual spectrometers [101]. The mission is expected to run for a minimum of 6 months.

The design was iterated three times during the feasibility study. Several design options were considered. A key mission driver was the size of the payload antenna which required a 3U platform and could not be accommodated with a propulsion system. After in-dept mission analysis, the propulsion system was discarded, greatly alleviating the cost and mass budgets. Eventually, the final iteration converged towards a single design option involving the 3D phase array antenna and a 3U platform shown on Figure 3.15. The primary and secondary payloads weight 2 kg. A 1.44 kg bus was designed to support the payloads. With a system margin of 10%, the final mass of the STRATHcube CubeSat is 3.78 kg. This is 5.6% below the maximum mass for a standard 3U CubeSat [100].

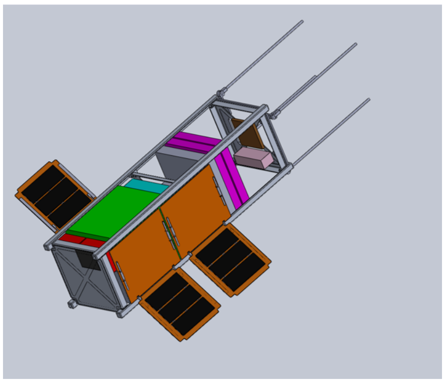


Figure 3.15: STRATHcube final iteration 3U configuration

As for the NEACORE study, participants were encouraged to report their subsystem equipment and parameters in the study's EM. Figure 3.13 displays the product tree for the final iteration of the STRATHcube study. The top element is the spacecraft, then subtypes corresponds to the different spacecraft subsystems. The *Subsystem - Power* branch is expanded to display its various equipment. In the left window, *Element definitions*, new components can be defined as well as their parameters. For instance a new solar panel can be defined, its volume and efficiency parameters specified. The *Element definitions* window is used to store equipment, as a virtual shelf. To be actually used on the spacecraft and appear on the product tree, they must be assigned to a subsystem as an *Element usage*.

3.4 Chapter Summary

This chapter presented the corpora, both unstructured and semi-structured, used to train the various methods and generate the results of this thesis. The text collection is the first open-source curated textual data set related to space systems. Table 3.10 summarises the usage of the unstructured data in Chapters 4 and 5. The semi-structured data set is used in Chapters 6.

Table 3.10: Unstructured data collection usage

Method		**Main texts**				**Additional texts**	
		Reports	Journal Publications	Books	Wiki Pages	Mission Req.	ECSS Req.
Lexicon Generation in 4.3	Analysis	X	X	X	X	-	-
Topic Modelling in 5.2	Training	X	X	X	X	-	-
	Case Study	-	-	-	-	X	-
Word Embedding in 4.4	Training	X	X	X	X	-	-
	Case Study	X	X	X	X	-	-
Document Embedding in 5.3	Training	-	-	-	-	-	X
	Case Study	X	-	-	-	-	-
Contextualised Embedding in 4.5	Training	-	X	X	X	-	-
	Case Study	-	-	-	-	-	X

Chapter 4

Towards a Space Systems Ontology

4.1 Chapter Overview

Large amounts of unstructured data have been accumulated over the past decades in the field of space systems engineering. To centralise data from heterogeneous sources into the DEA's Knowledge Graph, a common conceptual model, an ontology, is needed. Unfortunately, there is currently no standardised European space systems ontology, and creating an ontology is a tedious and lengthy task requiring several domain and ontology experts.

This chapter explores the possibility to semi-automatically generate the first elements of a space systems ontology based on the Ontology Learning Layer Cake framework. Following an introduction to the latter framework, statistical methods are applied to generate a domain-specific terminology from unstructured data. A new word embedding (word2vec) model is trained to then identify domain-specific synonyms and concepts. Lastly, a novel family of contextualised embedding models, based on the Transformer architecture, learns deep embeddings of concepts related to space systems.

4.2 Introduction to Ontology Learning

The universe of discourse represents all aspects of the world, concepts, facts addressed by a domain. A conceptual schema, or an ontology, formulates the universe of discourse by defining all of its relevant concepts, properties and relations. An ontology therefore contains the necessary knowledge to structure domain-specific textual data, and from there build the reasoning capacities of a virtual assistant. Unfortunately, there is currently no standardised European space systems ontology, although discussions were kick-started by the ESA Overall Semantic Modelling for System Engineering (OSMoSE) initiative in June 2019 [102]. The OSMoSE initiative aims to develop a space systems ontology to facilitate information exchange through interoperability between model-based infrastructures. Semantic interoperability is enabled by a common language: an ontology. Developing an ontology is however a lengthy and tedious process, involving several human domain experts, and therefore prone to human error and subjectivity.

The work presented here studies the possibility of accelerating the first phases of the ontology development by semi-automatically identifying terms, synonyms and concepts from unstructured data. The methodology relies on natural language processing and follows the steps of the Ontology Learning Layer Cake framework defined by Cimiano [103] and Buitelaar et al. [104] structuring automatic ontology generation. This bottom-up approach enables the identification of terms and concepts used in practice by engineers.

4.2.1 Knowledge Representation and Formal Logic

Lakemeter and Nebel [105] define Knowledge Representation (KR) as

> the area of Artificial Intelligence that deals with the problem of representing, maintaining, and manipulating knowledge about an application domain.

KR is, here, understood as a machine-understandable representation of reality, although it is also a medium for human expression and communication as well [106]. According to [106], KR provides a representation of the reality, and ontologies, by providing a committing set of terms and relations, are at the heart of KR systems.

Logic is required for representing knowledge and eventually allow reasoning according to Nebel [107]. Two common logic systems are First-Order Logic (FOL) and Description Logics (DL) [108]. In FOL, universal and existential quantifiers allow assertions about sets of objects as seen in the below logical axioms, 4.2 and 4.2. An axiom, also known as postulate or assumption, is a statement taken to be true. Axioms serve as starting point for further reasoning and arguments. With the FOL quantifiers, the predicate *All spacecraft have an orbit* in natural language becomes:

$$\forall X : Spacecraft(X) \Rightarrow hasOrbit(X) \tag{4.1}$$

and the predicate *There are (one or more) payload on spacecraft*:

$$\exists X : Spacecraft(X) \wedge hasPayload(X) \tag{4.2}$$

DL is a family of languages for KR. Most DLs are a subset of FOL, but they generally present a better compromise of expressivity and scalability than FOL [109]. DL systems are generally characterised by a T-box and an A-Box composing the Knowledge Base, as well as a R-box [110]. The T-box is a set of terminological axioms, it contains assertions about concepts, for instance, $Subsystem \sqsubseteq Spacecraft$ (Subsystem is a subsumption of Spacecraft) and $Contractor \equiv Stakeholder$ (Contractor is the equivalent of a Stakeholder). The A-box is a set of assertional axioms, it contains the instances of T-box concepts, for instance, $isMainPayload(TROPOMI, Sentinel-5)$ or $Sentinel-5 : Spacecraft$.The R-box is a role box and contains assertions about roles, for instance, $isLaunched \sqsubseteq hasOrbit$.

The logical basis of semantic web technologies, including the Web Ontology Language, OWL, is based on DL. This family of logic is in fact at the basis of several ontologies' formal models.

4.2.2 Ontologies

Definition

In 1993, Tom Gruber, a figure in the field of ontologies, defined the notion of an ontology as an "*explicit, formal specification of a shared conceptualization*" [111]. By *explicit*, Gruber means that all concepts must be defined, by *formal* that it must be machine-understandable, and by *shared* that there must be consensus.

Gruber refined his definition in the Encyclopedia of Databases Systems [112] in 2008:

> In the context of computer and information sciences, an ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members). The definitions of the representational primitives include information about their meaning and constraints on their logically consistent application.

Sabou in [113] concurs with the above adding that ontologies provide a common vocabulary field and concepts definition facilitating human-human, human-machine and machine-machine communications. To summarise, ontologies standardise the semantics of a field of knowledge. As underlined by Maedche and Staab, in [114], ontologies significantly contributed to the success of the Semantic Web. The latter is an extension of the World Wide Web through standards set by the World Wide Web Consortium (W3C) to structure the content of web pages into machine-readable data.

The basic elements of an ontology are the classes (entities), attributes and relationships. An example of a basic ontology for a space mission is displayed in Figure 4.1. The entity *Spacecraft* has two attributes, *status* and *missionObjectives*. Both of these attributes provide information on the mission's status, for instance, *operational*, *planned*, or *decommissioned*, and on its objectives, for instance, *wildfire monitoring* or *biomass measurements*. The *Spacecraft* entity is related to an *Orbit* entity. The latter has several attributes, *inclination*, *altitude*, *period*, and *type* defining the orbital parameters.

The *GroundStation* entity is linked to the *Spacecraft* entity through an *uplink* relationship, mimicking the upload of commands to the spacecraft. The latter is related to the *GroundStation* entity through a *downlink* entity representing the transmission of space-based measurements to the ground. Finally, the *GroundStation* entity is related to a *Country* entity by the *locatedIn* relation.

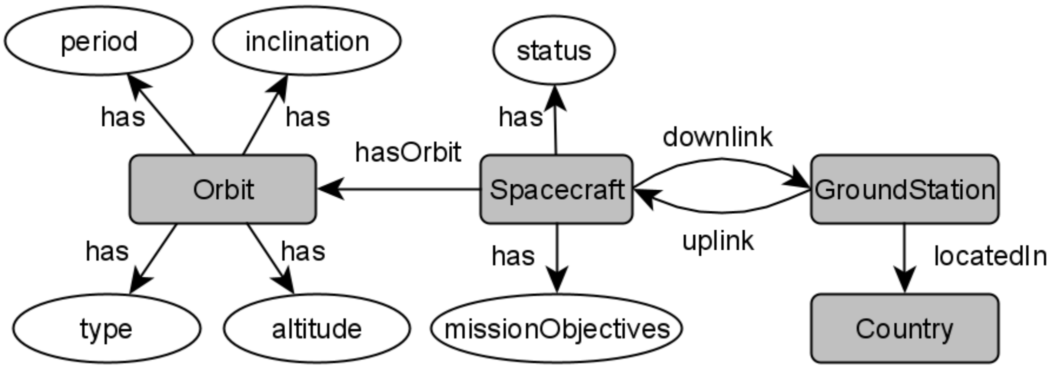


Figure 4.1: Example of a simple space mission ontology, entities are shown in grey rectangles, attributes in circles and relationships as lines.

Figure 4.2 displays an example of the above ontology, populated with data on the SMOS (Soil Moisture and Ocean Salinity)[1] mission.

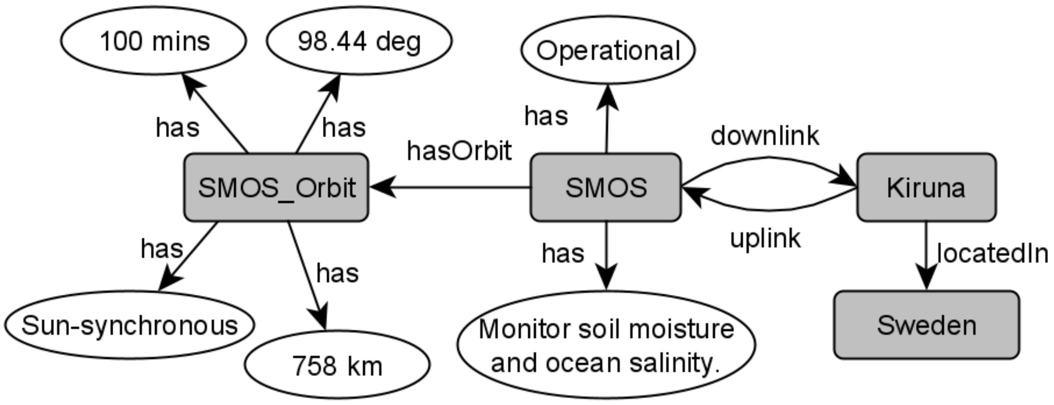


Figure 4.2: Example of a populated ontology

[1]https://directory.eoportal.org/web/eoportal/satellite-missions/s/smos

An ontology differs from a glossary of terms, a taxonomy or a thesauri as:

1. a glossary lists terms and their definition,
2. a taxonomy classifies concepts,
3. a thesauri provides the association and relations of concepts,
4. but only an ontology provides the rules and knowledge about which relations are allowed and make sense.

Ontology types

Roussey et al. [108] present two different types of ontology classification, either based on the language expressivity and formality, or based on the scope. According to the first classification, the goal is, in this thesis, to develop a Formal Ontology, relying on formal logic. This ontology type is the only one that contains logical definitions according to [108]. The standard of the World Wide Web, OWL is a formal ontology. Since the ontology developed here targets the domain of space systems, it is also a domain ontology. Domain ontologies are only applicable to a specific universe of discourse.

4.2.3 Ontology Learning Layer Cake

Coined by Maedche and Staab, in [114], Ontology Learning (OL) is a field of research encompassing the set of methods and techniques to build an ontology in a semi-automatic fashion. The work presented here addresses OL from text. The latter relies on text mining approaches to extract an ontology from textual data. The recent advances in NLP and ML have boosted OL from text as underlined by Lourdusamy and Abraham [115].

Cimiano [103] and Buitelaar et al. [104] attempted to structure the tasks of OL. The results is the so-called *Ontology Learning Layer Cake* shown in Figure 4.3. The output of the bottom layer feeds the upper layer, building step-by-step, an ontology. The first layer consists in the acquisition of the relevant basic building blocks, the terms. The second layer merges synonym terms. For instance, the words *satellite* and *spacecraft* are synonyms. Then, similar terms are clustered into concepts. The words *c-band* and *k-band* belong, for instance, to a *frequency band* concept. The *Relations* layer links the different concepts through various relationships. For instance, the *antenna* concept is linked to a frequency band. Lastly, the *Axioms* layer defines the axiom schemata and definitions. They are propositions which are always taken as true. As shown on Figure 4.3, each antenna x is connected to a frequency band y. Axioms can thus be used as a starting point for reasoning and deducing other axioms or for verifying the ontology integrity. Staad and Struder [116], Wong et al. [117], and Lehmann and Völker [118] all agree that the Layer Cake has become a reference to structure the steps of OL.

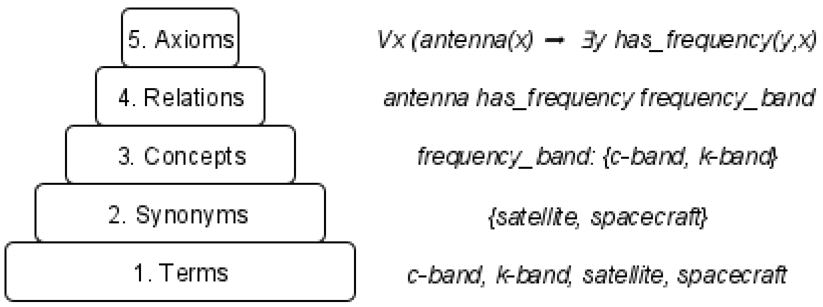

5. Axioms
∀x (antenna(x) → ∃y has_frequency(y,x)
4. Relations
antenna has_frequency frequency_band
3. Concepts
frequency_band: {c-band, k-band}
2. Synonyms
{satellite, spacecraft}
1. Terms
c-band, k-band, satellite, spacecraft

Figure 4.3: Ontology Learning Layer Cake derived from [103]

To summarise, the OL Layer Cake is a framework to semi-automatically build ontologies. To build an ontology from scratch, one has to follow all the steps described in Figure 4.3. Various methods can be applied to complete those steps.

4.3 Domain-Specific Lexicon Generation

The methodology and results from this section were partly presented in "Space mission design ontology: extraction of domain-specific entities and concepts similarity analysis" by Berquand, A., Moshfeghi, Y., and Riccardi, A. at the AIAA Scitech 2020 Forum invited session on Cognitive Assistants [11], and at the OSMoSE Space System Ontology first brainstorming workshop in June 2019 [102].

This section addresses the semi-automatic generation of the Term layer of the OL Layer Cake. The research presented here makes the following contributions:

1. The Ontology Learning Layer Cake framework is, for the first time, applied to a collection of texts related to space systems.
2. Statistical methods accelerate the construction of a domain-specific lexicon by identifying common concepts in the literature, but they are not sufficient to produce a complete lexicon.

Section 4.3.1 provides the background on previous similar work. Section 4.3.2 briefly summarises the approach which is detailed in depth in Section 4.3.3. Finally, the results of the domain-specific lexica generation and validation are presented and discussed in Section 4.3.4.

4.3.1 Background

In [119], Marciniak and Mykowiecka recover around 80% of known domain words from 1,200 hospital discharge documents with a frequency based method. Martin-Chozas and Calleja, in [120], apply two frequency-based methods, C-value and tf-idf, to extract a terminology related to Spanish labour law. The authors do note that the Part-Of-Speech (POS) tagging errors in the processing affect their results. Human validation will be used in this study to compensate similar POS errors. They also underline the advantages of using several extraction methods to highlight different features of the corpus. Finally, Ahmad and Gillam, in [121], describe an approach to extract keywords from a collection of documents related to nuclear physics, in the frame of ontology construction. The author do not filter nouns with Part-Of-Speech tagging but concludes that their keywords are mostly nouns. They produce a terminology based on a frequency analysis and a Weirdness Index filtering. their approach is similar to the one presented here.

4.3.2 Approach

Statistical methods of text mining are applied here to build the Term layer of the OL Layer Cake. The terminology related to space systems is semi-automatically extracted from the main text collection presented in Section 3.2.1.

The lexicon is to be generated in two steps: (i) domain-specific terms are identified from a frequency-based analysis enhanced with an additional filtering with either tf-idf or the Weirdness Index, (ii) the terms are validated against a general lexicon and a domain-specific dictionary. The approach is summarised in Figure 4.4 and further detailed in the methodology section.

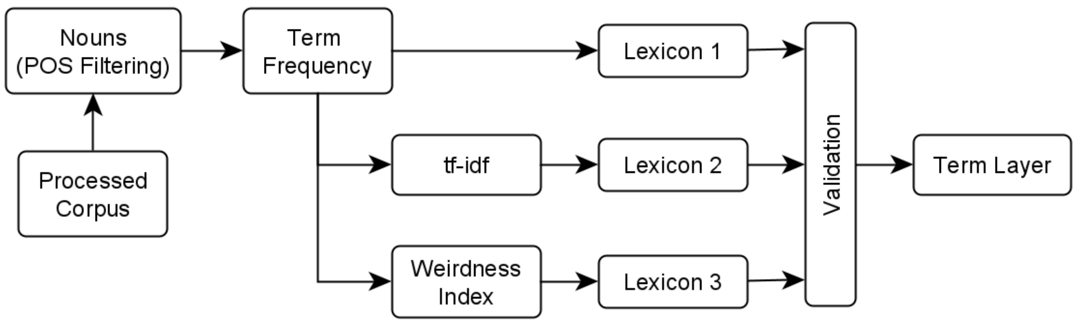


Figure 4.4: Methodology for lexicon generation

4.3.3 Methodology

Lexicon Generation:

In English, words are associated to a Part-Of-Speech (POS), also called syntactic category, addressing words with similar grammatical properties. The classic POS categories are *adjective*, *adverb*, *conjunction*, *determiner*, *interjection*, *noun*, *numeral*, *preposition*, *pronoun*, and *verb*. POS tagging is a common NLP task associating tokens with their syntactic category. Each category is divided into sub-categories identified by a unique tag by the NLTK POS tagger as shown in Table 4.1. Several non-grammatical tags such as *FW* (*Foreign Word*) or *SYM* (*Symbol*) can also be identified by the NLTK POS tagger.

Figure 4.5 displays the tags distribution for each subcorpus dictionary with the exception of categories representing less than 1% of words. Several categories are not represented as the POS tagging is applied to processed text and, for instance, all symbols and stop words have been filtered. To compare with a large corpus of general English text, the POS tags for the British National Corpus (BNC) dictionary are also compiled. The BNC[2] is a 100 million word collection from various general British English sources, ranging from regional newspaper to popular fiction and academic books. It is managed by Oxford University Computing Services on behalf of the BNC Consortium. As shown on Figure 4.5, the distributions are similar for the domain-specific and general dictionaries, with the *noun* being a predominant syntactic category.

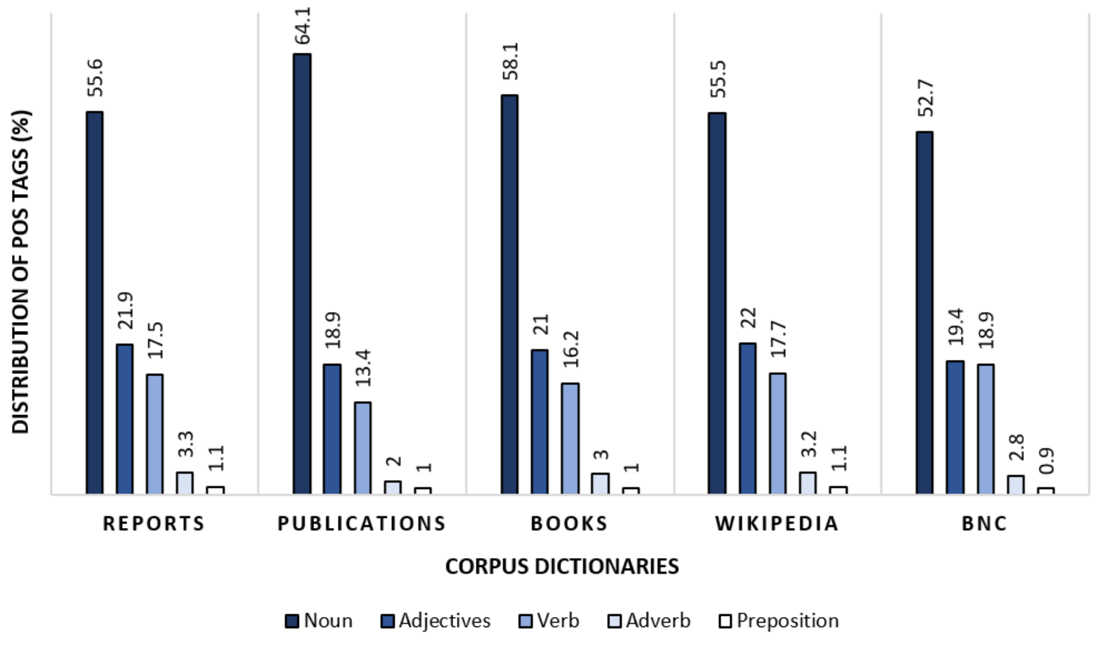


Figure 4.5: Distribution of the Part-of-Speech tags found in the study corpora and the British National Corpus (BNC).

[2]`http://www.natcorp.ox.ac.uk/`

Table 4.1: Part-of-Speech tags of the NLTK POS Tagger

Part of Speech	**Tag**	**Definition**	**Examples**
Adjective	JJ	adjective	*obscure, hot,*
	JJR	comparative adjective	*easier, cleaner*
	JJS	superlative adjective	*largest, greatest*
Adverb	RB	adverb	*safely, remotely*
	RBR	comparative adverb	*lesser, heavier*
	RBS	superlative adverb	*furthest, worst*
	WRB	wh-adverb	*however which*
Conjunction	CC	conjunction, coordinating	*altogether, either*
Determiner	DT	determiner	*another, every*
	PDT	pre-determiner	*both, many*
	WDT	wh-determiner	*that, what*
Interjection	UH	interjection	*wow, ah*
Noun	NN	common noun, singular or mass	*satellite, rocket*
	NNS	common noun, plural	*orbits, subsystems*
	NNP	proper noun, singular	*airbus, esa*
	NNPS	proper noun, plural	*Great Lakes, Andes*
Numeral	CD	numeral, cardinal	*seven, million*
Preposition	IN	preposition, subordinating	*upon, above*
	TO	"to" as preposition	*to*
Pronoun	PRP	personal pronoun	*she, ours*
	PRP$	possessive pronoun	*her, mine*
	WP	wh-pronoun	*whom, whosoever*
	WP$	possessive wh-pronoun	*whose*
Verb	VB	verb, base form	*disrupt, authorise*
	VBD	verb, past tense	*wrapped, filtered*
	VBG	verb, present participle or gerund	*stopping, refueling*
	VBN	verb, past participle	*fixed, registered*
	VBP	verb, present tense, not 3rd person singular	*launch, measure*
	VBZ	verb, present tense, 3rd person singular	*observes, uploads*
	MD	modal auxiliary	*should, might*
Other	EX	existential there	*there*
	FW	foreign word	*bonjour soleil*
	POS	genitive marker	*'s*
	LS	list item marker	*A.a, 1.2*
	RP	particle	*on, into*
	SYM	symbol	*%, $*

Concepts are primarily nouns, the lexicon terms are thus limited to singular and plural common nouns, respectively tagged as *NN* and *NNS*. Proper nouns, *NNP* and *NNPS*, are filtered as they represent named entities that could be seen as the instantiation of concepts. For instance, *Sentinel-1* is the name of a satellite, an instantiation of the concept *spacecraft*. Verbs will become of interest to link space systems concepts at a later stage. As shown in Figure 4.6, the majority of *noun* tags corresponds to singular and plural common nouns. The singular and plural proper noun actually represent less than 5% of *noun* tags. The BNC dictionary has a higher proportion of plural nouns as the domain-specific corpus was subject to lemmatisation in the NLP pipeline. The lemmatisation process is however not perfect and has failed to map several plural terms to their singular lemma.

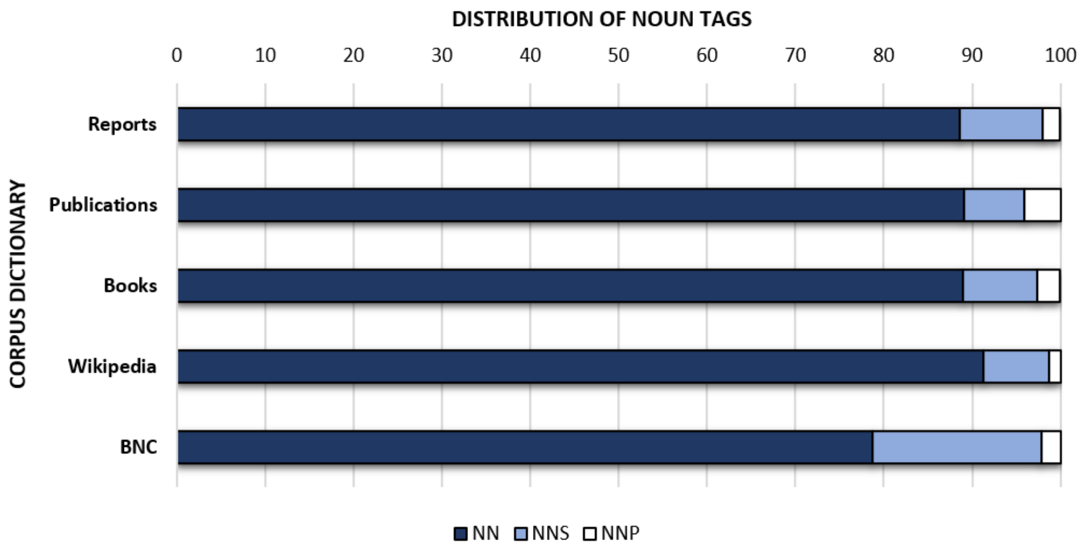


Figure 4.6: Distribution of common and proper nouns tags found in the corpora

A key assumption for generating the lexica is that a document related to a specific topic will more frequently use domain-specific words. Thus a first lexicon is generated with a frequency analysis. This analysis is however not sufficient to ensure that the extracted terms are domain-specific and not just common words frequently used in English. To remove frequent but common words, an additional filtering is done either with a tf-idf analysis or the Weirdness Index (WI) [121].

This method yields three lexica:

1. **Lexicon 1: Term Frequency**

   This lexicon includes all nouns with a frequency above a set threshold.

2. **Lexicon 2: Term Frequency + tf-idf**

   This lexicon is built on top of the first lexicon, filtering tokens with tf-idf scores below a set threshold. Tf-idf was previously defined in Section 3.2, Equation 3.1. In the NLP pipeline, the tf-idf scores are used to filter the terms with low tf-idf scores and therefore low informativeness value, while for the Terms layer, words with a tf-idf above a set threshold are kept.

3. **Lexicon 3: Term Frequency + Weirdness Index (WI)**

   This third lexicon is built on top of the first lexicon with an additional filtering based on the Weirdness Index (WI). This method is presented in [121] applied to a nuclear physics corpus. This index allows to compare the use of a word, based on its frequency, between a domain-specific corpus and a large corpus representing the general language. In this case, the latter corpus used is the BNC. The index, *WI*, is defined in [121] as:

   $$WI = \frac{N_G f_S}{(1 + f_G) N_S} \tag{4.3}$$

   where f_S is the frequency of the word in the specialised corpus, f_G its frequency in the general corpus, the BNC, and N_S, N_G are respectively the number of tokens in the specialised and general corpus. Only terms with a WI above a set threshold are kept.

This methodology is applied to each subcorpus to highlight the difference in language per source. A final human-curated lexicon is generated combining all four text sources. To select the cut-off thresholds, the variations of the z-score, also called standard score, are computed for each corpus. The z-score is a statistical measurement to position a value with respect to the average of a group of values. A z-score of 0 corresponds to the average, a score of 1 corresponds to a value one standard deviation from the average. The z-score is defined as:

$$z_i = \frac{x_i - \mu}{\sigma} \tag{4.4}$$

where z_i is the z-score of a value, μ the average of the values sample, and σ the standard deviation.

Lexicon validation:

As underlined in [115], there is currently no gold standard evaluation for OL. To evaluate the quality of the lexica generated, two comparisons are done with a general English lexicon and a domain-specific dictionary:

1. **Comparison to a general English lexicon: WordNet.** WordNet is an open-source large lexical database of English, developed by the Princeton University [92]. Calling Wordnet a 'lexicon', a list of words, is a misuse of language as the authors developed a tool that resembles a thesaurus, gathering synonyms into synsets. WordNet is however used as a mere lexicon for this validation step. The assumption is that classic English thesauri such as WordNet do not contain all domain-specific terms. Therefore, to represent a domain-specific lexicon, the final lexicon of candidate entities should minimise the number of entities also found in WordNet.
2. **Comparison to a domain-specific dictionary: ECSS Terms and Acronyms.** The ECSS glossary of terms and definitions is a human validated dictionary of terms related to space systems [122]. This glossary is already partly used in the NLP pipeline to identify multi-words. The glossary is a human validated space systems lexicon. The ECSS list of abbreviated terms [18] complements the multi-words lexicon for this validation steps. The final lexicon of candidate entities automatically extracted should maximise the number of entities also found in the ECSS documents.

4.3.4 Results & Discussion

Threshold selection

For each corpus's dictionary, the z-score of each term's frequency is computed. The distributions of z-scores obtained for the book corpus is displayed Figure 4.7. A similar trend emerges for all corpora, where a majority of words are between −0.5 of the standard deviation and 0 (corresponding to the frequency average). Similar figures are obtained with the tf-idf and WI-based lexica. To focus on the terms with highest frequency, tf-idf and WI scores, the threshold is set to a null z-score, thus all terms respectively above the average frequency, tf-idf or WI score are kept.

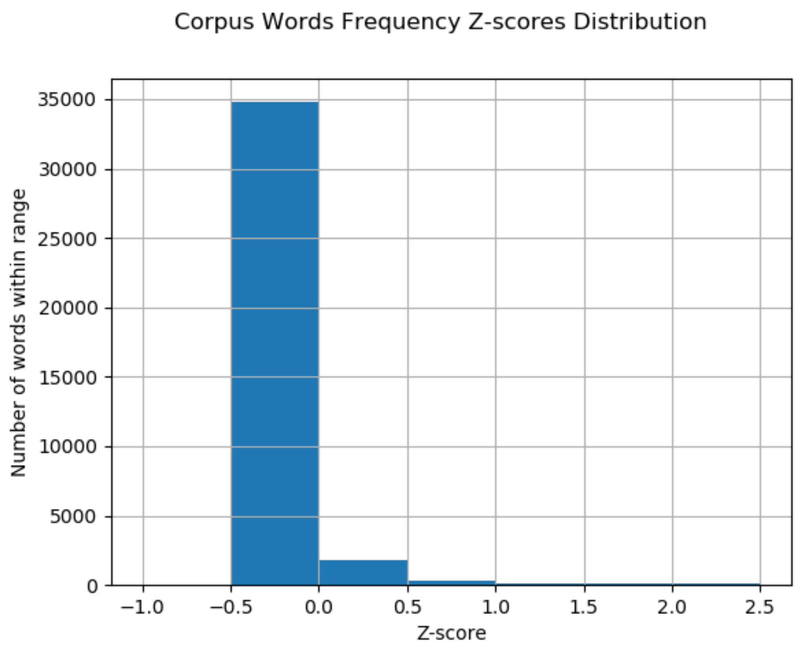


Figure 4.7: Frequency-based Z-score distribution for the book corpus

Lexica Generation

Tables 4.2, 4.3, and 4.4 respectively display the lexica obtained with the frequency analysis, tf-idf, and the WI scoring. The largest frequency lexica is extracted from the Publications corpus, which is also the largest corpus containing 151, 972 unique common nouns. Similar words are found within the top terms of each lexicon, suggesting a high overlap of the frequency lexica' content. The redundancy in frequency lexica is confirmed by Figure 4.8a. Since the tf-idf and WI lexica are built from the frequency lexicon, they contain less terms but they do push forward terms that are more specific to each corpus. The tf-idf score is computed with respect to all four corpora, as shown in Figure 4.8b, this results in a lexicon specific to each corpus. For instance, the top words of the reports corpus include words such as *requirement*, *margin*, *design driver* and *preliminary* (design) which correspond to a corpus focused on describing design options answering to a set of requirements and mission objectives. On the other hand, the WI lexica highlight domain-specific multi-words, 2-grams and higher, which are less likely found in a common corpus. As seen in Table 4.4, there is a higher density of multi-words found in the top 10 words, such as *global positioning system* or *low earth orbit.* The WI lexicon does also succeed in highlighting the specifics of each corpus, with words such as *trade-off* or *technology readiness level*, found in the reports corpus's lexicon. However, by focusing on highly uncommon words, the WI-based lexicon skips basic terms such as *mission* or *satellite* that are necessary to build an ontology of space systems. Several words, such as *arise* or *surveying*, were misclassified as common nouns and eluded the POS filtering. Removing these, as well as additional noise, requires human annotators.

Table 4.2: Frequency lexica. Words in blue appear in more than one lexicon.

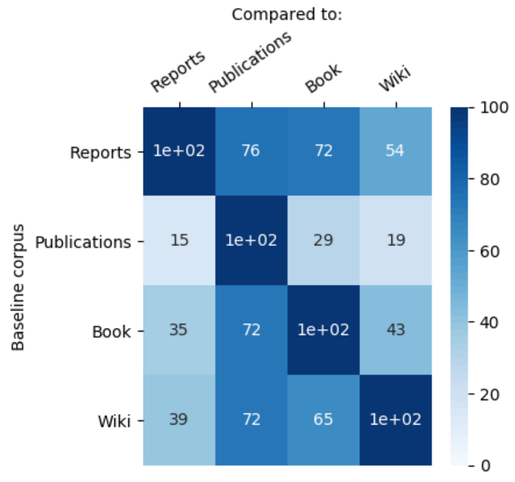
Corpus	Reports	Publications	Books	Wikipedia
Dict. Size	28,651	254,482	69,905	28,782
Unique Nouns	15,185	151,972	37,644	15,436
Lexicon Size	1,367	6,957	2,812	1,862
Average (std)	49 (363.2)	58 (1025.6)	38 (403.7)	17 (95.9)
Top 10 words	**mission**	model	**satellite**	**space**
	mass	**time**	**system**	**system**
	system	control	**orbit**	**satellite**
	design	**system**	**spacecraft**	**spacecraft**
	spacecraft	**method**	**space**	energy
	requirement	result	**time**	earth
	power	**space**	**data**	**power**
	orbit	flow	**mission**	field
	option	value	**design**	**time**
	cost	**data**	**power**	**mission**
Bottom 5 words	close-up	bnnts	grass	de-orbit
	projection	beam plasma	mean time to failure	surveying
	radiosisotope heater unit	turbulence flame	cryocooler	barycenter
	hydrogen	gpls	coopetition	gradiometers
	yelle	seaplane	exceedance	multi junction

Table 4.3: Tf-Idf lexica. Words in blue appear in more than one lexicon.

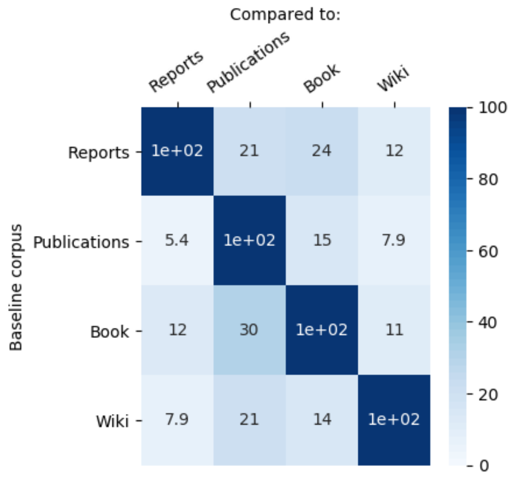
Corpus	**Reports**	**Publications**	**Books**	**Wikipedia**
Frequency Lexicon Size	1,367	6,957	2,812	1,862
Tf-idf Lexicon Size	339	1,323	661	493
Average (std)	0.06 (0.15)	0.06 (0.17)	0.036 (0.11)	0.017 (0.045)
Top 10 words	**requirement**	model	**satellite**	earth
	minimum	level	**orbit**	material
	orbit	**science**	**space**	**space**
	propellant	control	altitude	radio
	margin	**space**	impact	wave
	payload	flow	**requirement**	**satellite**
	design drivers	algorithm	**science**	right
	potential	orbital	linear	sail
	degree	profile	operating	cite
	preliminary	**satellite**	accuracy	permanent
Bottom 5 words	outer	arise	fibre	curiosity
	swing by	solenoid	sidereal day	portrait
	shell	orlando	telecontrol	modeling
	paint	coronal mass ejection	acceptance	joule
	stabilisation	enter	education	ambiant

Table 4.4: WI lexica. Words in blue appear in more than one lexicon.

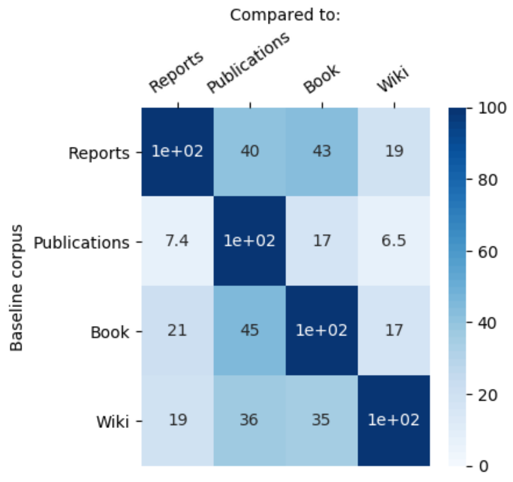
Corpus	Reports	Publications	Books	Wikipedia
Frequency Lexicon Size	1,367	6,957	2,812	1,862
WI Lexicon Size	270	1,480	551	263
Average (std)	19,694 (63,500)	8,589 (31579)	5,886 (21,815)	2,145 (8,117)
Top 10 words	attitude and orbit control system	**global positioning system**	**maneuver**	**global positioning system**
	trade-off	aerosp	**global positioning system**	flyby
	technology readiness level	computational fluid dynamics	**geostationary orbit**	dynamic random access memory
	design drivers	scramjet	**low earth orbit**	**maneuver**
	perigee	combustor	on-board	pressurised pressure vessel
	guidance navigation and control	**real time**	**radio frequency**	deep space
	electric propulsion	**geostationary orbit**	on-board computer	**radio frequency**
	downlink	root mean square	**downlink**	delta-v
	low earth orbit	**low earth orbit**	**real-time**	alternating current
	ground station	finite element	power control and distribution unit	central processing unit
Bottom 5 words	yelle	near space	astrophys	download
	telemetry	protoflight model	fiber	payload
	deg/sec	human machine interface	gyroscope	radioisotope
	antenna	international terrestrial reference system	dayside	fresnel
	ccds	compressor	mbps	joule



(a) Frequency lexica



(b) Tf-Idf lexica



(c) WI lexica

Figure 4.8: Overlapping of lexica

Lexica Validation

Figures 4.9 and 4.10 respectively display, for each lexicon and each corpus, the percentage of tokens also found in WordNet and in the ECSS glossary of terms and acronyms. As previously described, the assumption is that a valid space system set of terms would minimise the similarity with a general lexicon such as WordNet and maximise the similarity with a domain-specific lexicon such as the ECSS glossary of terms and acronyms. Based on Fig.4.9, the percentages of terms found in both corpora and WordNet are high for all frequency-based and tf-idf lexica. The WI lexica have significantly less in common with the WordNet lexicon, and is thus confirmed as the most efficient method for extracting domain-specific terms.

On the other hand, the similarity with the ECSS-based lexicon was low. The maximums for each lexicon types were achieved for the ESA reports corpus. The latter is the most likely to adhere to the European standards, and therefore to include ECSS terms and acronyms. The WI approach had the most overlap with the glossary, demonstrating the approach's efficiency in highlighting highly technical terms. Yet, the maximum similarity observed was only of 25.9%. Basic words such as *orbit* and *mass*, but also complex terms such as *line of sight* or *design driver*, were found in the automatically generated lexica but not in the ECSS glossary. Table 4.5 provides additional examples of the overlap and separation.

The semi-generated lexica and the ECSS glossary of terms have a too low overlap to demonstrate that statistical methods are sufficient to reconstruct a complete domain-specific lexicon. These results highlights the difference between the bottom-up and top-down approaches, and the fact that the ECSS glossary might not reflect the terminology used in practice. The terms discovered in the texts could thus be used to enrich lexica generated with a top-down approach.

Table 4.5: Sample of terms only found in the semi-automatically generated lexicon or in the ECSS glossary, as well as terms found in both.

Terms only found in the semi-automatic lexica	**Terms found in both lexica**	**Terms only found in the ECSS lexicon**
acceptance review,	baseline,	nominal condition,
center of mass,	design,	on-board memory,
delta v,	lifetime,	product tree,
ground station,	payload,	software unit,
perigee	propellant	transmitter

Merged Lexicon

The four frequency-based lexica, which include all terms from the tf-idf and WI lexica, are merged into a single combined lexicon. It could have been argued that only the WI lexica should be merged as they shared the most similarities with the ECSS glossary.

However, these lexica also failed to cover basic terms such as *mission* and *satellite*, found in the frequency-based lexica, which are highly relevant to a space systems ontology.

The terms are sorted in alphabetical order, discarding the frequency values as the corpora have disproportionate sizes. The final curated lexicon contains 3,511 terms, validated by human annotators. This lexicon includes 22% of the ECSS glossary terms. It is used as input for the Synonym layer implementation presented in Section 4.4.

4.3.5 Conclusions

This section outlined a first application of the OL Layer Cake to a corpus related to space systems. The methodology for the Terms layer, based on a frequency analysis, complemented with a tf-idf and a Weirdness Index filtering, semi-automatically identify key terms in space systems engineering. The frequency-based lexica extract fundamental and basic terms, while the tf-idf and WI methods respectively highlight corpus-specific and complex words (n-grams). With the POS misclassification and remaining noise, human domain-experts are still required to validate the lexicon terms.

These statistical methods are however not sufficient to reconstruct a complete domain-specific lexicon as the semi-generated lexica have a low overlap with the domain-specific glossary. On the other hand, this also means that the manually curated lexicon does not integrate terms commonly found in the domain-specific literature. The bottom-up statistical approach presented here could therefore be used to complement top-down methods, or to promptly lay the basis for a new domain-specific lexicon.

In future work, additional text sources could be added to ensure the completeness of the lexicon. When an official space systems ontology is released it will become possible to more precisely quantify the completeness of the approach proposed here. In the meantime, simple statistical methods as the ones presented here provide a robust base for the construction of domain-specific terminology.

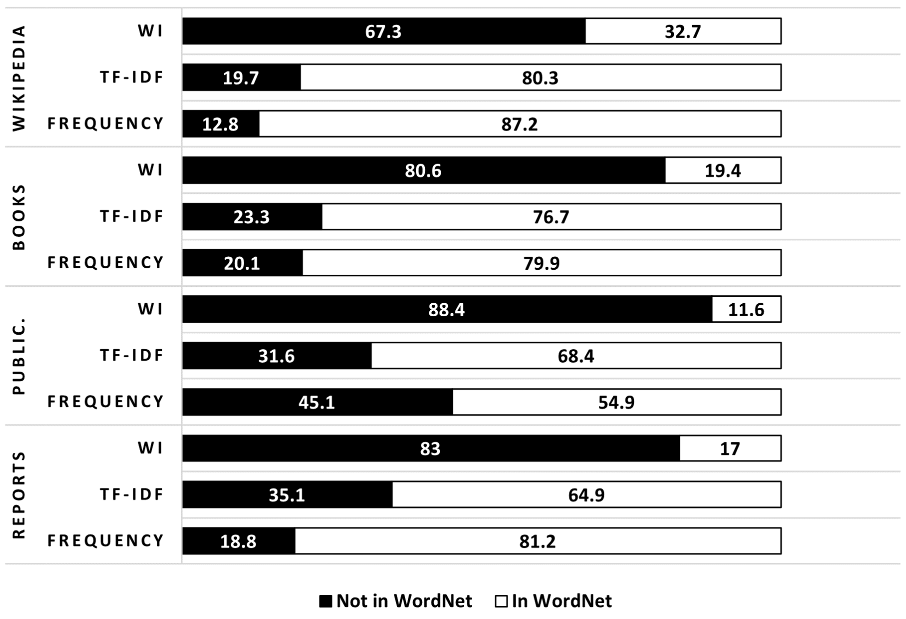


Figure 4.9: Comparison between WordNet and all automatically generated lexica

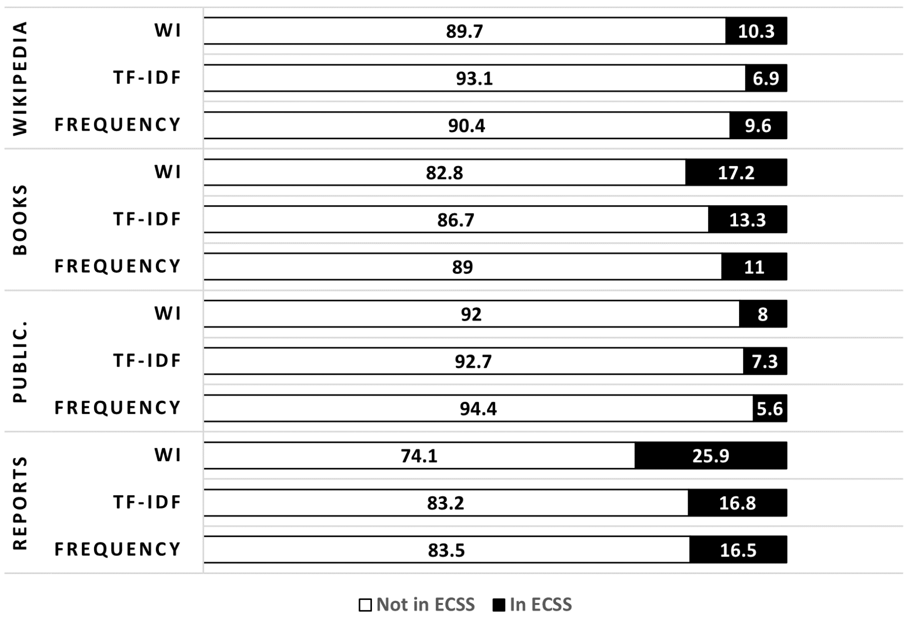


Figure 4.10: Comparison between the ECSS glossary of terms and acronyms and all automatically generated lexica

4.4 Synonyms and Concepts with Word embedding

The approach and methodology from this section were presented in "Space mission design ontology: extraction of domain-specific entities and concepts similarity analysis" by Berquand, A., Moshfeghi, Y., and Riccardi, A. at the AIAA Scitech 2020 Forum invited session on Cognitive Assistants [11], and at the OSMoSE Space System Ontology First brainstorming workshop in June 2019 [102].

In Section 4.3, a domain-specific lexicon is semi-automatically derived from a corpus of textual data. This lexicon lays the first layer of an Ontology Learning Layer Cake. As shown in Figure 4.3, the next layers are related to synonyms and concepts. These steps refine the lexicon by merging synonym entities and clustering terms into concepts. For instance, the words *astronaut* and *cosmonaut*, or *satellite* and *spacecraft* are interexchangeable synonyms. The words *c-band*, *l-band*, and *p-band* all belong to the concept of *communication wavelength.* Available British thesauri such as WordNet provide a basis for similar concept identification. However, these thesauri are usually not adapted to domain-specific terminologies.

The work presented here identifies the synonyms and concepts from a domain-specific lexicon with a word embedding model trained for the first time on a corpus of documents related to space systems. Word embedding methods map a word to a representation vector depending on its context. Similar terms have similar context and thus representation vectors. The word representations learned by the model can thus contribute to the semi-automatic generation of the synonyms and concepts layers of a space systems ontology. The vector representations of the domain-specific terms are learned with a word2vec model trained with the main corpus of unstructured data presented in Section 3.2.1. To summarise, the research presented in this section makes the following contributions:

1. The second layer of the Ontology Learning Layer Cake is, for the first time, applied to a collection of texts related to space systems.
2. A word embedding model, word2vec, is for the first time trained on a space

systems corpus to produce domain-specific representation vectors.

3. The potential of using word embedding to identify similar terms and concepts related to space systems engineering is demonstrated.

Section 4.4.1 provides the background on word embedding methods, the word2vec architecture, as well as past similar work. Section 4.4.2 summarises the approach to discover synonyms and concepts from the semi-automatically lexicon generated in Section 4.3. Section 4.4.3 presents the hyper-parameters of the word2vec model and the comparison metrics of cosine similarity. Finally, the representation vectors obtained with the trained word2vec model are compared and discussed in Section 4.4.4.

4.4.1 Background

A detailed background on the origin and evolution of word embedding methods is provided in Appendix 8.2, Section 8.2.2. This section focuses on the word2vec framework.

Continuous Bag-of-Words and Skip-gram architectures

Mikolov et al. present two architectures for their word2vec framework [123], shown in Figure 4.11. The first proposed architecture is the Continuous Bag-of-Words (CBOW) model, similar to the FNNLM presented by Bengio et al. [124], but without the hidden layer. While the FNNLM architecture only takes into account previous terms to predict a next word, the CBOW architecture looks at the words both before and after the target word. The range of words taken into account is called the *context window*. In this architecture, the word order is not considered. The Skip-gram architecture is similar to CBOW's, but instead of predicting a word based on its surrounding terms, the model attempts to predict the surrounding terms based on the target word. As defined in [125], the training objective of the Skip-gram model is to maximise the average log probability defined in Equation 4.5 as:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t) \tag{4.5}$$

where T is the number of embedded words, c the size of the context window and

$p(w_{t+j}|w_t)$ the probability of word w_{t_j} to be within the context window of the target word w_t.

In [123], Mikolov et al. observed than both CBOW and Skip-gram architectures achieved higher accuracy on semantic and syntactic tasks than the feedforward NNLM and Recurrent NNLM methods. The semantic tasks included for instance the identification of capital city relationship such as *Bogota-Colombia* or of currency such as *France-Euro.* The syntactic tasks targeted among others the extraction of superlative relationship such as *fast-fastest* or of nationality adjective such as *Scotland-Scottish.* The Skip-gram architecture was shown to perform slightly worse than the CBOW on syntactic but significantly better on semantic tasks. For the purpose of concept and synonyms identification, the Skip-gram architecture will therefore be favoured here.

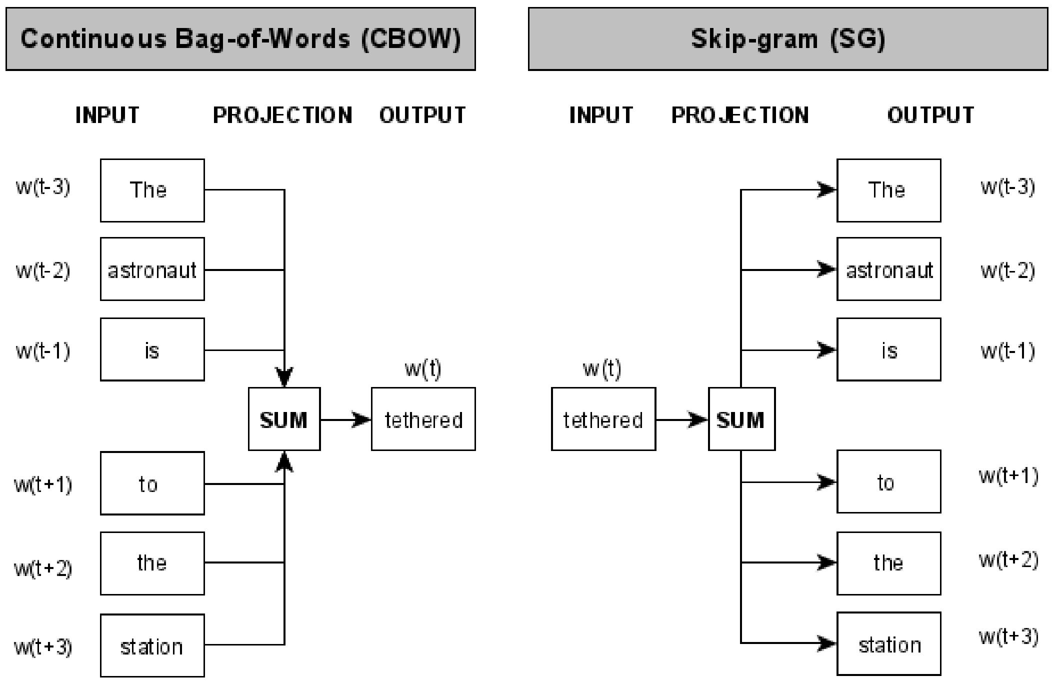


Figure 4.11: CBOW and SG Architecture, derived from [123]

Optimisation of the Skip-gram model

Soon after introducing the CBOW and Skip-gram architectures, Mikolov et al. released a second paper [125], in which they suggested the use of sub-sampling of frequent words and Negative Sampling to reduce the computational cost and improve the embedding quality of the Skip-gram architecture. With sub-sampling, the authors filter frequent words with low informativeness value such as *the* or *a* and accelerate the learning. The sub-sampling depends on the probability of each word w_i defined in Equation 4.6 as:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{4.6}$$

where $f(w_i)$ is the frequency of the word w_i, and t a threshold of around 10^{-5} as suggested by the authors. Mikolov et al. encourage the use of subsampling as it improves the training time and results in significantly better word representations.

The second contribution of their paper is the introduction of Negative Sampling as an alternative to hierarchical softmax. In a neural network, softmax is a common classifier activation for the output layer which provides a probability distribution summing to 1. The softmax layer updates all output vectors at each training instance, and therefore quickly becomes unpractical to scale up and costly for large vocabularies. This downside was already underlined in [124]. An alternative is to limit the number of output vectors updated per training iteration, either with hierarchical softmax or sampling. As summarised in [126], hierarchical softmax is a technique for approximating the softmax. The model uses a binary tree to organise the vocabulary, each word being a unique leaf of the tree with a unique path linking it to the root as shown in Figure 4.12. In their first paper [123], Mikolov et al. use a Huffman tree to represent the vocabulary. A Huffman tree places rare words at deeper levels (long binary codes), and the most frequent words at shallower level (short binary code) of the tree. When the softmax would require x outputs to be evaluated, the hierarchical softmax reduces the complexity to the computation of $\log_2(x)$ and the Huffman tree decreases it even more [123].

In their second paper [125], Mikolov et al. propose a more straightforward solution based on Negative Sampling. Instead of reorganising the vocabulary with hierarchical softmax, Negative Sampling simply reduces the output layer computational cost by updating only a sample of output vectors. Furthermore, negative samples (random

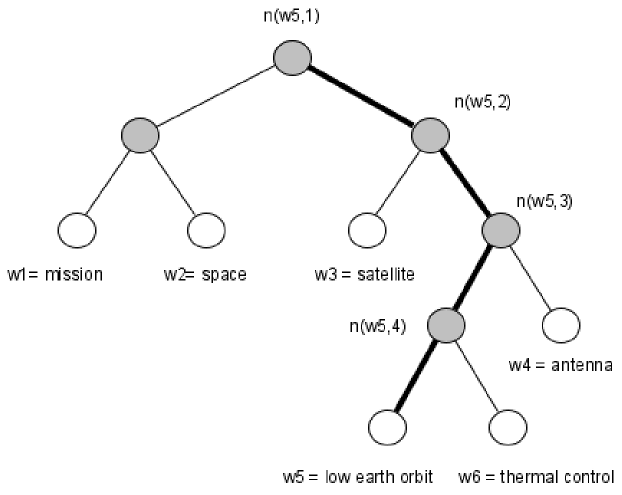


Figure 4.12: An example of Huffman binary tree for the hierarchical softmax, an example of path from the root to the $w5$ word is highlighted. Each leaf is a word and the coloured units are inner units. Derived from [126].

output words) are chosen for a noise distribution. Mikolov et al. recommend a number of negative samples k between 5 and 20 [125]. The training objective E with Negative Sampling is thus defined by Equation 4.7 defined in [126] as:

$$E = -\log \sigma(\mathbf{v}'^{T}_{w_O}\mathbf{h}) - \sum_{w_i \in W_{neg}} \log \sigma(\mathbf{v}'^{T}_{w_i}\mathbf{h}) \tag{4.7}$$

where w_O is the output word and the positive example, $\mathbf{v}'_{w_O}$ is the output vector, $\mathbf{h}$ the output value of the hidden layer, W_{neg} is the set of k negative samples, and $\mathbf{v}'_{w_i}$ the update vectors of the negative sampling. Mikolov et al. encourage the use of Negative Sampling as it contributes to the learning of high-quality word representations and significantly speeds up the computation.

Similar Work

The recent advances in word embedding have benefited the research on Ontology Learning, embedding were successfully applied to the development of domain-specific ontologies in the legal [127], medical [128], biomedical [129], and nutrition [130] fields. The works presented in [127] and [131] are the closest to the approach proposed here.

In [127], a dictionary of legal terms is first computed based on statistical methods and POS tagging. Then, synonym terms are clustered with a word2vec model and a similarity measurement called cosine similarity. In [131], keywords are extracted from the English Wikipedia and then clustered into concepts with a word2vec embedding. The authors of [128] and [130] focus on enriching pre-existing ontologies from either medical or nutrition texts with word representation learned with a word2vec model. In [130], the authors achieve a 89.7% increase in accuracy compared to an expert-curated ontology using word2vec embedding. Finally, the authors of [129] focus on the merging of similar entities in a biomedical ontology with a word2vec model. The authors underlines how previous matching methods based on terminological and structural features lack the semantics understanding that word embedding can achieve.

Although word embedding appears as a common method for building and enriching ontologies, it has not yet been applied to the field of space systems. As a matter of fact no word2vec model trained on a space systems corpus was encountered in the Literature.

4.4.2 Approach

The objective is to leverage word embedding methods to merge synonym entities and identify the concepts in the domain-specific lexicon semi-automatically generated in Section 4.3. A word2vec model is trained with the main corpus of unstructured data presented in Section 3.2.1 including the ESA feasibility reports, books, publications and Wikipedia pages. As words with similar context yield similar representation vectors, the comparison of the representation vectors should lead to the semi-automatic generation of the Synonym and Concepts layers in the Ontology Learning Layer Cake. The approach, displayed in Figure 4.13, thus leverages (i) word embedding to map the domain-specific lexicon terms to unique representation vectors with the word2vec model, and (ii) the common similarity metrics of cosine similarity to compare the vectors.

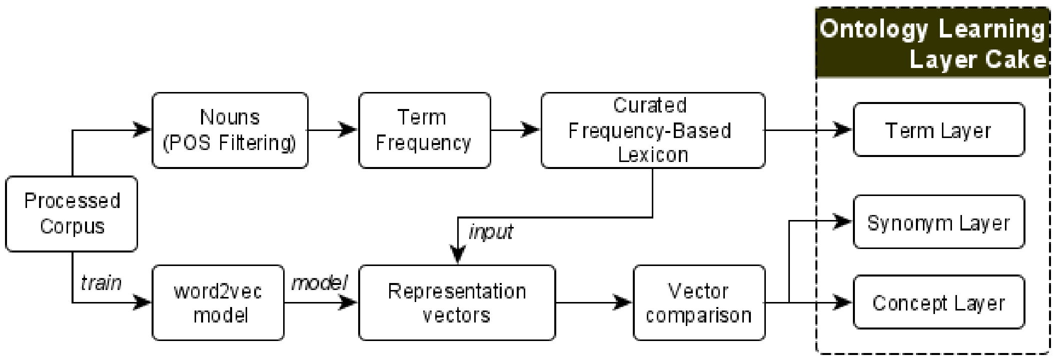


Figure 4.13: Methodology for the semi-automatic generation of Synonym and Concepts layers

4.4.3 Methodology

Word2vec model hyperparameters

The key hyperparameters of the word2vec model are summarised in Table 4.6. Based on the suggestions of [125], the Skip-gram architecture, sub-sampling and Negative Sampling are used instead of the CBOW architecture and hierarchical softmax. A grid search is completed for a window size from 2 to 10 with a step of 1, an embedding dimension from 100 to 400 with a step of 100, and a negative sample range from 5 to 9 with a step of 2. The models are trained with the open-source Gensim Python library [132]. The model with a window of 2 (on each side of the target word), an embedding dimension of 100, and a number of negative samples of 7 performed the best.

Table 4.6: Word2vec Hyperparameter

Hyperparameter	**Setting**	**Parameter Description**
Model Architecture	Skip-gram	Architecture of the algorithm (either CBOW or Skip-gram)
Window Size	2	Maximum distance between the current and target word within a sentence
Embedding Dimension	100	Size of the representation vector
Negative Sampling	7	Number of noise words introduced for negative sampling
Epoch	15	Number of training iterations
Minimum Count	2	Minimum word frequency threshold (all words with a lower frequency are ignored)

Cosine Similarity

The cosine similarity is a common method to compute the similarity between the vectors. The cosine similarity $cos(\theta)$, between two vectors A and B, of dimension n, is defined by [127] as:

$$cos(\theta) = \frac{A.B}{\mid A \mid\mid B \mid} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{4.8}$$

A cosine similarity of 1 denotes identical vectors. The higher the cosine similarity value, the most similar the terms are.

4.4.4 Results & Discussion

To select the cosine similarity threshold, the similarities between the vectors of each lexicon term are computed. The distribution of the 10 highest cosine similarity values obtained for each lexicon term is displayed on Figure 4.14. The average is chosen as threshold. The terms with a cosine similarity above the average are thus considered as potentially synonyms or items of a concept. Out of the original curated lexicon of 3,511 terms, 1,756 are linked to at least another term with a cosine similarity above the threshold of 0.71.

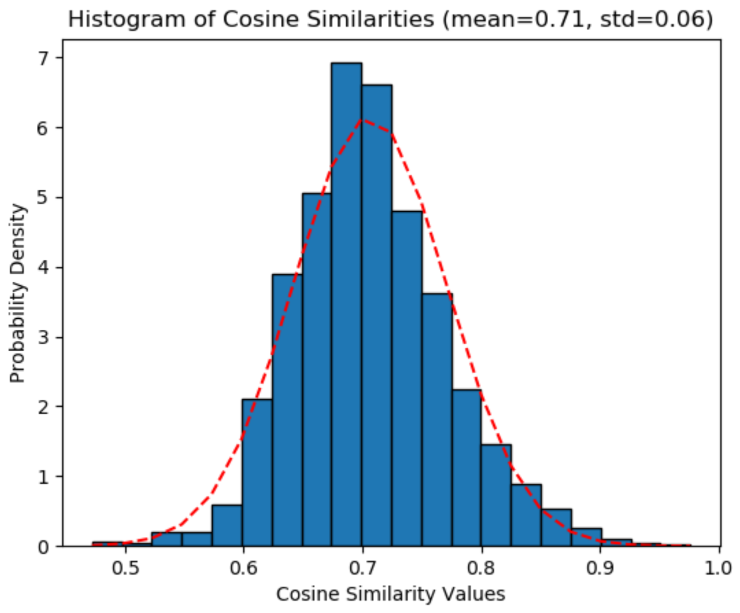


Figure 4.14: Distribution of the 10 highest cosine similarities values obtained for all lexicon terms

Vectors with high dimensions are difficult to interpret. However, with a Principal Component Analysis (PCA) [133], the vectors can be projected in a two-dimensional space. Similar concepts should have similar representation vectors and therefore should appear in the same area of the graph. Figure 4.15 displays the PCA projection of a selected sample of terms. The model appears to be successful in clustering terms into concepts. The cluster in the bottom right corner of the graph includes orbital parameters terms such as *inclination* or *eccentricity*. The cluster with the terms *rocket*, *launch vehicle* and *upper stage* could be related to a *launcher* concept.

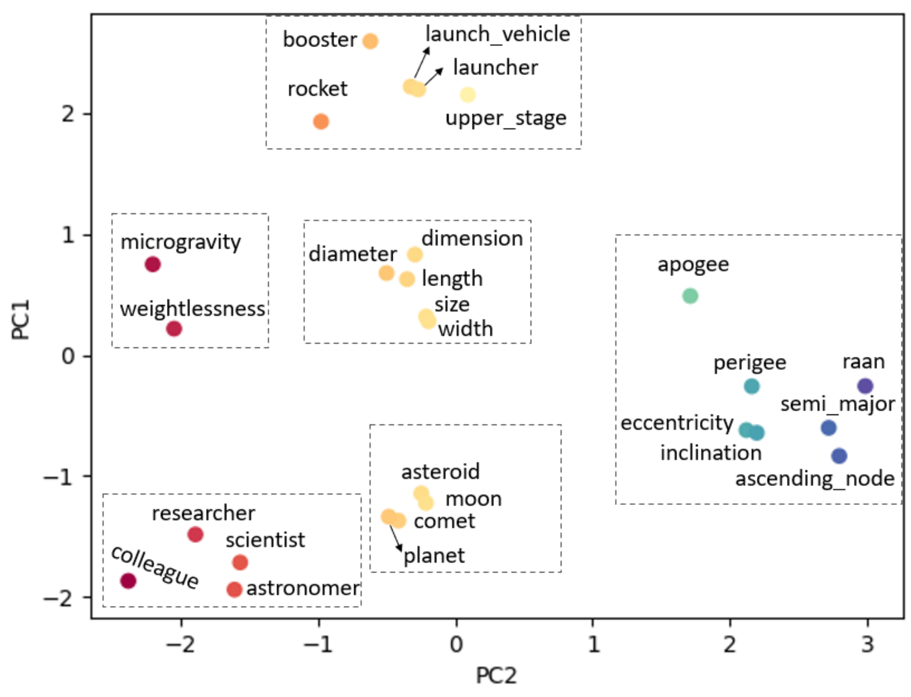


Figure 4.15: PCA projection for sample of concepts clusters obtained with the word2vec model

The manual verification of the similarities reveals a fraction, around 3.7%, of similar terms correspond to exchangeable synonyms. A sample of discovered synonyms is shown in Figure 4.17a. The value of the cosine similarity between these synonyms also varies greatly as shown on Figure 4.16. It is thus not sufficient to use a higher cosine similarity threshold to distinguish inter-exchangeable synonyms from concept clusters. Human validation is necessary. Moreover, as shown in Figure 4.17b, antonyms also have similar representation vectors as they are often found in comparable contexts.

The embedding has additional unexpected benefits. Acronyms and their expansions, words sharing the same lemma, or words with different spellings that were missed by the NLP pipeline also have similar context and could thus be identified. The merging of these redundant terms contributes to enhancing the quality of the lexicon and completing the expansions of domain-specific acronyms. As seen in Table 4.7, the representation vectors of the *dofs* and *degrees of freedom* words have a high cosine similarity. Similarly, words such as *singularity* which should have been filtered by the POS can

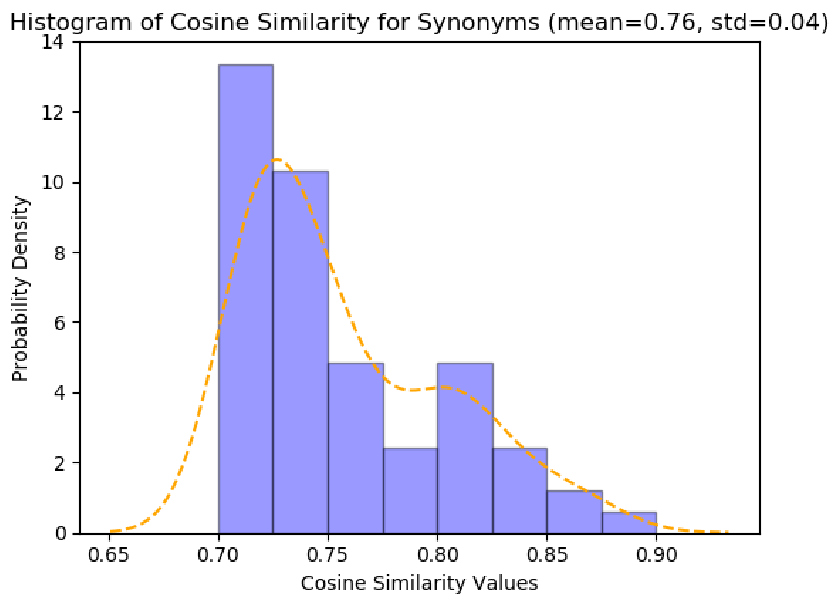
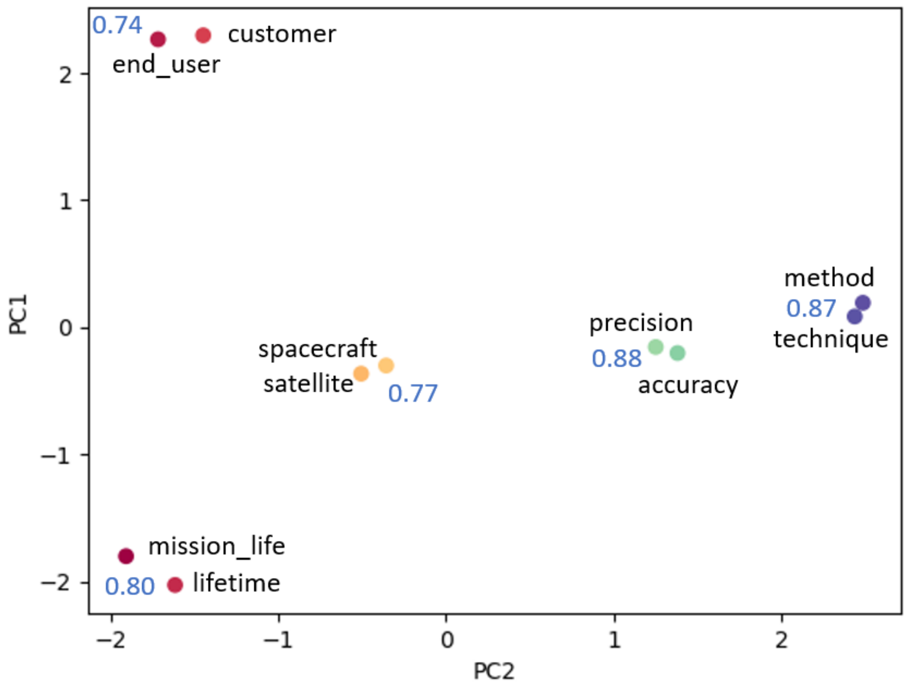


Figure 4.16: Distribution of the cosine similarities values obtained for the synonyms

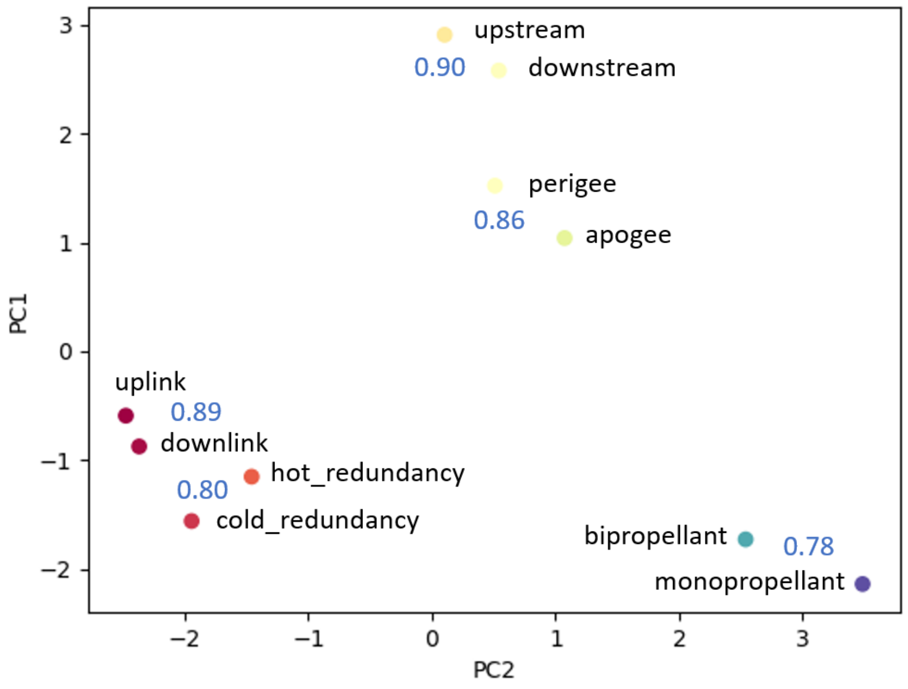
now be linked to the *singular* term. The English spelling of *vapour* is found similar to its American variant *vapor*. Finally, the representation vectors seem to also be able to contribute to identifying the hierarchies between terms, which could be useful to further semantic analysis. Thus, as shown in Table 4.7, the terms *space debris* and *debris* are found to be related.

Table 4.7: Alternative types of similarities discovered by the word2vec model

Application	**Examples** ***(term 1, term 2: cosine similarity value)***
Acronyms	adcs, attitude and orbit control system: 0.77 dofs, degrees of freedom: 0.82
Lemma	model, modeling: 0.78 backscatter, backscattering: 0.73 singularity, singular: 0.73
Spelling	re-entry, reentry: 0.86 vapor (US), vapour (UK): 0.90 bi-propellant, bipropellant: 0.77
Hierachy	cosmic ray, cosmic: 0.78 galactic cosmic ray, cosmic: 0.76 atomic clock, clock: 0.72 space debris, debris: 0.86



(a) Synonyms



(b) Antonyms

Figure 4.17: PCA Projection of selected terms to illustrate the similarities between synonyms and antonyms terms

4.4.5 Conclusions

Theses results confirmed that the word2vec model and cosine similarity could successfully identify and associate similar concepts, as well as contribute to improving the Term layer of the OL Layer Cake. However, domain-experts are needed to assess the type of context similarity as not only concepts and synonyms but also antonyms, acronyms and misspelled terms have similar representation vectors.

4.5 SpaceTransformers: Concept Recognition

This section presents an approach and results published by the IEEE Access journal in the paper "SpaceTransformers: language modeling for space systems" by Berquand. A, Darm, P. and Riccardi, A. [14]

Traditional word embedding techniques such as word2vec and GloVe output a single representation vector per word disregarding the polysemy of terms. For instance, the word *spring* which may stand for a season, a source of water or an elastic body has a single word2vec representation vector. Contextualised embedding addresses the pitfall of these global embedding methods by building word vectors based on the context in which they appear. This type of embedding would yield three vectors for the word *spring*. The BERT language model, standing for Bidirectional Encoder Representations from Transformers, proposed by Delvin et al. in [77] is one of the most popular contextualised embedding methods. BERT is a deep learning language model, a model assigning a probability distribution over sequences of words matching the distribution of a language. BERT is inspired from the Transformer architecture and attention mechanism introduced by Vaswani et al. in [134].

Contextualised embedding has profoundly impacted the NLP landscape. The BERT model advanced the State-Of-The-Art (SOTA) performance on 11 NLP tasks [77]. It is today largely used to power Google queries, response suggestion and word predictions in Gmail. BERT is a large pre-trained language model which can be fine-tuned to various downstream NLP applications. This process is called Transfer Learning, and it consists of two stages: (i) a pre-training phase in which contextualised words embeddings are learned through self-supervised training tasks on a large unlabelled corpus, and (ii) a second phase in which the pre-trained model is fine-tuned for a specific task. The performance of the downstream NLP tasks are thus greatly improved with the knowledge transferred from the pre-trained models. Transfer learning brings a decisive advantage for NLP applications, especially for domains where annotated corpora are scarce such as space systems engineering. The introduction of these large pre-trained model has been dubbed as the ImageNet moment of NLP as they may have a similar impact on the field as ImageNet models had on computer vision.

This section addresses the development of contextualised embeddings for space systems. While pre-trained transformer models such as BERT or RoBERTa (Robustly Optimized BERT Pretraining Approach) [135] are trained on general corpora, domain-specific models such as SciBERT [136] have proven to be more adapted to domain-specific downstream tasks. There are however currently no models providing contextualised embedding for space systems. Pre-training language models from scratch is highly resource intensive, requiring large corpora (160 GB for RoBERTa [135]) and costly computational resources (7 days of training on a Tensor Processing Unit (TPU) for SciBERT [136]). Instead, it is suggested here to further pre-train a general baseline model on a domain-specific corpus. The BERT-Base, RoBERTa-Base and SciBERT-scivocab models are thus selected to build SpaceTransformers, a family of three models for space systems language modeling: SpaceBERT, SpaceRoBERTa and SpaceSciBERT. While models pre-trained on a general corpus learned contextualised words embeddings for a general or scientific English vocabulary, with the additional further pre-training, these models will become specialised in space systems engineering. The models performance are evaluated through a fine-tuning Concept Recognition (CR) task applied to space systems terms annotated by hand by three human annotators. CR is a first essential step for the identification and extraction of domain-specific fundamental concepts, enabling the structuring of accumulated data via the construction of ontologies.

The contributions of this section are summarised as follow:

1. A novel family of models, SpaceTransformers, including three models: SpaceBERT, SpaceRoBERTa and SpaceSciBERT is further pre-trained from BERT, RoBERTa, and SciBERT on a space systems corpus.
2. A novel labelling scheme based on space standards and its corresponding hand-annotated data set for Concept Recognition (CR) of space systems terms is released.
3. A thorough comparison of the performance of domain-specific models with respect to several baseline models on a classification task are provided for the first time.

4. Further pre-training from RoBERTa-Base is demonstrated to considerably improve the results on the downstream domain-specific CR task.

Section 4.5.1 briefly introduces the attention mechanism and the Transformer architecture, subword tokenization, as well as several general and domain-specific pre-trained models. Section 4.5.2 summarises the approach to train the spaceTransformer family. Section 4.5.3 presents the training hyperparameter selection and the labelling scheme applied to the ECSS requirements corpus for the case study. Finally, Section 4.5.4 and Section 4.5.5 respectively detail the training and CR results as well as reflections on the contextualised embedding approach.

4.5.1 Background

A detailed background for this section is provided Appendix 8.2, Section 8.2.3 and is recommended to get acquainted with the attention mechanism and the Transformer's architecture. The appendix also reviews known general pre-trained models, including their training tasks and tokenization approaches.

As a reminder, Figure 4.18 summarises the contextualised embedding process with the BERT model. The input sentence is tokenized with WordPiece [137], a subword tokenizer. RoBERTa relies on another tokenizer as discussed in Appendix 8.2. The subword tokenizer adds two special tokens, *CLS* and *SEP*, respectively used as classification token and to mark the end of a sentence. Each token is mapped to a vocabulary id. The embedding relies on the attention mechanism as presented in Figure 8.6 in Appendix 8.2. The model outputs are the contextualised embeddings, vectors with a size corresponding to the number of hidden units, 768 of the BERT model.

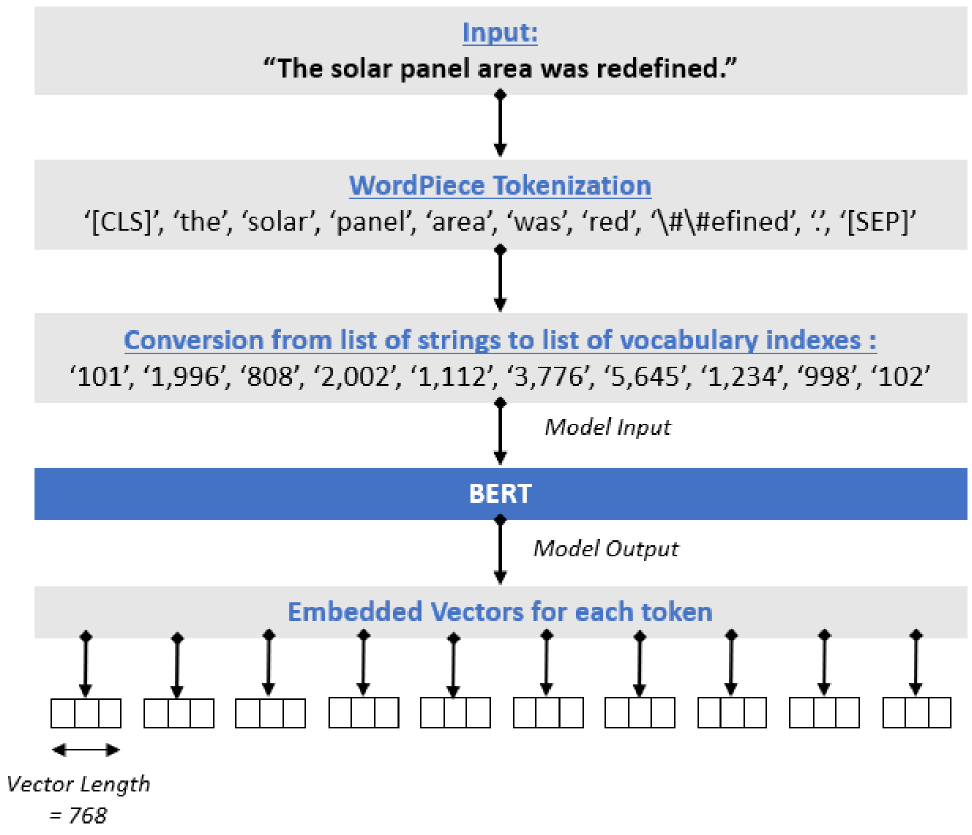

Input:
"The solar panel area was redefined."
WordPiece Tokenization
'[CLS]', 'the', 'solar', 'panel', 'area', 'was', 'red', '\#\#efined', '.', '[SEP]'
Conversion from list of strings to list of vocabulary indexes :
'101', '1,996', '808', '2,002', '1,112', '3,776', '5,645', '1,234', '998', '102'
Model Input
BERT
Model Output
Embedded Vectors for each token
Vector Length = 768

Figure 4.18: Overview of the BERT embedding process

Transfer Learning

The purpose of transfer learning is to first learn from an initial training objective, then apply it to a different target objective. Let s be an input sequence consisting of m words such that:

$$s = (t_1, .., t_m) \tag{4.9}$$

where t_i is the i^{th} word of the sequence. These tokens have a fixed initial embedding of dimension n, noted as x_i. The pre-training phase yields a contextualised embedding y_i of dimension d for each embedding x_i of a term t_i:

$$f : \mathbb{R}^n \times \Theta_f \to \mathbb{R}^d, \quad f(x_i, \theta_f) = y_i \tag{4.10}$$

where $\theta_f \in \Theta_f$ represents a particular set of model parameters. In the pre-training phase, the model f is trained in a self-supervised fashion. In a second phase, the pre-trained model is fine-tuned for a specific task. The contextualised representations previously obtained are used as inputs to the model

$$g : \mathbb{R}^d \times \Theta_g \rightarrow \mathbb{R}^q, \quad g(y_i, \theta_g) = z_i \tag{4.11}$$

The output is a probability distribution through an identity or softmax activation function, configured by the parameters $\theta_g \in \Theta_g$ and of dimension q. The parametrisation of the fine-tuned model is thus configured by

$$\theta_{ft} = [\theta_f, \theta_g] \tag{4.12}$$

This framework has proven to be more efficient than training a task-specific model from scratch, requiring at least 10 times less task-specific data samples [77, 138]. The number of pre-training parameters, θ_f, is usually much higher than the number of fine-tuning parameters θ_g. For instance, the configuration of BERT-Base involves a $\theta_{f,BERT}$ of $110M$ parameters [77]. Thus, the training set required for fine-tuning is significantly smaller than for the pre-training, while avoiding over-fitting.

Domain-Specific Language Models

There are three approaches found in the Literature to generate domain-specific language models: (i) a generic model is fine-tuned on a domain-specific task, (ii) a model is further pre-trained from a generic pre-trained model with a domain-specific corpus, or (iii) a model is trained from scratch on a domain-specific corpus.

Fine-tuning a pre-trained model for a domain-specific task is the quickest and easiest approach. In [139], the authors fine-tuned BERT-Base on a patent database for a classification task. Their model, patentBERT achieved better results than the previous SOTA method based on a Convolutional Neural Network (CNN) and word vector embedding. Krishnan et al., in [140], describe a downstream application similar to the work presented here. The authors fine-tuned BERT-Base on a CR task to identify concepts related to space systems engineering. Their study is so far the only application of Transfer Learning in the space field. Their labelled dataset was however based on a single document, the NASA System Engineering Handbook [141] and they chose high-levels labels such as *event* or *location* whereas the labels used in this study cover all management, product assurance and engineering disciplines found in 126 ECSS space standards.

Pre-training from scratch or further pre-training on a domain-specific corpus enables the introduction of domain-specific words embeddings in the language model, improving the performances on downstream domain-specific tasks. BioBERT [142] and VNLawBERT [143] were both further pre-trained from BERT-Base respectively with biomedical publications and a Vietnamese legal corpus. A clinical language model presented in [144] was further pre-trained from BERT-Base and from BioBERT. Both ClinicalBERT [145] and FinBERT [146] were trained from scratch on an architecture similar to BERT's with, respectively, a corpus of clinical notes and a large financial corpora. The benefits of either further pre-training or training from scratch on a domain-specific corpus have been largely proven by these studies as they all outperformed their baseline general language models on domain-specific tasks.

Further pre-training or training from scratch appears as a trade-off between the domain-specific corpus size, the available computational resources, and the fine-tuning performances sought-after. Training from scratch is resource intensive, it requires a large domain-specific corpus and heavy computational resources. Both BERT and SciBERT use a corpus of around 3B tokens. The training of BERT-Base was performed in 4 days on 4 cloud TPUs [77]. RoBERTa was trained in one day over 1024 V100 GPUs [135]. SciBERT took 7 days to train from scratch with a single TPU v3 with 3 cores [136]. In [147], a legal language model, LEGAL-BERT, is trained on a 12GB corpus of legal text in English, either from scratch or further trained from BERT-Base. The authors found that both were valid approaches with similar results. The training corpus used here has a similar size as [147] and a single NVIDIA V100 GPU with 16 cores is available through the ARCHIE-WeST High Performance Computer[3] to train

[3] www.archie-west.ac.uk

the models. Based on these limitations, the decision was taken to further pre-train the domain-specific models rather than train them from scratch.

Concept Recognition for space systems

CR is a NLP task used to identify, extract, and classify relevant terms from unstructured text. It is a word-level annotation exercise and concepts are loosely defined as sequences that represent a specific cognitive construct in their domain [148]. In the context of systems engineering, these concepts could be "*engineering unit*", "*system architecture*" or "*system analysis*", labelled as examples for the label "*system concepts*" by Krishnan et al. in [140].

There are various CR approaches found in the Literature. Rule-based and pattern matching systems leverage hand-crafted rules on the text and its linguistic features to extract concepts as shown in [149]. Alternatively, other methods are based on supervised ML methods, trained from example inputs and their expected outcomes. Linguistic feature-based ML systems such as support-vector-machines (SVM), decision trees, and conditional random fields used to be the preferred methods for CR [150]. In the last years though, they have been increasingly replaced by Deep Learning approaches using word embedding as input features [151, 152]. Language models and Transfer Learning have notably advanced the field in recent years. Transfer Learning increases the performances of CR applications, as shown by [153] and [154], requiring a smaller labelled dataset than before. The contextualised representation of a word in the transformer architecture contributes to recognising and differentiating concepts based on the context, increasing the accuracy of the model's predictions.

4.5.2 Approach

The approach, summarised in Figure 4.19, applies the transfer learning process. The SpaceBERT, SpaceRoBERTa and SpaceSciBERT models are respectively further pre-trained from BERT-Base, RoBERTa-Base, and SciBERT-scivocab. The models are then fine-tuned on a domain-specific CR task. The further pre-training corpus includes the main text sources presented in Section 3.2.1, with the exception of the ESA CDF reports. The models were trained with the ARCHIE-WeSt High Performance Computer based at the University of Strathclyde. Access to the reports being only granted through a remote ESA server, this subcorpus could not be used for external computations on ARCHIE. Using the abstracts of the publications was also found to yield better results than using the full journal publications documents. The reason is most likely that papers include mathematical notations, figures and tables which introduce noise. The fine-tuning data set includes the ECSS requirements corpus presented in Section 3.2.2. The text is not processed with the NLP pipeline presented in Section 3.2.3 as the BERT and RoBERTa models rely on their own subword tokenizers. The statistics of both corpora are reminded in Table 4.8.

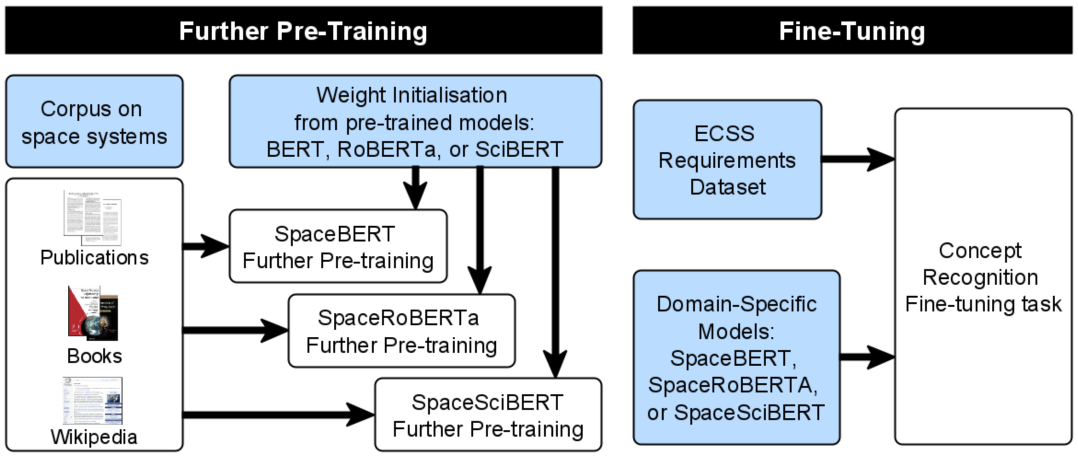

Further Pre-Training
Fine-Tuning
Corpus on space systems
Weight Initialisation from pre-trained models: BERT, RoBERTa, or SciBERT
Publications
Books
Wikipedia
SpaceBERT Further Pre-training
SpaceRoBERTa Further Pre-training
SpaceSciBERT Further Pre-training
ECSS Requirements Dataset
Domain-Specific Models: SpaceBERT, SpaceRoBERTA, or SpaceSciBERT
Concept Recognition Fine-tuning task

Figure 4.19: Overview of the further training and fine-tuning approach with SpaceBERT, SpaceRoBERTa, and SpaceSciBERT.

Table 4.8: Reminder of the Further Pre-training and fine-tuning corpus statistics

	Further Pre-training corpus				**Fine-tuning corpus**
Corpus	**Publications Abstracts**	**Books**	**Wikipedia Webpages**	**All**	**ECSS Requirements**
Number of documents	4,991	39	242	5,327	27,016
Number of sentences	37,957 (10.5%)	274,807 (76.5%)	46,544 (13%)	359,308 (100%)	27,016

4.5.3 Methodology

Further Pre-training

Further pre-training a model

$$f : \mathbb{R}^n \times \Theta_f \rightarrow \mathbb{R}^d, \tag{4.13}$$

means that in the pre-training phase, instead of randomly initialising the weights θ_f, the weights values of a baseline model such as BERT, RoBERTa or SciBERT are reused. Hence the weights θ_f for the three further pre-training tasks are initialised with the following set of weights

$$\theta_{f,0} = \theta_{f,BERT}, \tag{4.14}$$

$$\theta_{f,0} = \theta_{f,RoBERTa}, \tag{4.15}$$

$$\theta_{f,0} = \theta_{f,SciBERT} \tag{4.16}$$

where $\theta_{f,BERT}$, $\theta_{f,RoBERTa}$ and $\theta_{f,SciBERT}$ are respectively the set of weights of the pre-trained models BERT, RoBERTa and SciBERT.

Weights initialisation from a pre-trained model also implies the reuse of the original model vocabulary. The authors of the SciBERT model [136] observed an average improvement of only +0.76 F1 score on biomedical tasks when using their domain-specific vocabulary. They concluded that training with a domain-specific corpus had more impact than using a domain-specific vocabulary. A study similar to the one presented here, BioBERT [142], chose to rely on the bert-base vocabulary. The authors assessed that since the WordPiece tokenization used to build the BERT vocabulary reduces OOV issues it was fit to represent and fine-tune their domain-specific terms. An alternative to training from scratch with a domain-specific corpus is to replace "Unused" tokens in the vocabulary with domain-specific words.

To assess if a modification of the original vocabulary was necessary, the top thousand most frequent words from the domain-specific corpora are extracted and compared to the vocabulary of bert-base-uncased, roberta-base, and scivocab-uncased. The top 10 most frequent words in the domain-specific frequency-based lexicon are: "*satel-*

lite", "*system*", "*orbit*", "*space*", "*spacecraft*", "*data*", "*time*", "*mission*", "*model*", and "*control*". Out of this frequency-based lexicon, 87, 8% of the words were already included in the bert-base-uncased vocabulary, 88, 8% in the roberta-base vocabulary, 89, 9% in the scivocab. Within these 1000 words, the 10% most frequent words were included in all three vocabularies already. Table 4.9 gives a sample of the words not found in the generic models vocabularies. As the amount of domain-specific vocabulary not covered by the original vocabularies was not significant (less than 15%), it was decided to re-use the vocabularies and tokenizers of the baseline models.

Table 4.9: Sample of terms not found in the generic models, words in bold are missing from more than one vocabulary.

Vocabulary	**bert-base-uncased**	**roberta-base**	**scivocab**
Number of terms not found in baseline vocab.	123	112	101
Top 5	subsystem, **thruster**, **propellant**, **perturbation**, **telemetry**	**thruster**, **propellant**, **perturbation**, **telemetry**, **actuator**	rocket, **thruster**, **propellant**, lunar, **telemetry**

The configuration and pre-trained weights of the BERT-Base, RoBERTa-Base and SciBERT models are accessed through the HuggingFace library and their Python Transformers library [155]. For each model the pre-training weights and hyperparameters are thus initialised from one of the three baseline models with the exception of the batch size and maximum sequence length. The batch size is set to 256, as for RoBERTa [135], with a gradient accumulation step of 16. The maximum sequence length of the input is set to 512 as defined in BERT [77]. The models are further pre-trained for 70 epochs on one NVIDIA V100 GPU hosted on ARCHIE-WeST High Performance computer. The further pre-training corpus is split between a training and a testing set, based on the classic 80%/20% partition.

Requirements labelling

For the fine-tuning of the pre-trained models, the corpus presented in Section 3.2.2 is used as a basis for the annotated dataset. The requirements are written in a precise and brief manner, with a high density of concepts relevant to space systems, making them useful for generating a CR dataset in this domain. An annotation scheme is carefully designed to cover the whole spectrum of the ECSS standards and their requirements, creating labels for each of the three main branches: *Management*, *Product assurance* and *Engineering*. The labels are constructed from domain-experience of three human annotators as well as with the help of online available taxonomies in the space domain such as the ESA Technology tree [156], the ESA Product tree [157] and the NASA taxonomy viewer [4]. 18 labels were eventually defined for the annotation scheme. The complete description for each label is found at `github.com/strath-ace/smart-nlp`. Table 4.10 summarises the annotation scheme, providing a short description and examples for each label.

The single requirements were annotated with the commercial software tool Prodigy from the software company explosion.ai [5]. To facilitate the annotation process, requirements addressing similar topics were annotated simultaneously. The process was repeated for all topics, also ensuring that similar numbers of requirements were selected so that the resulting dataset would be balanced. Subsequently, requirements from the whole corpus were picked to cover the complete scope of the ECSS in a balanced way. The annotation process was considered done once the performance of the CR classifier were within an acceptable accuracy. Eventually 882 requirements were annotated. Each annotator labelled the whole fine-tuning corpus independently. These results were then compared, showing a high level of inter-annotator agreement of 96.5%. Discrepancies between the three different datasets of the respective annotators were discussed and eliminated from the final set. The resulting numbers of annotated concepts present in the final dataset are shown in Table 4.11. The number of unique concepts found per label, as well as the ratio of unique concepts to the total number of concepts, called non-overlapping, are also displayed.

[4]https://techport.nasa.gov/view/taxonomy
[5]`https://prodi.gy/`

Table 4.10: Annotation scheme summary

Label	**Short description**	***Examples***
Management		
Project documentation	Project deliverables	*Certificate of conformity, PCB definition dossier, final review team report*
Project scope	Functions, characteristics, and goals of the mission	*mission phases, project requirements, product functionality*
System engineering	Design, components and functions of a system solution	*dynamic architecture design, system level considerations*
Product assurance		
Nonconformance	Non-fulfilment of a requirement	*explosion, material degradation, cuts, abrasions*
Quality control	Compliance with requirements and specifications	*acceptance tests, evaluation report, unit level testing*
Safety & risk control	Dependability, availability, maintainability, and safety	*safety-approved procedure, worst-cases, emergency controls*
Engineering		
Cleanliness	Contamination and sterilisation	*system cleanliness, contamination control plan, particle counter*
Communication	Communication and navigation infrastructure for telemetry/telecommand (TM/TC)	*telecommand packet, Link budget, message subtype 28*
Guidance Navigation & Control (GN&C)	Design and implementation of control subsystem, analysis and definition of trajectory	*star sensor, natural perigee rise, apogee fall*
Materials & EEE	Electrical, electronic and electro-mechanical (EEE) Components, materials	*Printed Circuit Board, sandwich items, fastener*
Measurement	Physical units	*30 J, 60 mW*
On-Board Data Handling (OBDH)	Data management, data acquisition, data storage, on-board networking and network management	*data-sending lane, DATA OUT signal, Distribution Transfer Descriptor*
Parameter	Generic characteristic	*supplier performance, track width, manufacturing tolerances*
Power	Power subsystem architecture, energy storage, power generation; distribution; and conditioning	*nuclear-energy sources, RTGs, output short circuit, power-energy resources*
Propulsion	Generation of forces and torques to change velocity and orientation of S/C	*thruster generated plasma, sloshing analysis*
Space environment	Effects and environmental conditions governing the space environment	*displacement damage, secondary protons, electron-bremsstrahlung*
Structure & mechanisms	Structural and mechanical subsystem, mechanism subsystem devices	*satellite mechanical structure, static unit load, attachment devices*
Thermal	Thermal management	*thermo-optical properties measurement, heat pipe, two- phases heat transport equipment*

Table 4.11: Summary of annotated concepts per label

Labels	**Number of concepts**	**Number of uniques**	**Ratio non-overlapping**
Quality control	529	366	0.69
Space environment	518	392	0.76
Parameter	433	327	0.76
OBDH	410	317	0.773
System engineering	331	192	0.58
Measurement	301	260	0.86
Power	287	234	0.815
Materials & EEEs	275	203	0.74
Structure & mechanisms	253	223	0.88
Safety & Risk control	225	188	0.84
GN&C	206	164	0.8
Project scope	203	153	0.75
Communication	191	152	0.8
Thermal	185	139	0.75
Project documentation	154	122	0.79
Propulsion	151	118	0.78
Nonconformance	118	88	0.75
Cleanliness	91	64	0.70
Sum / mean*	**5447**	**4112**	**0.78***

Fine-tuning for Concept recognition

The Python Transformers library from HuggingFace [155] was again used to load the pre-trained and further pre-trained models. For CR, a linear layer is added as output layer with a softmax activation function. The models were trained three times with a 10-fold, 80%/20% split, cross validation. Another assumption for the training was to reinitialise the weights of the final layer if the fine-tuning resulted in a failed run for the fold. This is in accordance with previous studies, which stated that the random initialisation of the fine-tuning layers can have a significance influence on the fine-tuning results in computer vision [158] as well as NLP [159]. A failed run was defined as when the validation accuracy stayed below classifying all examples with the majority class, classifying every word as a non-concept [160] .

Further hyperparameters for the fine-tuning were a linear decreasing learning rate and a batch size of 16. The models were trained for up to 10 epochs. To compare the models' predictions, the results of the epoch with the lowest validation loss for each respective fold were taken. One benefit of the further pre-training was already observed during fine-tuning. In comparison to RoBERTa with three failed runs overall, SpaceRoBERTa did not fail any.

4.5.4 Results

Models Selection

Several trials were run for both uncased and cased vocabularies and various batch sizes. Further pre-training on uncased vocabulary yielded better results than cased vocabulary. This was to be expected as the labelled concepts are not named entities and thus casing is not relevant to the application. A higher batch size of 256 also yielded better results than lower batch sizes of 16 or 32.

The models are further pre-trained for 70 epochs which is enough to achieve the convergence of the evaluation perplexity as shown in Figure 4.20. Perplexity is a common metrics for evaluating language models. It quantifies how well a model reduces the uncertainty in the prediction of the language in a tokenized sequence of text X. Perplexity PPL is derived from the cross-entropy H and is defined in [161] as:

$$PPL = 2^{H_p(X)} \tag{4.17}$$

with

$$H_p(X) = \frac{1}{|X|} \log_2 \frac{1}{P(X)} \tag{4.18}$$

where $|X|$ is the number of words in text X, $P(X)$ is the probability of the sequence of words provided by the model, $H_p(X)$ the cross-entropy of the text in relation to the model, and finally PPL the perplexity of the model.

These results were obtained using the ARCHIE-WeSt High Performance Computer based at the University of Strathclyde. The SpaceBERT model trained for 60 epochs, the SpaceRoBERTa model trained for 57 epochs, and the SpaceSciBERT trained for 54 epochs were eventually selected. These models either correspond to the start of the perplexity convergence or to a local minimum close to convergence. Although of disparate initial configuration and pre-training corpus, these models interestingly take

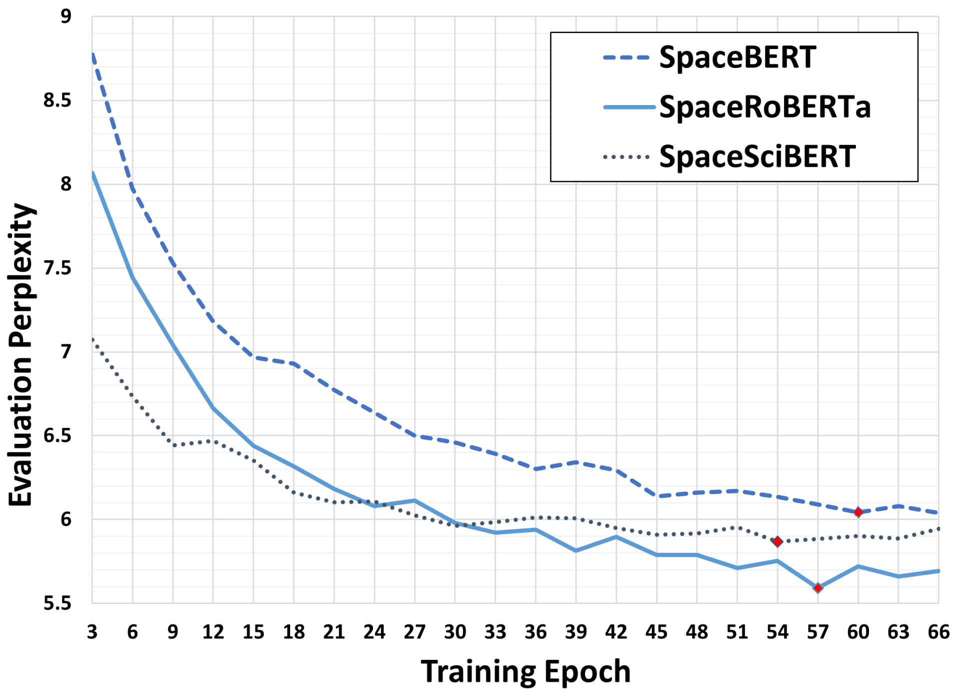


Figure 4.20: Evolution of the evaluation perplexity in function of the number of further pre-training epochs.

a similar number of further pre-training epochs to converge.

Concept Recognition Results

Figure 4.21 displays the evolution of the validation loss for all models with respect to the number of fine-tuning epochs. The validation loss curves have a parabola-like shape reaching a minimum after a certain number of epochs. When comparing the minimums of each model, the validation loss appears to be the lowest for SpaceRoBERTa and the highest for BERT. While SpaceSciBERT and SciBERT have similar validation losses, SpaceRoBERTa, and SpaceBERT demonstrate significant improvements with respect to their respective baseline models. Although the results were averaged over 30 folds, the standard deviation for the validation loss is still high. Former studies [159, 160] reported similar issues for comparable dataset sizes.

The CR F1 scores for all 6 models and 18 labels are reported in Table 4.12. The

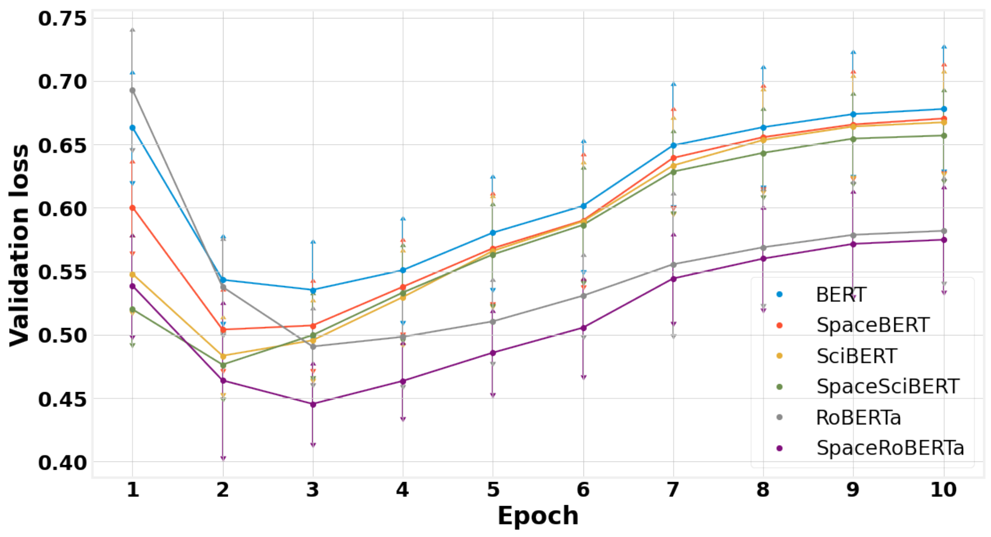


Figure 4.21: Evolution of the validation loss in function of the number of fine-tuning epochs

results were computed from the epochs with the lowest average validation loss, averaged over all 30 folds. The standard deviation is provided along with the F1 score. The *weighted* label represents the averaged F1 score over all the labels weighted by the number of examples in the validation set, and is defined as:

$$weighted = \frac{1}{\sum_{l \in L} n_{\hat{y}_l}} \sum_{l \in L} n_{\hat{y}_l} F_1(y_l, \hat{y}_l) \tag{4.19}$$

where l is one label from the set L of all labels, $\hat{y}_l$ is the set of true samples for label l, y_l is the set of predicted samples for label l, $F_1(y_l, \hat{y}_l)$ is the F1 score calculated for label l, and $n_{\hat{y}_l}$ is the number of true samples for label l.

Considering only this weighted F1 score, SpaceRoBERTa clearly outperforms the other models, followed by SpaceSciBERT. BERT and RoBERTa obtain the lowest scores.

Furthermore SpaceRoBERTa ranks the highest on several labels. As shown on Table 4.12, the labels, displaying the most significant improvements compared to the baseline of BERT, are *GN&C* with a 7.8% improvement, then *Space environment* with 4.5%, followed by *Thermal* with around 4% improvement, and *Structure & mecha-*

nism 3.8%. SpaceSciBERT substantially improves the score of the *Communication* and *OBDH* labels, respectively by 12% and 4%, compared to BERT.

Altogether, the reported F1 scores are consistent with the observed validation loss trends, with SpaceRoBERTa leading the F1 score table and the further pre-trained models outperforming their baselines. The standard deviations of the single scores are still generally high, usually exceeding the achieved improvement between the baseline and the further pre-trained models. Therefore, statistical tests are conducted and summarised in Section 4.5.4 to evaluate the statistical significance of the results.

To fully assess the impact of the further pre-training with a domain-specific corpus, the scores of the baseline models are compared to their respective space variant in Figure 4.22. SpaceRoBERTa again displays the most significant improvements compared to its baseline model RoBERTa. All three domain-specific models show substantial improvements for the *Propulsion*, *Space environment*, *Structure & mechanisms*, *Communication*, *GN&C*, and *OBDH* labels. These labels corresponds to the main engineering disciplines of a spacecraft subsystems. However the score of more general labels such as *Safety & risk control*, *Nonconformance*, and *Quality control* were either unaffected or slightly deteriorated by the further pre-training. These labels all belong to the ECSS branch of *Product assurance.* For the remaining labels, no clear trend can be inferred as the further pre-training resulted either in an improvement or a deterioration of performances depending on the model used.

A more thorough investigation is conducted for the SpaceRoBERTa model as it achieved the highest performance. Figure 4.23 displays the confusion matrix for the SpaceRoBERTa model. The majority of samples are concentrated on the diagonal, thus predictions are predominantly accurate. A few incorrect classifications occur between the *OBDH* and *Communication* labels, indicating a lack of clear boundaries between the two topics.

Table 4.12: Results for the F1 scores of 30-fold cross-validation for each model and each label. The best score for each label is highlighted with a grey background. The standard deviation is presented for each label below the respective F1 score.

Label	BERT	SpaceBERT	SciBERT	SpaceSciBERT	RoBERTa	SpaceRoBERTa
Cleanliness	$0.709_{0.101}$	$0.699_{0.108}$	$0.71_{0.109}$	$0.708_{0.127}$	$0.703_{0.138}$	**$0.722_{0.105}$**
GN&C	$0.703_{0.082}$	$0.723_{0.064}$	$0.715_{0.068}$	$0.739_{0.055}$	$0.708_{0.075}$	**$0.758_{0.071}$**
Materials & EEEs	$0.664_{0.083}$	$0.659_{0.073}$	$0.655_{0.086}$	$0.654_{0.062}$	$0.647_{0.071}$	**$0.665_{0.069}$**
Measurement	$0.888_{0.024}$	$0.879_{0.024}$	$0.883_{0.033}$	$0.875_{0.034}$	$0.889_{0.03}$	**$0.89_{0.026}$**
Nonconformance	**$0.558_{0.079}$**	$0.542_{0.105}$	$0.517_{0.097}$	$0.517_{0.072}$	$0.543_{0.126}$	$0.537_{0.098}$
OBDH	$0.736_{0.054}$	$0.755_{0.063}$	$0.751_{0.051}$	**$0.767_{0.057}$**	$0.706_{0.062}$	$0.761_{0.054}$
Parameter	$0.521_{0.046}$	$0.504_{0.056}$	$0.528_{0.054}$	$0.516_{0.048}$	$0.505_{0.04}$	**$0.539_{0.055}$**
Power	$0.816_{0.042}$	$0.805_{0.058}$	$0.824_{0.041}$	**$0.829_{0.04}$**	$0.786_{0.076}$	$0.819_{0.044}$
Project documentation	$0.508_{0.058}$	$0.487_{0.072}$	$0.489_{0.081}$	$0.491_{0.076}$	$0.479_{0.1}$	**$0.511_{0.073}$**
Project scope	**$0.624_{0.052}$**	$0.623_{0.063}$	$0.621_{0.063}$	$0.616_{0.062}$	$0.607_{0.069}$	$0.617_{0.06}$
Propulsion	$0.699_{0.065}$	$0.712_{0.057}$	$0.684_{0.063}$	$0.707_{0.057}$	$0.637_{0.066}$	**$0.722_{0.053}$**
Quality control	**$0.734_{0.049}$**	$0.718_{0.046}$	**$0.734_{0.045}$**	$0.722_{0.048}$	$0.731_{0.053}$	$0.723_{0.049}$
Safety & risk control	$0.689_{0.047}$	$0.678_{0.052}$	$0.688_{0.06}$	$0.676_{0.051}$	**$0.701_{0.063}$**	$0.692_{0.067}$
Space environment	$0.74_{0.068}$	$0.75_{0.053}$	$0.757_{0.055}$	$0.772_{0.049}$	$0.725_{0.11}$	**$0.773_{0.056}$**
Structure & mechanisms	$0.542_{0.084}$	$0.56_{0.075}$	$0.547_{0.092}$	$0.556_{0.084}$	$0.499_{0.113}$	**$0.563_{0.087}$**
System engineering	$0.617_{0.061}$	$0.631_{0.06}$	$0.592_{0.064}$	$0.629_{0.062}$	$0.61_{0.075}$	**$0.63_{0.064}$**
Communication	$0.644_{0.084}$	$0.677_{0.068}$	$0.672_{0.094}$	**$0.721_{0.059}$**	$0.616_{0.134}$	$0.682_{0.108}$
Thermal	$0.742_{0.045}$	$0.76_{0.063}$	$0.758_{0.046}$	$0.756_{0.054}$	$0.712_{0.108}$	**$0.772_{0.055}$**
weighted	$0.699_{0.019}$	$0.701_{0.024}$	$0.703_{0.02}$	$0.709_{0.019}$	$0.662_{0.114}$	**$0.715_{0.029}$**
Control method	Bonferroni-Dunn test					
SpaceRoBERTa	1.639●	1.833●	1.778●	1.694●	3.055●	-

Bonferroni-Dunn test $CD_{\alpha=0.05} = 1.606$

● Statistically difference with $\alpha = 0.05$

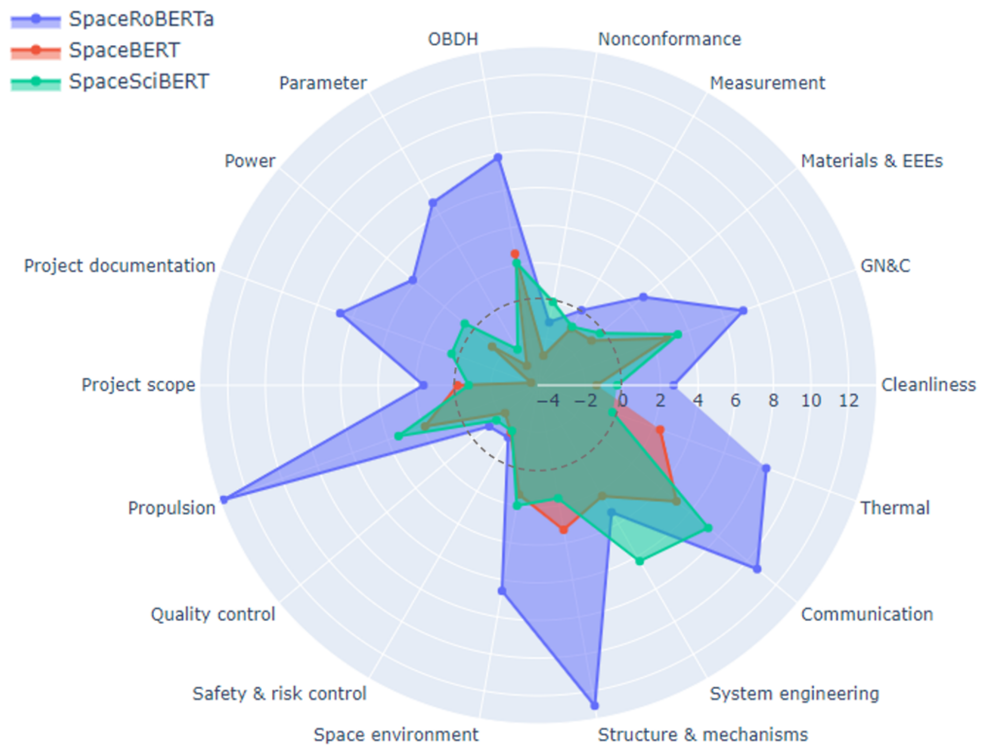


Figure 4.22: Variations (in %) between the performance of the baseline models and respective further pre-trained space models.

The annotated requirement shown in Figure 4.24 illustrates this overlap. SpaceRoBERTa wrongly associates the concepts found in this requirement to the *Communication* label instead of the *OBDH* label as they were manually assigned to. These concepts, including *communication frame* and *command word*, actually fall under the domain of signal processing and can be used both in a communication or data handling context. The requirement was here extracted from a standard related to data handling. This information is however hidden from the model and therefore cannot be used to guide the model. The ambiguity of these terms were already highlighted by the human annotators.

Figure 4.25 quantifies the number of new concepts not seen by the model during training but found in the validation set, demonstrating the ability of the model to generalise over the training set and discover new concepts in unevaluated samples. The

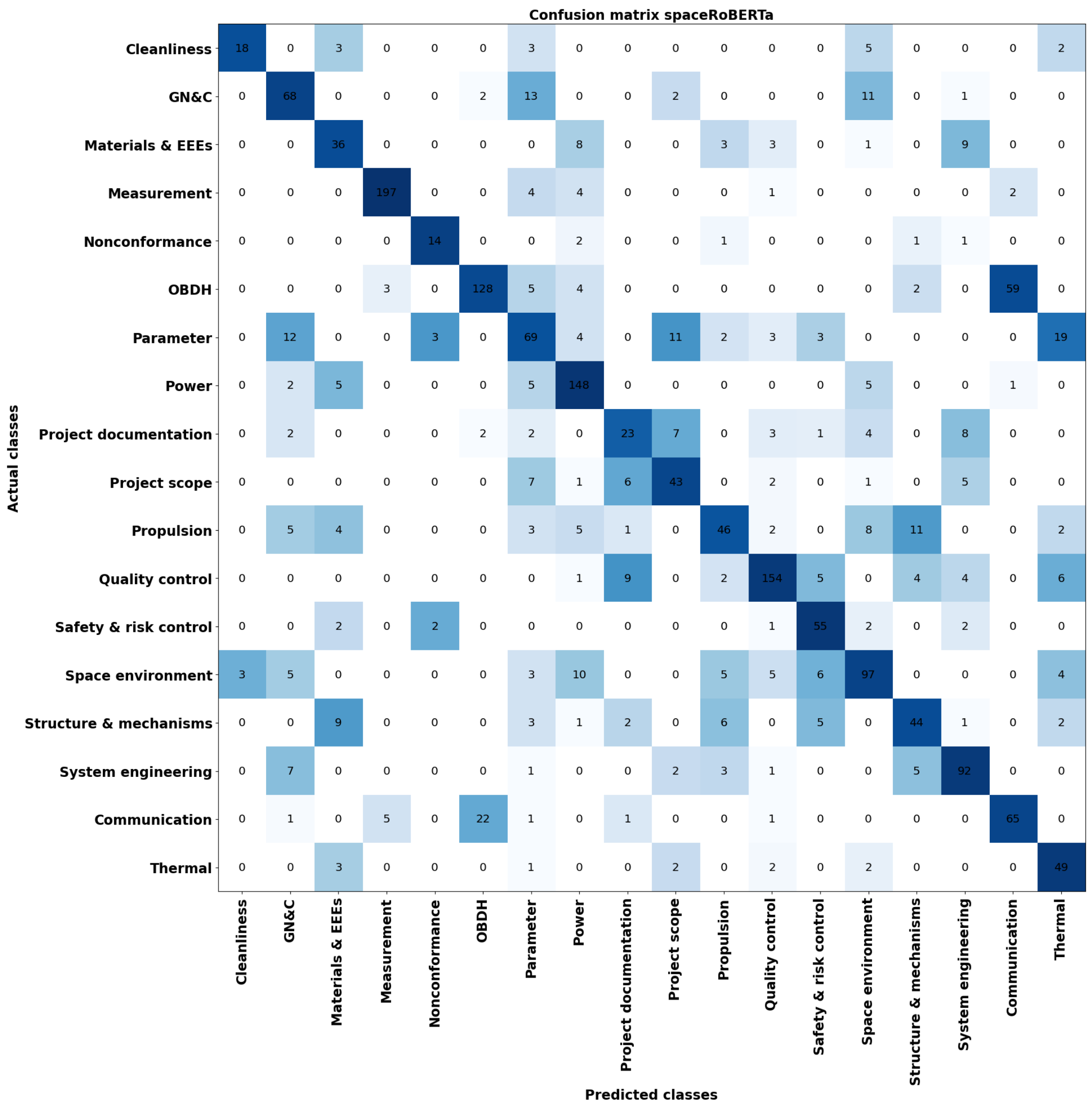
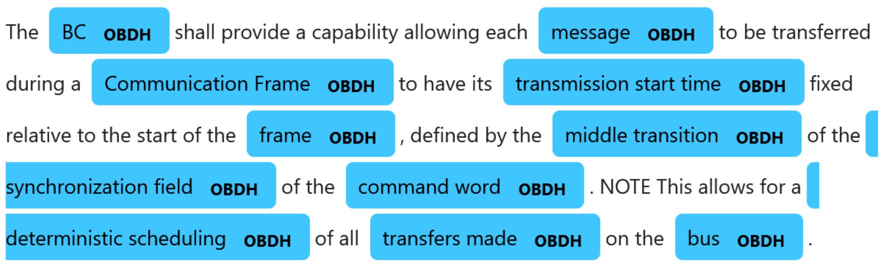


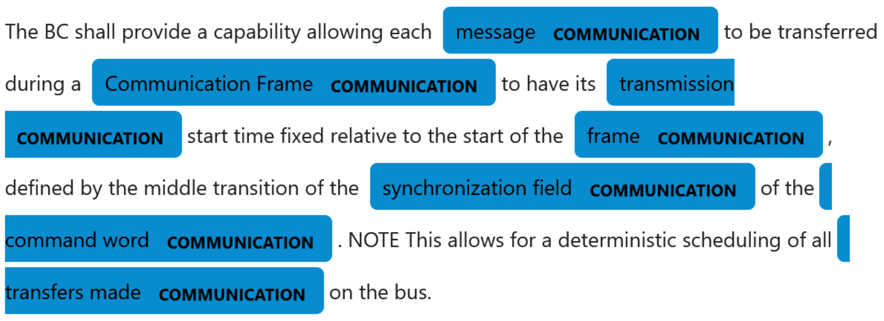
Figure 4.23: Confusion matrix of the fine-tuned SpaceRoBERTa model (the majority class "non-concept" is excluded).

prediction of the model was compared for one fold to a simple look-up approach. The latter method identifies concepts present in both training and validation sets. As seen in Figure 4.25, the prediction with the fine-tuned model achieves substantially better results than the look-up approach. Out of 844 unique concepts, 690 were recognised exactly by the model and 78 concepts were partly recognised. For partial recognition, the span was either too long or too short. For instance, the concept *50W resistors*, corresponding to two labelled concepts *50W* and *resistor* were merged by the model.



The BC OBDH shall provide a capability allowing each message OBDH to be transferred during a Communication Frame OBDH to have its transmission start time OBDH fixed relative to the start of the frame OBDH , defined by the middle transition OBDH of the synchronization field OBDH of the command word OBDH . NOTE This allows for a deterministic scheduling OBDH of all transfers made OBDH on the bus OBDH .

(a) Hand-annotated requirement



The BC shall provide a capability allowing each message COMMUNICATION to be transferred during a Communication Frame COMMUNICATION to have its transmission COMMUNICATION start time fixed relative to the start of the frame COMMUNICATION , defined by the middle transition of the synchronization field COMMUNICATION of the command word COMMUNICATION . NOTE This allows for a deterministic scheduling of all transfers made COMMUNICATION on the bus.

(b) Annotations obtained with SpaceRoBERTa

Figure 4.24: Comparison of manual annotation and model prediction

The concept *flight production* was extracted by the model while the full labelled concept was *proto-flight production*. Alternatively, the look-up approach resulted in only 170 complete matches, and 187 part matches.

Statistical Tests

The results obtained have been statistically analysed with the Friedman pre-hoc and the Bonferroni-Dunn and Nemenyi post-hoc tests. To determine the statistical significance of the F1 score of each method with respect to the labels set, a non-parametric Friedman test was completed with the ranking of the F1 score of the best model as the test variable. The Friedman test shows that the proposed method is statistically significant at a level of 5% as the confidence interval is $C_0 = (0, F_5 = 2.322)$ and the F-distribution statistical values is $F^* = 6.330 \notin C_0$. Consequently Friedman test rejects the null-hypothesis that all models perform equally well in mean ranking. Based on this rejection the Nemenyi post-hoc is completed to compare the performances of the different models. The difference in ranking, as resulting from the Nememyi tests can be observed in Figure

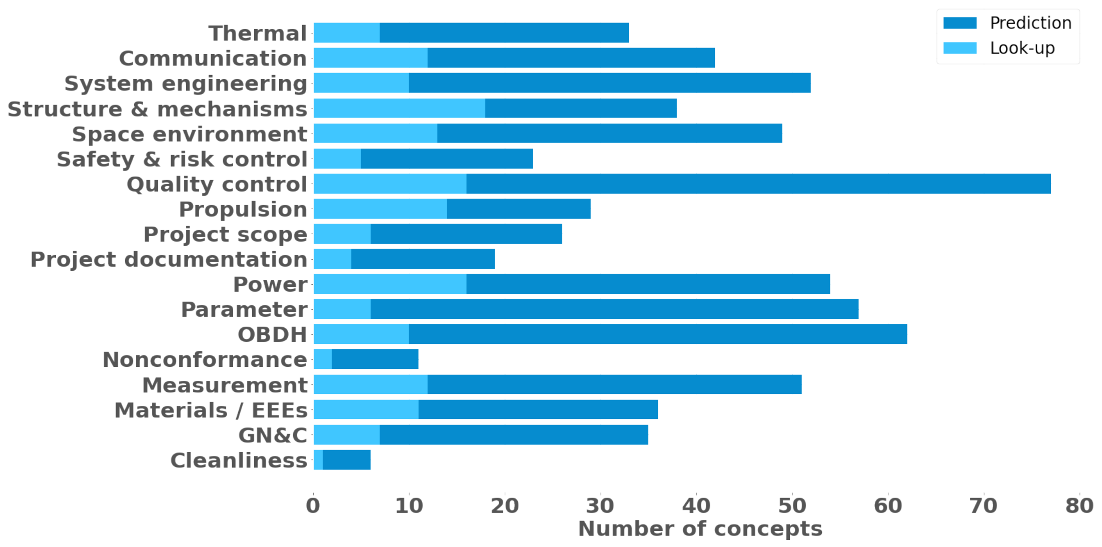


Figure 4.25: Number of unique concepts detected by the SpaceRoBERTa model, compared to a simple look-up approach

5.7, for $\alpha = 0.05$. The results of the Bonferroni-Dunn test for $\alpha = 0.05$ are reported in Table 4.12. From the results of both tests it can be concluded that SpaceRoBERTa has a significant higher ranking than all the other methods and RoBERTa, its baseline, the lowest one. The remaining methods, BERT, SciBERT and their space counterpart instead, have not a significant difference in mean ranking.

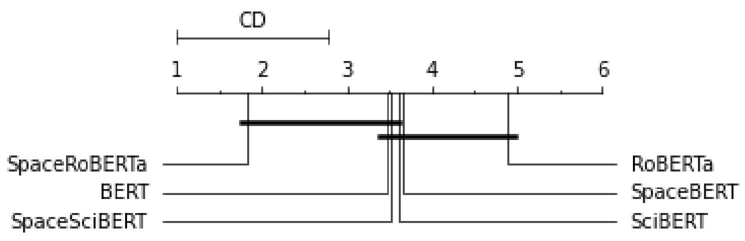


Figure 4.26: Nemenyi CD diagram comparing the generalisation F1 score rankings of the different methods ($\alpha = 0.05$).

4.5.5 Discussions & Future Work

The weighted F1 scores demonstrate that the domain-specific models outperformed their respective baseline models. SpaceRoBERTa benefited the most from the further pre-training with an increase of 8% F1 score with respect to RoBERTa. SpaceBERT and SpaceSciBERT have less significant improvements, respectively displaying increases of 0.3% and 0.85%. Both SpaceSciBERT and SciBERT outperformed SpaceBERT and BERT proving that the scientific pre-training gave an additional advantage to training from a general model. The decisive advantage came from combining the domain-specific training corpus with the alternative pre-training architecture and tokenizer of RoBERTa. Indeed, the latter model is pre-trained on a single Masked Language Model task [135] whereas the BERT-based models are also trained on a Next Sentence Prediction task [77, 136]. The statistical analysis and Bonferroni-Dunn test, ignoring the number of labels in the evaluation set unlike the weighted F1 score, demonstrated that there is no significant difference between SpaceBERT, SpaceSciBERT and their baseline counterpart. The Bonferroni-Dunn test however confirmed the significant higher ranking of SpaceRoBERTA.

Labels covering less domain-specific concepts such as *Nonconformance*, *Project Scope*, and *Quality Control* benefited less from the further pre-training. Domain-specific labels such as *Propulsion*, *Structure & Mechanisms*, and *Communication* however saw their F1 score significantly increased for all space models. These results were obtained for one fine-tuning task. When fine-tuning for another task it is recommended to not discard SpaceSciBERT nor SpaceBERT as different models might be more adapted to different applications.

In future work, other pre-training tasks, beyond MLM and NSP, could be explored as in [162] where a domain-specific model was trained on four different tasks. This is a resource intensive approach requiring additional computational power and a larger training set. The study could also be extended to other BERT-based model, notably DistilBERT [163], a lighter version of BERT which still achieves similar performances but is 60% faster to train. DistilBERT illustrates a counter-trend in the field of Transformers which disapproves of the exponential growth of parameters for economical and environmental reasons. Strubell et al. in [164] estimated the approximate financial and environmental cost of these large pre-trained models. According to Strubell et al., training BERT on GPU is roughly equivalent to a trans-American flight with an estimated 700 kg of CO_2 emission. Futhermore, Strubell et al. estimate the compute cloud cost of GPT-2, trained for 168 hours (1 week) on 32 TPUv3 chips, to £9,290.–£30,940.

To improve the performances over ambiguous concepts that could belong to several engineering disciplines, information should be integrated about the original document the requirements were extracted from. Related to the fine-tuning, the comparison could be extended to additional downstream tasks to further compare the performances of SpaceRoBERTa, SpaceSciBERT and SpaceBERT. CR can as well support additional text mining operations on the ECSS standards. Standards contain key information on space systems, and they are highly correlated. Thus, a follow-up task could be to associate similar requirements based on common concepts. This application could facilitate the identification of requirements relevant to a new project. Darm explores this application in his master thesis [165].

4.5.6 Conclusions

This study introduced SpaceTransformers a new family of three models: SpaceBERT, SpaceRoBERTa and SpaceSciBERT, providing contextualised words embeddings for space systems. Three novel domain specific models were further pre-trained from BERT-Base, RoBERTa-Base and SciBERT-scivocab on a domain-specific corpus. The pre-trained and further pre-trained models were evaluated on a CR task with a new labelled dataset of space systems concepts. All further pre-trained models outperformed their respective baseline models. The model further pre-trained from RoBERTa-Base, SpaceRoBERTa, achieved the most significant improvement, and the highest ranking. The statistical analysis however showed a lack of significant difference in mean ranking for the remaining models.

This new family of models can contribute to the fine-tuning of any NLP downstream tasks, improving the performances on domain-specific applications.

4.6 Chapter Summary

Findings from this chapter demonstrated that text mining and NLP methods could accelerate the development of domain-specific ontologies, by supporting the identification of the terminology, concepts and synonyms. These methods alone are, however, not enough yet, and domain experts are required to validate the outputs. These methods do enable a bottom-up approach which complements the classic top-down strategy.

The field of NLP is fast evolving, contextualised embedding were just emerging at the start of this thesis. The Transfer Learning tsunami originated by the Transformer architecture has deeply impacted the NLP landscape and will contribute to open further opportunities for text mining applications. Further pre-training and fine-tuning from large pre-trained models overcome the corpus size obstacle, enabling new applications in fields with few open source data set or labelled corpus.

Chapter 5

Task Automation

5.1 Chapter Overview

As highlighted by the survey results presented in Section 2.2, the overwhelming volume of accumulated data hinders the design process of space missions. Past accumulated data contain a wealth of information that experts want to access, for instance, they ranked past mission reports as their second preferred source of information in Figure 2.9. Knowledge reuse is, however, currently mostly done manually, or relying on colleagues' knowledge.

This chapter presents the domain-adaptation of two text mining methods, Topic Modelling and document embedding, to automate tasks usually done manually by systems engineers. A probabilistic model of Topic Modelling is trained to learn the probability distributions of topics related to key spacecraft subsystems, and automatically recognise the topics of mission requirements. A document embedding (doc2vec) model learns the representation vectors of past mission reports to identify similar missions and accelerate heritage analysis.

5.2 SpaceLDA: Requirements Categorisation with Topic Modelling

The approach, methodology and results (apart from the publications corpus) from this section were adapted from Berquand, A., Moshfeghi, Y., and Riccardi, A., "SpaceLDA: Topic Distributions Aggregation from a Heterogeneous Corpus for Space Systems" published in the journal of Engineering Applications of Artificial Intelligence [13]

The design of highly complex systems such as spacecraft are driven by hundreds of requirements evolving throughout the various design stages. Requirements Management, the process of documenting, analysing and tracking requirements, is an essential but time-consuming task. This section suggests a novel strategy based on Topic Modeling (TM) to facilitate the management of spacecraft design requirements, notably supporting their categorisation. TM is a ML method used to identify, learn, and extract latent topics from documents. Following a bottom-up approach, TM learns from a corpus of documents the word distributions of the topics found in the documents. The goal is thus to train a TM model on a domain-specific corpus to learn the word distributions of key spacecraft subsystems such as the Propulsion, Thermal, or Power subsystems. Once the model learns to recognise subsystems topics, it can link a requirement to its relevant subsystem.

A novel domain-specific semi-supervised Latent Dirichlet Allocation (LDA) model enriched with lexical priors and an optimised Weighted Sum (WS) called spaceLDA is thus trained on the corpus introduced in Section 3.2.1. The spaceLDA model is fine-tuned for the extraction of topics related to space systems. As the training corpus is heterogeneous, the spaceLDA model is independently trained on the four corpus subsets to avoid under-representing smaller corpora. To merge the topic distributions obtained from each model, a novel aggregation approach based on an optimised WS is applied. The training of the spaceLDA model is enriched by human-validated lexical priors encouraging the discovery of topics related to key spacecraft subsystems.

The spaceLDA model is then evaluated with a case study addressing the categorisation of spacecraft design requirements. Requirements extracted from public ESA reports, presented in Section 3.2.2, are submitted as unseen data to the spaceLDA model. The resulting aggregated topic distribution enables the association of each requirement with a spacecraft subsystem. The spaceLDA performances are compared to an unsupervised LDA model trained on the same heterogeneous training corpus. The performance of the WS aggregation method is compared to a state of the art aggregation method based on the Jensen-Shannon (JS) divergence. The results demonstrate that the spaceLDA model successfully identifies the topics of new (unseen) requirements and that the proposed approach surpasses the use of a classic LDA model and the state of the art aggregation method. To summarise, the research presented in this section makes the following contributions:

1. A novel domain-specific LDA model, named spaceLDA, enriched with lexical priors and an optimised WS is trained.

2. The spaceLDA approach is shown to outperform the unsupervised LDA model and a literature method for aggregating per-topic word distributions.

3. The potential for Requirements Management with Topic Modelling is demonstrated through a practical case study.

Section 5.2.1 introduces Topic Modelling, the LDA model and past applications of Topic Modelling in the field of space systems and Requirement Managements. Section 5.2.2 details the approach followed to train the spaceLDA model, and Section 5.2.3 the key building blocks of the methodology applied. Section 5.2.4 compares the per-topic word distributions obtained from each subcorpus, and Section 5.2.5 presents the results of the case study. The results of this study are discussed in Section 5.2.6. The source code and domain-specific models are available at `github.com/strath-ace/smart-nlp`.

5.2.1 Background

LDA was first introduced in [166] as a generative probabilistic model for discrete data collections. Topic Modelling assumes that a document is a mixture of topics, and an

LDA model represents a corpus of documents as a distribution probability over latent (hidden) topics. Each latent topic is described by a word distribution, a sorted list of words associated with a probability indicating their likelihood to belong to the latent topic. For instance, a *Propulsion* topic's distribution could include the words *thruster*, *engine* or *propellant*. LDA is an extension of Probabilistic Latent Semantic Analysis (pLSA) [167] using Dirichlet priors. The evolution of Topic Modelling methods, including pLSA, is provided in Appendix 8.2, Section 8.2.4. A Dirichlet distribution creates probability distributions summing to 1 and is commonly used as priors in Bayesian statistics to establish the prior belief of the probability distribution. Within an LDA model, each document is a probability distribution over topics, and each topic is a probability distribution over words. Thus, the probability distribution of topics T among a corpus of documents can be defined as in [168]:

$$p(M|\alpha, \beta) = \int p(\theta|\alpha) \left(\prod_{i=1}^{N} \sum_{z=1}^{T} p(z|\theta) p(w_i|\beta_z) \right) d\theta \tag{5.1}$$

where M is a document composed of N words w_i, z is the topic for the N-th word of document M, $p(z|\theta)$ is a multinomial distribution given by θ and followed by topic z, $p(w_i|\beta_z)$ is the probability that word w_i belongs to topic z given by β_z. β and α are the Dirichlet distribution parameters, respectively for the per-topic word distribution and for the per-document topic distribution. θ follows the hyperparameter α. Figure 5.1 displays the equivalent graphical representation of a LDA model as presented in [166].

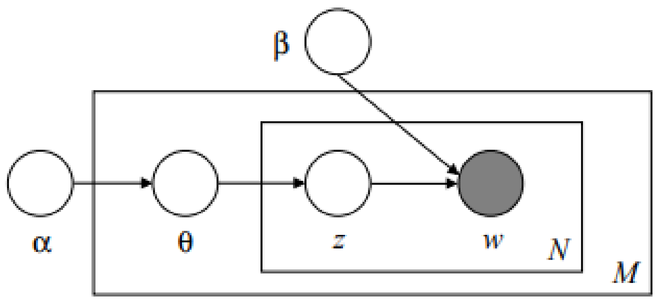


Figure 5.1: LDA model graphical representation as seen in [166]

Semi-supervised Latent Dirichlet Allocation

The LDA model can be trained in a semi-supervised fashion to guide the extraction of latent topics. The initial probability distribution of a word to belong to a topic, $p(w_i|\beta_z)$, is randomly set at the start of the modelling process. For a semi-supervised LDA, this initial probability can be increased at the start of the process to influence the composition of the per-topic word distribution. The concept of inputting lexical priors, or seed words, into a model is presented by Jagarlamudi et al. in [169]. With the Gensim Python library developed by [132] and used to train the spaceLDA model, η, a matrix representing for each topic, the probability of each word to belong to it, can be provided to the model to impose the asymmetric priors over the word distribution.

Application of Topic Modeling to space systems

While TM has been commonly used for text mining tasks, such as collaborative filtering [168] or trend forecasting [170, 171], applications in the space field are scarce.The publication of Layman et al., [172], is the closest work to the approach presented here. Layman et al. apply LDA to identify topics and trends in NASA problem reports. These reports include textual descriptions of anomalies detected during testing and operations, as well as details on the resulting corrective actions. TM successfully enables the authors to extract trends of reported anomalies from thousands of documents. From these previous studies found in the Literature, the use of TM at the early design stages of a space mission for requirements categorisation appears as a novel application.

Requirements Management and Machine Learning

Requirements Management is the process of documenting, analysing and tracking requirements. It is, therefore, an essential process for large-scale and complex projects. Iqbal et al. in [173], recently surveyed the ML methods applied to Requirement Engineering, noticing an increasing effort to merge both fields. The classification methods mentioned by the authors mostly include classic approaches such as Support Vector Machines, Conditional Random Field Network and Naïves Bayes. The authors only briefly mentioned Topic Modeling and the more recent method of word embedding, word2vec. The latter word embedding method proposed by Mikolov and his team in [123] enables the mapping of the context of a term into a vector. Word2vec was addressed in Section 4.4. Both TM and word embedding appear as novel methods for Requirements Management, let alone Requirements Management in the space field. However, without detailed preliminary knowledge of words most likely to describe each spacecraft subsystem, a word embedding approach could not be directly applied. The TM approach was therefore preferred as it enabled the definition of per-topic word distributions for each spacecraft subsystem. Based on the analysis of the current state of the art, the training of a domain-specific LDA tailored to space systems and its application to Requirements Management addresses a knowledge gap.

5.2.2 Approach

The core approach of the spaceLDA model training is based on two key components:

1. Lexical priors to steer the topics extraction.
2. An optimised WS to aggregate models trained on different corpus subsets.

Lexical priors reflecting key spacecraft subsystems can influence the extraction of topics relevant to space systems. The priors' probabilities are set to 0.95 while the probabilities of the remaining words are set to 0. This approach entails a semi-supervised training of the spaceLDA model. The lexical priors selection is further detailed in Section 5.2.3.

The spaceLDA approach proposes a novel method to aggregate topic distributions. A first approach to aggregate results of models trained on heterogeneous data is usually to alter the basic architecture of the LDA model and combine the data sets during the model training. In [174], the authors combined the Author Topic Model, developed by [175], with LDA to form a Heterogeneous Topic Model. However, a simpler and preferred approach would not alter the classic LDA architecture. Several authors have investigated post-training aggregations, meaning a merging of the per-document topic distributions. To aggregate models, a common first step is to identify similar topic distributions. Blei and Lafferty, [176], rely on a graph-based method. Blair et al., [177], compare the cosine similarity and the JS divergence to identify similar topics, proving the higher performance of the latter in creating more coherent topics. Blair et al. furthermore implemented the aggregation of models trained on an unchanging corpus but with varying Dirichlet priors and topic numbers.

The scope of this study differs from previous work as it intends to combine models trained on a heterogeneous corpus to capitalise on the diversity of the space mission design data collected. Combining independent models has the main advantage that it requires no modification of the underlying architecture of the basic LDA models. Thus allowing the use of standard libraries such as the Gensim Python library. In [178], the authors combine models pre-trained with various LDA parameters and subsets of the same corpus preprocessed with disparate methods. However, the authors do not aggregate similar topics together but rather consider that they form a large list of topic distributions. The concept of weighting and optimising the topic distributions obtained from different models over the same unseen data with an optimised WS is introduced here. Only in [179], the concept of weight matrix is mentioned but is adopted to regulate the influence of more recent inputs to update an online LDA model. The aggregation of per-document topic distributions based on an optimised WS with a state of the art method to aggregate per-topic word distributions based on the JS divergence will thus be compared.

Figure 5.2 summarises the approach to train the spaceLDA model. From the curated domain-specific text collection, a spaceLDA model is trained in a semi-supervised fashion with lexical priors. To avoid eclipsing smaller data sets, the model is trained separately on each training set based either on publications, feasibility reports, books, or Wikipedia pages. A requirement is extracted from the case study corpus and submitted to the models. Each model yields a per-requirement topic distribution, highlighting its most salient topics. To converge towards a single topic distribution, the distributions are aggregated based on an optimised WS.

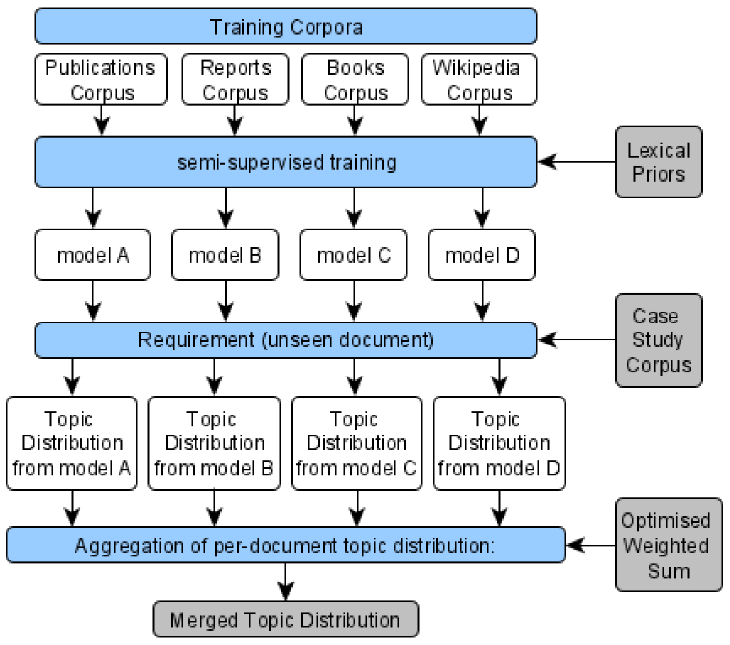

Training Corpora
Publications Corpus
Reports Corpus
Books Corpus
Wikipedia Corpus
semi-supervised training
Lexical Priors
model A
model B
model C
model D
Requirement (unseen document)
Case Study Corpus
Topic Distribution from model A
Topic Distribution from model B
Topic Distribution from model C
Topic Distribution from model D
Aggregation of per-document topic distribution:
Optimised Weighted Sum
Merged Topic Distribution

Figure 5.2: Overview of the spaceLDA approach

5.2.3 Methodology

Hyperparameters study

Three main inputs are required to train a model with the Python Gensim Library:

- the *dictionary*, which maps words, or tokens, to identification numbers (ids),
- the *corpus*, or *document-term matrix*, which provides per document, the words identification numbers and their frequency within the document,
- the *number of latent topics* to be discovered.

The Dirichlet prior alpha is set to $1/n$ where n is the number of topics. The number of training passes is set to 500. The first two inputs are derived from the corpus. To determine the number of latent topics, several spaceLDA models with different numbers of topics are trained with the Gensim Python library and compared. The evaluation metric of perplexity, presented in the next section, determines which model is best fitted to represent the corpus' topic distribution. The training corpus is split between a training and a testing set, following the classic 80%/20% partition. 5-fold cross-validation is applied to find the number of optimal topics and retain the final model. The testing set is used for the final evaluation of the retained model post-optimisation.

SpaceLDA model evaluation

Perplexity is an intrinsic evaluation metric used to evaluate LDA topics [166,170]. The same evaluation metrics was used in Section 4.5. Perplexity evaluates here how well the probability distribution generated represents the corpus and measures the likelihood that the model will perform well with unseen, new, data. The value of perplexity must be minimised, and is defined by Equation 4.17.

Topics labelling

The latent topics' word distributions produced by the model are not labelled. Therefore, human-validated labels are provided to ensure the reproducibility of the results. Three human annotators were involved in assigning topic labels to the word distributions, working independently and manually. A final label was elected when at least two of the three annotators agreed. Without a clear majority, the human annotators shortly debated to converge towards a single label. The annotators were given the following labels to choose from: *AOCS*, *Communication*, *Environment*, *Ground Segment*, *Launch*, *Mission Analysis*, *OBDH*, *Payload*, *Power*, *Propulsion*, and *Thermal*. The annotators also had the option to propose a topic label outside of this selection. It was made clear to the annotators that they could associate more than one label to each word distribution and that one label could be associated with several distributions. Word distribution associated with a same label are not merged, thus there can be several distributions within a model representing a same subsystem.

Lexical priors selection

Seven sets of lexical priors were defined in an attempt to steer the model towards topics corresponding to 7 key spacecraft subsystems: *AOCS*, *Communication*, *Environment*, *OBDH*, *Power*, *Propulsion*, and *Thermal*. Each set is composed of around 20 words, selected with domain-specific experts, based on the analysis of unsupervised models' distributions and on the results presented in Section 4.3. Each word can only belong to one set to avoid topic overlap. The priors selected, presented in Table 5.1, are based on a list of keywords or relevant concepts associated to each topic. The list is validated by the same human annotators who performed the manual labelling.

Table 5.1: Set of lexical priors per topics (organised alphabetically)

Topic Label	**Lexical Priors**
Attitude and Orbit Control Subsystem (AOCS)	angular, attitude, attitude control, body, freedom, gravity gradient, guidance, gyroscope, magnetotorquers, momentum, motion, navigation, reaction wheel, sensor, spin stabilised, stabilisation, star tracker, torque, torquer, wheel
Communication	antenna, band, bandwidth, c-band, command, communication, frequency, ka-band, l-band, receiver, reception, relay, satellite communication, s-band, telecommand, telemetry, tracking, transmitter, packet, x-band,
Environment	background, charging, cosmic, debris, dose, electron, environment, gamma radiation, gamma ray, geomagnetic, particle, protection, radiation, ray, shield, single event, shielding, single event upset, space debris, van allen
On-Board Data Handling	bit, bitrate, computer, cpu, data, data handling, data rate, decoder, downlink, dram, encoder, execution, gbit, instruction, measurement, memory, operation, processor, ram, sram, storage, tag, uplink
Power	battery, battery powered, cell, charge, circuit, current, cycle, depth of discharge, discharge, energy, lithium, photovoltaic, power, power supply, primary, secondary, solar cell, solar power, voltage, watt
Propulsion	delta v, electric, electric propulsion, engine, exhaust, fuel, impulse, ion, isp, nuclear, plasma, propellant, propellant mass, propulsion, propulsion system, sail, spacecraft propulsion, thrust, thruster, total impulse
Thermal	coating, cooling, degree, heat, heat pipe, heater, heating, insulation, louver, mirror, multi layer insulation, radiator, reflective, reflector, temperature, thermal, thermal control, thermal control system, thermodynamics, overheating

Optimised weighted sum

The WS approach acts on the level of the per-document topic distribution. $\hat{\theta}_i$ denotes the aggregated topic distribution for the unseen data, document i. The WS combines the topic distributions $\theta_{(i,j)}$ of each model M_j for the same document i but balanced by a model weight w_j as shown on equation 5.2 based on [180].

$$\hat{\theta}_i = \frac{\sum\limits_{j=1}^{M} w_j \theta_{(i,j)}}{\sum\limits_{j=1}^{M} w_j} \tag{5.2}$$

Since this method does not produce new word distributions, it does not entail the tedious process of relabelling. The weights are optimised with a Tree of Parzen Estimators (TPE) algorithm [181] available through the hyperopt Python library [182]. The objective function maximises the area of the accuracy plot. Figure 5.3 summarises the WS aggregation approach.

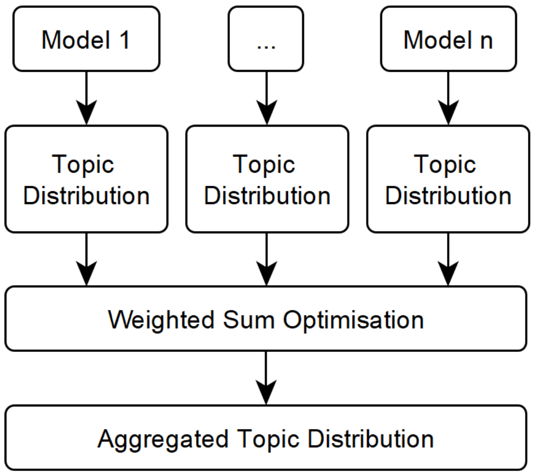


Figure 5.3: Schema of the proposed aggregation method

Topic identification of unseen data

The dictionary of the model is used to map words to their ids. A new corpus document-term matrix is generated based on this dictionary. The topic distribution defined by the spaceLDA model can then be applied to the new (unseen) document. The output is a list of latent topics along with their probability to represent the document.

Case study evaluation: Accuracy Score and Mean Reciprocal Ranking

The accuracy score only takes into consideration the primary topic of a per-document topic distribution. If this topic matches the requirement's ground truth, then the matching is considered a success. The accuracy score is divided by the number of unseen data, or requirements, submitted to the model. Therefore, the best performance corresponds to an accuracy score of 1.

The Mean Reciprocal Ranking (MRR) takes into consideration the top n topics of a per-document topic distribution. The score is inversely proportional to the correct answer, topic rank, as shown in Equation 5.3 based on the definition from [183]:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{5.3}$$

with Q the number of queries, $rank_i$, the rank of the ground truth. Only the top two topics will be taken into consideration.

5.2.4 Model Training Results

A hyperparameter optimisation is run to identify the optimal number of latent topics for each sub-corpus. Four LDA models are then trained with lexical priors in a semi-supervised fashion. Their resulting per-topic word distributions are analysed and compared with distributions generated by unsupervised LDA models trained on the same domain-specific corpora. At this stage, the models are not aggregated.

Hyperparameters optimisation

For each training corpus, the optimisation process described in Section 5.2.3 is run to identify the optimum number of latent topics. The optimisation process is run for a number of topics ranging from 4 to 100 and for each training set. The resulting average perplexity measures are displayed on Figure 5.4 up to 40 topics.

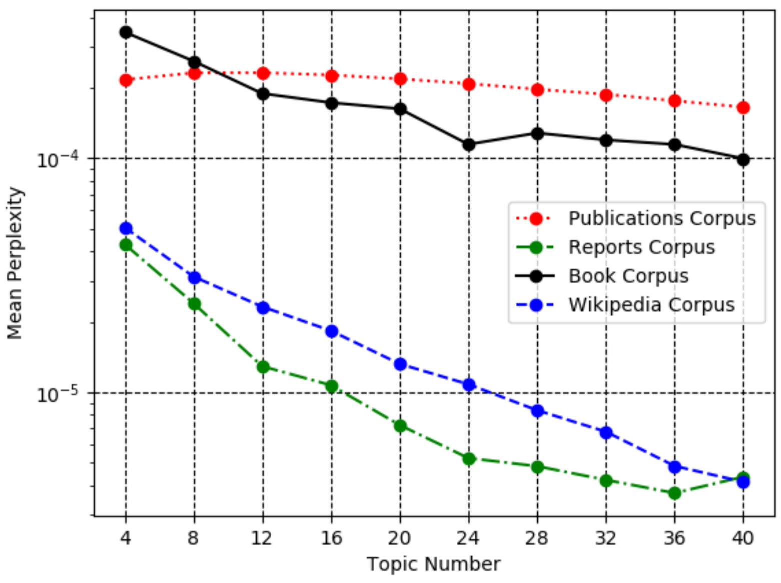


Figure 5.4: Perplexity evolution

Setting a perplexity threshold to 0.9E-4 and manually investigating the topics content of models satisfying the perplexity threshold, a topic number of 22 was chosen for the Wikipedia training set. Following the same process, a topic number of 30 latent topics was selected for the reports training set. A model was trained with this set, obtaining a perplexity of 1.35e-06 in its final evaluation, computed with the held-out part of the training corpus. The higher number of latent topics was assumed to result from the variety of missions covered by the reports, as well as the higher complexity of the experts' vocabulary writing these reports. For the books training set, a topic number of 24, corresponding to the local minimum, was chosen. A final model obtained a perplexity score of 1.85e-05, similar scores to the above models, although the values of the mean perplexity measures are generally higher for this training set. The publications corpus representing a significantly higher amount of data requires a higher number of topics. To keep the number of topics at a manageable level, the threshold requirement is waived. The perplexity analysis is completed with a manual verification of the topic word distributions, and thus a topic number of 40 is selected. A final model is trained on the publications corpus for 40 topics and achieves a perplexity of 1.69E-4 on the training set.

SpaceLDA per-topic word distributions

The LDA model training is a stochastic process; however, the results presented in this subsection represent trends observed with several trained models. The spaceLDA models are trained using the 164 lexical priors presented in Table 5.1. Not all lexical priors could be found in the corpora' dictionaries. Therefore nineteen words, including the terms "*bit*", "*cpu*", "*magnetotorquers*", "*power supply*", "*satellite communication*", "*spacecraft propulsion*" and "*spin stabilised*", are not boosted. Unsupervised models are also trained on the same corpora to compare the latent topics extracted with and without lexical priors.

Tables 5.2, 5.3, 5.4, and 5.5 display the top 5 terms of per-topic word distributions extracted from the training corpora by the spaceLDA and the unsupervised LDA models. In these tables, the lexical priors are highlighted in blue and bold. Terms found in unsupervised word distributions that happened to match a lexical prior are underlined for comparison purposes. The complete word distributions of each model are available at `https://github.com/strath-ace/smart-nlp`.

The distributions obtained from the publications, Table 5.2, and the Wikipedia, Table 5.5, training corpora are similar for both training approaches. The unsupervised model seems to already be able to identify the topics of interest within these corpora without the priors guidance. The number of lexical priors found in the publications-based distributions is few. The large vocabulary of the publications corpus might have reduced the impact of the lexical priors.

With the reports training corpus, Table 5.3, complex phrases (more than 2-grams words) such as '*ultra high frequency*' seems to be given less attention in the semi-supervised distributions influenced by lexical priors. The unsupervised distributions promote terms which are less domain-specific, such as '*laser*' (found in an *Environment* topic), '*bipods*' or '*telescope*' (both found in a *Thermal* topic).

Finally, the unsupervised distributions extracted from the books, Table 5.4, appear to combine different topics. For instance, the *Communication* topic unsupervised word distribution includes the term '*low thrust*', a propulsion concept. The *Propulsion* topic distribution includes the term '*thermal control*'. Overall the word distributions extracted by the spaceLDA model provided a more accurate representation of each spacecraft subsystem. The lexical priors notably enabled to remove noisy terms from distributions based on the reports and books corpora.

The word distributions of each model are manually labelled. Figure 5.5 summarises the labels obtained for each corpus. Its large number of topics and corpus size allows the publications-base model to cover most topics of interest. A significant improvement is noticed between the unsupervised and the semi-supervised training. Three topics which were not previously extracted, *OBDH*, *Payload*, and *Power*, are now found. The priors are thus successful to encourage the extraction of topics that might be less prevalent and are overlooked by the unsupervised model. Within the Wikipedia corpus, most of the topics of interest could also be identified. The variation between the spaceLDA models and the unsupervised training is again minimal for this corpus. The benefits of lexical priors is more apparent for the other training sets. In the case of the reports corpus, it is clear that the unsupervised model focused on two topics, *Propulsion* and *Environment*. The priors balanced this attention, enabling the expression of other topics of interest. Similarly, in the case of the books corpus, several noisy topics (tagged as *Other* and not represented in the Figure) were extracted by the unsupervised model. In conclusion, the lexical priors contributed to the extraction of more relevant topics.

To further analyse the content of word distributions, the average numbers of lexical priors found in the top 50 of each topic word distributions is computed and displayed in Figure 5.6. For the unsupervised models, the priors are not used as lexical priors as they don't interfere in the training process, however, they can still appear in the word distributions. For the publications-based models, the average of lexical priors actually slightly decreases from $15,6\%$ to 13.2% from the unsupervised to the semi-supervised training. The average of the unsupervised model is driven by the high percentage of lexical priors found in the *Launch* topic. Nevertheless the semi-supervised training introduces lexical priors in the definition of several other topics. For the model trained with the Wikipedia set, the unsupervised topics, considering all categories, contain on average 18,1% of lexical priors against 19,2% for the spaceLDA model. For the model trained with the reports or books sets, the difference is far more prominent. For the reports training set, over all categories, the average for the spaceLDA topics is 44%, compared to 4% for the unsupervised model. Similarly, for the books corpus, the average number of lexical priors increased from 8% to 31% with the spaceLDA approach.

Table 5.2: Comparison of per-topic word distributions obtained from spaceLDA and unsupervised models trained with the publications corpus. Terms in blue and bold correspond to lexical priors.

Topic Label	**Training**	**Topic Word Distribution Top 5 elements**
Attitude and Orbit Control Subsystem	spaceLDA	**attitude**, spacecraft, **angular**, space, **torque**
	Unsupervised	**attitude**, spacecraft, **torque**, **angular**, target
Launch	spaceLDA	rocket, fuel, rate, combustion, propellant
	Unsupervised	thrust, thruster, rocket, engine, propulsion
Mission Analysis	spaceLDA	orbit, time, transfer, satellite, trajectory
	Unsupervised	satellite, orbit, mission, constellation, orbital
On-board Data Handling	spaceLDA	error, **measurement**, estimation, state, filter
	Unsupervised	image, feature, detection, frame, processing
Thermal	spaceLDA	**temperature**, **heat**, model, **thermal**, flow
	Unsupervised	**temperature**, **heat**, **thermal**, flow, wall

Table 5.3: Comparison of per-topic word distributions obtained from spaceLDA and unsupervised models trained with the reports corpus. Lexical priors are highlighted.

Topic Label	**Training**	**Topic Word Distribution Top 5 elements**
Communication	spaceLDA	**x-band**, **band**, **telemetry**, modulation, **telecommand**
	Unsupervised	ultra high frequency, orbiter, localisation, conjunction, arrival
Environment	spaceLDA	**radiation**, **shielding**, **particle**, **environment**, **electron**
	Unsupervised	bench, interferometer, decoherence, nanoparticles, laser
Power	spaceLDA	**energy**, capacity, panel, solar power, **voltage**
	Unsupervised	laser interferometer space antenna, electric propulsion, laser, constellation, telescope
Propulsion	spaceLDA	**propellant**, transfer, refuelling, optical, geostationary orbit
	Unsupervised	**electric propulsion**, asteroid, eprop, boost, arrival
Thermal	spaceLDA	**overheating**, **thermodynamics**, **reflective**, **thermal control system**, **thermal control**
	Unsupervised	**cooling**, telescope, cryogenic, spectro, bipods

Table 5.4: Comparison of per-topic word distributions obtained from spaceLDA and unsupervised models trained with the books corpus. Lexical priors are highlighted.

Topic Label	**Training**	**Topic Word Distribution Top 5 elements**
Attitude and Orbit Control Subsystem	spaceLDA	**attitude**, vector, matrix, control, frame
	Unsupervised	quaternion, covariance, kalman, kinematics, architect
Communication	spaceLDA	orbit, service, data, **antenna**, **communication**
	Unsupervised	decentralised, nonlinear, synchronisation, topology, low thrust
On-Board Data Handling	spaceLDA	on board computer, **data**, interface, frame, power, channel
	Unsupervised	on board computer, power control and distribution unit, connector, register, spacewire
Propulsion	spaceLDA	**propulsion**, **fuel**, electric, **thruster**, **propulsion system**
	Unsupervised	nozzle, packet, combustion, thermal control, qualification
Thermal	spaceLDA	**thermal**, **heat**, **temperature**, control, orbit
	Unsupervised	**thermal control**, **insulation**, **coating**, pumped, vapour

Table 5.5: Comparison of per-topic word distributions obtained from spaceLDA and unsupervised models trained with the Wikipedia corpus. Lexical priors are highlighted.

Topic Label	**Training**	**Topic Word Distribution Top 5 elements**
Attitude and Orbit Control System	spaceLDA	**momentum**, **angular**, velocity, **motion**, particle
	Unsupervised	**attitude**, **sensor**, **wheel**, orientation, **momentum**
Communication	spaceLDA	radio, **frequency**, signal, **antenna**, **receiver**
	Unsupervised	radio, **antenna**, **receiver**, signal, wave
Environment	spaceLDA	**cosmic**, **radiation**, **particle**, belt, allen
	Unsupervised	**radiation**, gamma, **cosmic**, **particle**, decay
On-board Data Handling	spaceLDA	**memory**, dynamic random access memory, cable, **data**, cell
	Unsupervised	**memory**, dynamic random z access memory, cable, cell, **computer**
Power	spaceLDA	**cell**, **power**, **photovoltaic**, pressurised pressure vessel, electricity
	Unsupervised	capacitor, **voltage**, **circuit**, capacitance, resistance
Thermal	spaceLDA	**heat**, **heating**, **temperature**, material, **thermal**
	Unsupervised	**heat**, **temperature**, **thermal**, **heat pipe**, **cooling**

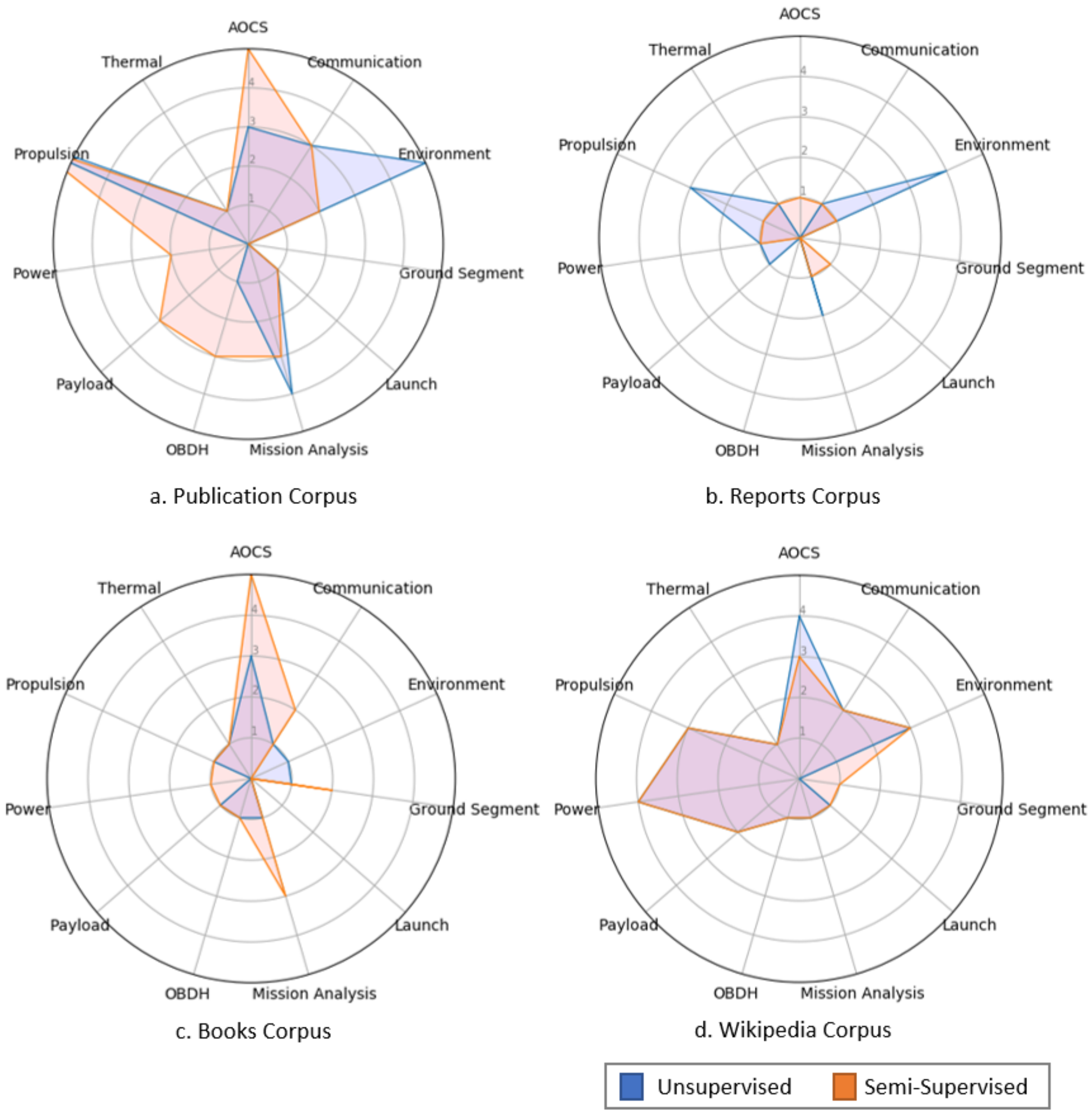


Figure 5.5: Evolution of topics labelling between unsupervised models (blue distribution) and spaceLDA (semi-supervised) (orange distribution) for each training set.

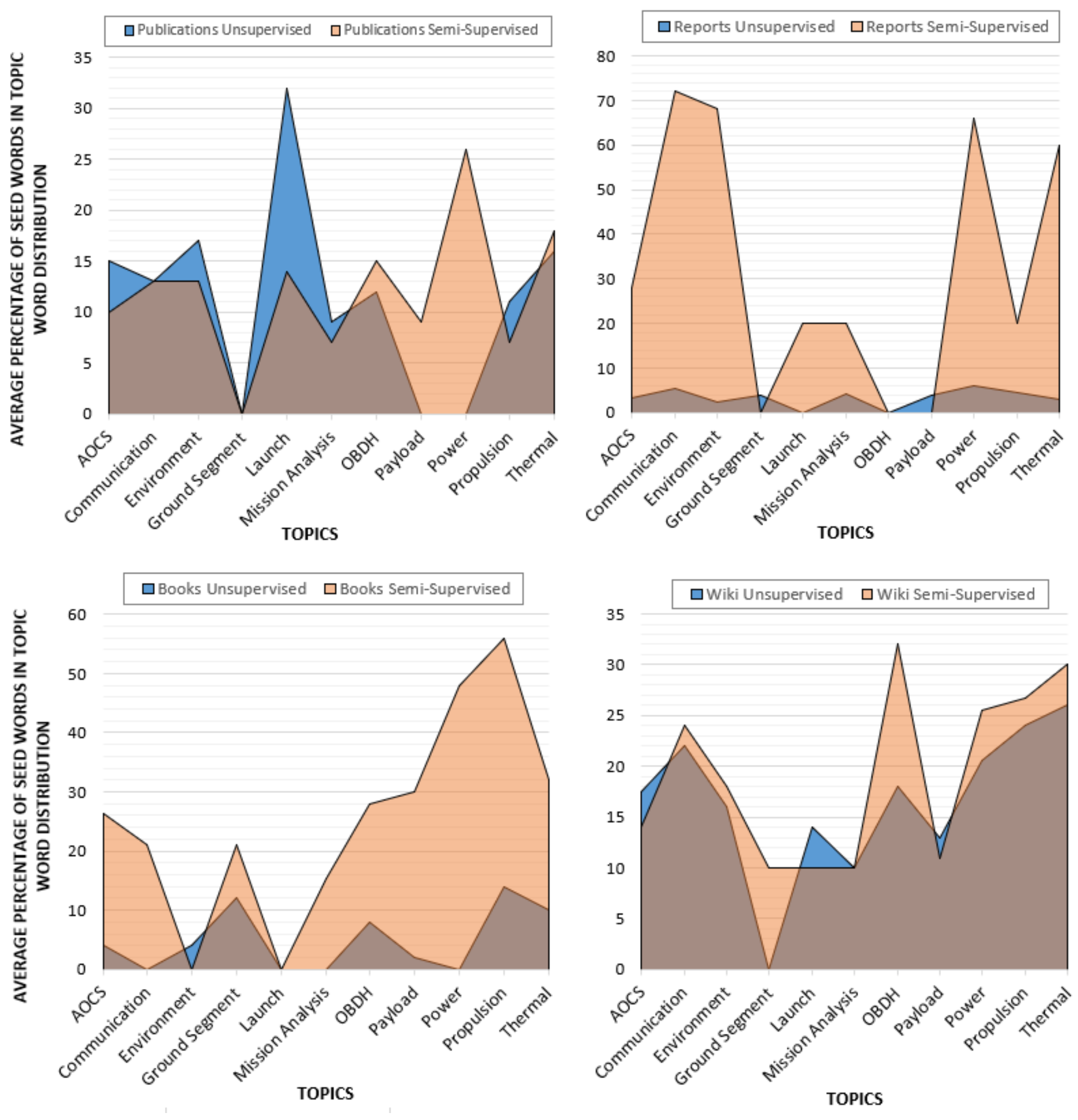


Figure 5.6: Average percentage of lexical priors found in each topic distribution and each training set with the spaceLDA (semi-supervised) or unsupervised models.

5.2.5 Case Study Results

This section presents the aggregation methods parameters and the case study results. A selection of design requirements are submitted as new unseen documents to the spaceLDA model. The model provides in return a topic distribution for each requirement. These are then compared to the requirements' ground truth. The latter ground truth corresponds to the chapter the requirement was extracted from. For instance, all requirements related to the *Thermal* subsystem will be extracted from the relevant *Thermal Subsystem* chapter and their ground truth labelled as *thermal*.

The performance of the spaceLDA is compared to three approaches, either based on a semi or unsupervised training, and aggregated with the optimised Weighted Sum (WS) or the Jensen-Shannon divergence. All assessed methods are summarised in Table 5.6. To ensure robustness, the training process is run ten times for both training approaches and each of the 4 training set, yielding in total 40 spaceLDA models and 40 unsupervised LDA models. Each aggregation method is applied to each set of 40 models. For instance, with the *method c*, the 40 unsupervised LDA models are aggregated with the JS divergence.

Table 5.6: Overview of compared methods in the case study

	spaceLDA	**method a**	**method b**	**method c**
Training with lexical priors	X	X		
Unsupervised training			X	X
Optimised WS aggregation	X		X	
JS divergence aggregation		X		X

Aggregation of per-document topic distribution with an optimised weighted sum

With this aggregation method, the merging occurs after each model has generated a topic distribution for the unseen document. The contribution of each model is balanced with weights to optimise the categorisation performance. The hyperparameter optimisation ran with the hyperopt Python library yields the following linearised weight combinations:

1. SpaceLDA models: 0.55 for the Wikipedia-based, 0.38 for the book-based, 0.056 for the report-based, and 0.019 for publication-based models.

2. Unsupervised models (*method b*): 0.70 for the Wikipedia-based, 0.16 for the publication-based, 0.13 for the report-based, and 0.015 for the book-based models.

For this case study, the weights of the Wikipedia-based are significantly higher than for the other training corpora. Indeed, selected Wikipedia pages are more likely to briefly describe the architecture of space systems than the feasibility reports detailing applied examples of spacecraft design or the books and publications corpus covering broader topics. Therefore, the Wikipedia corpus is given more influence with a higher weight. The weights would need to be fine-tuned again for another case study.

Aggregation of per-topic word distribution with the Jensen-Shannon divergence

As presented by Blair et al. in [177], the Jensen-Shannon (JS) divergence enables the symmetric measurement of similarity between two or more probability distributions. A value of the JS divergence equals to 0 indicates a complete similarity and a value of 1 a complete dissimilarity. Provided the divergence between n similar topics of M models with T_i topics is lower than the JS divergence threshold, γ, an aggregated topic $\hat{\varphi}_k$ can be generated following equation 5.4 based on [177]. $\varphi_{(i,j)}$ being the per-topic word distribution of topic T_j in model M_i.

$$\hat{\varphi}_k = \begin{cases} \sum_{i=1}^{M} \sum_{j=1}^{T_i} \frac{\varphi_{(i,j)}}{n}, & \text{if } D_{JS}(\varphi_{(i,j)} || \varphi_x) \leq \gamma \\ 0, & \text{otherwise} \end{cases} \tag{5.4}$$

The aggregation process is separately run on the 40 unsupervised models (*method c*) and on the 40 spaceLDA models (*method a*). In each case, 40 models (10 per training set) amount to 1,160 word distributions to be aggregated into one model. To compare the distributions obtained from heterogeneous sources, a new dictionary is generated based on the vocabulary gathered from all word distributions. The probability distributions are reorganised according to this common dictionary. The JS divergence is computed for each per-topic word distribution with regards to the 1,159 other distributions. A threshold of 0.3 is set to retain only the closest distributions. All topic distributions with a JS divergence lower than this threshold are aggregated. Otherwise, topics are kept as such. Eventually, the unsupervised models yield one aggregated model with

906 topics, while the semi-supervised models yield one aggregated model with 829 topics. 196 word distributions were thus aggregated for the unsupervised models and 373 for the semi-supervised models. This demonstrates that the lexical priors yield more similar per-topic word distributions. The topics are manually labelled by human annotators. The unseen data will be converted to a bag of words based on the aggregated model dictionary.

Categorisation results and comparison

The case study corpus includes 68 mission requirements, presented in Section 3.2.2, related to the topics of *AOCS*, *Communication*, *Environment*, *OBDH*, *Power*, *Propulsion*, and *Thermal*. The documents' chapters, corresponding to the spacecraft's subsystems, from which the requirements are extracted are used as ground truths. Accuracy and Mean Reciprocal Ranking (MRR) are used to evaluate the models' performances. The performances of the different models are compared in Tables 5.7- 5.8 where the spaceLDA method clearly outperforms the other approaches.

Table 5.7: Categorisation Accuracy - the highest score per category are underlined in bold and the results of the proposed method are highlighted in the grey column.

Training		Lexical Priors		Unsupervised	
Aggregation Method		WS (spaceLDA)	JS Divergence (method a)	WS (method b)	JS Divergence (method c)
Labels	**AOCS**	0.64	0.36	**0.73**	0.55
	Communication	**0.7**	0.2	0.6	0.4
	Environment	**0.2**	0.1	**0.2**	0
	OBDH	**0.6**	0.5	0.3	0.1
	Power	0.64	0.27	**0.73**	0.36
	Propulsion	**0.8**	0	**0.8**	0.4
	Thermal	**0.73**	0.18	0.55	0.64
Average Accuracy		**0.61**	0.23	0.56	0.35
Average Ranking		**1.43**	3.57	1.86	3.14
Control Method		**Bonferroni-Dunn Test**			
SpaceLDA		-	**2.14**	0.43	**1.71**

Table 5.8: Categorisation MRR - the highest scores per category are underlined in bold and the results of the proposed method are highlighted in the grey column.

Training		Lexical Priors		Unsupervised	
Aggregation Method		WS (spaceLDA)	JS Divergence (method a)	WS (method b)	JS Divergence (method c)
Labels	**AOCS**	**0.73**	0.45	**0.73**	0.59
	Communication	**0.7**	0.4	0.65	0.45
	Environment	**0.2**	**0.2**	**0.2**	0.05
	OBDH	**0.7**	0.6	0.3	0.25
	Power	0.68	0.32	**0.77**	0.45
	Propulsion	**0.8**	0.5	**0.8**	0.5
	Thermal	**0.82**	0.36	0.68	**0.82**
Average MRR		**0.66**	0.4	0.59	0.44

The accuracy results have been statistically analysed with the Friedman pre-hoc, and the Bonferroni-Dunn and Nemenyi post-hoc tests. To determine the statistical significance of the accuracy score of each method with respect to the categories set, a non-parametric Friedman test is completed with the ranking of the best model set as test variable. The Friedman test shows that the proposed method is statistically significant at a level of 5% as the confidence interval is $C_0 = (0, F_5 = 3.16)$ and the F-distribution statistical values is $F^* = 9.98 \notin C_0$. Therefore the Friedman test rejects the null-hypothesis that all models perform equally well. Following this rejection, a Nemenyi post-hoc test is completed to compare the performances of the different models. The difference in ranking, as resulting from the Nememyi tests can be observed in Figure 5.7, for $\alpha = 0.05$, where the critical difference (CD) is of 1.77. The results of the Bonferroni-Dunn test for $\alpha = 0.05$ are reported in Table 5.7. The Bonferroni-Dunn critical difference is of 1.65. From the results of both tests it can be concluded that spaceLDA has a significant higher ranking than method a and c. SpaceLDA however does not have a significant difference in mean ranking with respect to method b, suggesting that the aggregation method choice contributed more than the use of lexical priors to the performance increase.

In Tables 5.9, 5.10, 5.11, 5.12, and 5.13, samples of per-requirement topic distributions obtained with both training and aggregation approaches are displayed. Each distribution is a probabilistic distribution of the topics most likely to be found in the

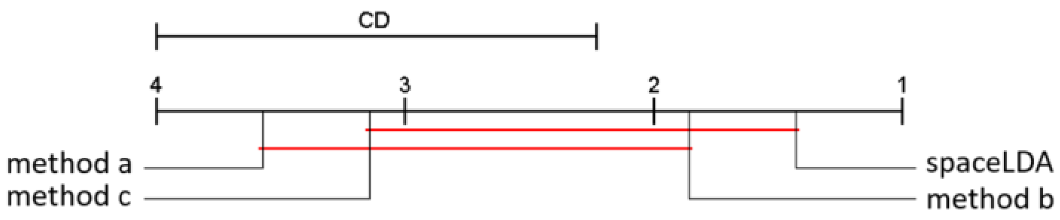


Figure 5.7: Nemenyi CD diagram comparing the generalisation accuracy score rankings of the different methods ($\alpha = 0.05$).

assessed requirement. The spaceLDA model notably outperforms the other methods for the *OBDH* and *Communication* labels, as illustrated in Tables 5.9-5.10. The spaceLDA model also achieves a high accuracy for the *Thermal* category, while the *method a* obtains the lowest accuracy. This trend is reflected in Table 5.11 where the spaceLDA, *methods b* and *c* successfully associate the requirement to its correct category, while *method a* selects it as second choice.

On the other hand, as seen in Tables 5.7-5.8, the categorisation of *Power* requirements is one of the only two categories for which the spaceLDA is outperformed by another approach. Even then, the WS approach, relying on the unsupervised models' aggregation, performs better than the JS divergence aggregation. To improve the performance of the spaceLDA model, the lexical priors used to identify and define the *Power* topics should be improved. This trend is illustrated by an example in Table 5.12, where only the *method b* succeeds in identifying the main topic from the first try.

Finally, as shown in Table 5.13, a last example is provided for an OBDH requirement. All methods successfully identify the requirements' topics either as a first or second choice.

Table 5.9: Example of topic distributions obtained for an OBDH requirement. The ground truth topic is underlined in bold in the distributions.

OBDH Requirement		*The DHS shall provide reconfiguration capabilities in case of failure detection.*
Training	**Aggregation**	**Topic Distribution**
Lexical Priors	WS	**'OBDH'**: 0.21, 'communication': 0.14
	JS	'AOCS': 0.75
Unsupervised	WS	'launch': 0.3, 'AOCS': 0.17
	JS	'AOCS': 0.5, **'OBDH'**: 0.17

Table 5.10: Example of topic distributions obtained for a communication requirement. The ground truth topic is underlined in bold in the distributions.

Communication Requirement		*All images taken by navigation cameras and required to be sent to ground (e.g. asteroid shape model, local slopes around sampling sites, etc), if any, shall be downloaded.*
Training	**Aggregation**	**Topic Distribution**
Lexical Priors	WS	**'communication'**: 0.3, 'mission analysis': 0.19
	JS	'other': 0.65, **'communication'**: 0.26
Unsupervised	WS	'mission analysis': 0.16, 'environment': 0.15
	JS	'OBDH': 0.63, 'other': 0.27

Table 5.11: Example of topic distributions obtained for a Thermal requirement. The ground truth topic is underlined in bold in the distributions.

Thermal Requirement		*The TCS shall ensure survival thermal environment under the established anomaly conditions.*
Training	**Aggregation**	**Topic Distribution**
Lexical Priors	WS	**'thermal'**: 0.33, 'other': 0.19
	JS	'communication': 0.42, **'thermal'**: 0.23
Unsupervised	WS	**'thermal'**: 0.30, 'mission analysis': 0.16
	JS	**'thermal'**: 0.7 , 'other': 0.15

Table 5.12: Example of topic distributions obtained for a Power requirement. The ground truth topic is underlined in bold in the distributions.

Power Requirement		*Cell performance and degradation factors shall be justified according to in orbit experience and supporting ground testing.*
Training	**Aggregation**	**Topic Distribution**
Lexical Priors	WS	‘other’: 0.24, **‘power’**: 0.17
	JS	‘propulsion’: 0.86
Unsupervised	WS	**‘power’**: 0.30, ‘propulsion’: 0.16
	JS	‘other’: 0.46, **‘power’**: 0.36

Table 5.13: Example of topic distributions obtained for an OBDH requirement. The ground truth topic is underlined in bold in the distributions.

OBDH Requirement		*The payload module data handling functionality shall be implemented in a correlator and control unit (CCU).*
Training	**Aggregation**	**Topic Distribution**
Lexical Priors	WS	**‘OBDH’**: 0.4, other: 0.1
	JS	**‘OBDH’**: 0.75 , ‘communication’: 0.13
Unsupervised	WS	**‘OBDH’**: 0.21, launch: 0.17
	JS	other: 0.63, **OBDH**: 0.37

5.2.6 Discussion

The semi-supervised training of the spaceLDA model outperforms the classic LDA unsupervised training. The outputs of the unsupervised models were, however, useful in supporting the definition of the lexical priors which are then used for the spaceLDA training. The heterogeneity of the training data sets meant that, for each design requirement, as many topic distributions as models were found. Hence the proposition to merge distributions with an optimised WS to converge towards a single per-requirement topic distribution. This aggregation method outperformed the JS divergence aggregation method. The WS is a more flexible option allowing to balance the influence of the different corpora, enabling fine-tuning depending on the case study. In addition, this method does not require relabelling the topics of the merged model, enabling quicker re-use of pre-trained models. The WS optimisation assigned heavier weights to the Wikipedia corpus. This was expected as the per-topic word distributions based on the Wikipedia corpus covered most of the topics of interest and were of high quality. The selected Wikipedia pages solely targeted spacecraft subsystems, and were more likely to efficiently and shortly describe the architecture of space systems. On the other hand, the feasibility reports provided applied examples of spacecraft design and the publications and books were larger corpora and thus covered broader ranges of topics.

5.2.7 Conclusion & Future Work

This section introduced a novel domain-specific TM model, spaceLDA, tailored to space systems, enriched with lexical priors and an optimised WS. The practical application of TM to support space mission design was established through a case study on the categorisation of design requirements. The statistical analysis demonstrated the significant higher ranking of spaceLDA with respect to methods a and c, thus showing that the optimised WS method outperforms the state of the art aggregation method based on the JS divergence. SpaceLDA showed a lack of significant difference in mean ranking with respect to method b, thus suggesting that the aggregation method had a higher impact on the performances than the usage of lexical priors.

Although applied to a corpus related to space systems, the approach proposed here is extendable to any domain-specific corpus. In future work, the application of TM could be extended to the classification of documents. The TM approach could be compared to a word embedding approach. Labels could be automatically assigned to topics to mitigate the subjectivity of the manual labelling.

5.3 Document Embedding for Heritage Analysis

The doc2vec model used to generate the results presented here was first published in Berquand, A. & Riccardi, A., "From Engineering Models to Knowledge Graph: Delivering New Insights Into Models" at the 9th International Systems & Concurrent Engineering for Space Applications Conference (SECESA 2020) [12]. The methodology and results were presented to the ESA CDF team in February 2021.

Heritage analysis, screening for past similar missions, is one of the first tasks performed by engineers in preparation of a feasibility study. Experts may be inspired from previous design solutions, anticipate trade-offs, and overall avoid "re-inventing the wheel". The heritage analysis contributes to refining the initial design parameters on which the design convergence will depend. This process is critical as it helps kick-off the study. According to the expert survey presented in Section 2.2, the identification of past similar studies is mostly done through the interactions with experienced colleagues. The ESA CDF is also currently deploying a study portal where past feasibility report are identified by a set of metadata. Metadata keyword search is another basic tool for heritage analysis but it does require human annotation.

The challenge of automatic heritage analysis is addressed twice in this thesis. The approach presented in this section is based on unstructured data, the ESA CDF feasibility reports introduced in Section 3.2.1, and document representation vectors. In Chapter 6, the similarity of missions is assessed from the semi-structured data contained in the Engineering Models introduced in Section 3.3 and based on the ESA CDF's metadata list. The doc2vec model presented in this section is involved in the similarity computation of both case studies.

This section is meant as a brief add-on to Section 4.4 discussing word representation vectors. A doc2vec model, the document-level extension of word2vec, is for the first time trained on a corpus related to space systems and applied to the discovery of past similar space missions.

Section 5.3.1 provides the background on document embedding methods, focusing on the doc2vec model and past similar work. Section 5.3.2 briefly describes the approach to map full study reports or chapters to representation vectors with a doc2vec model. Section 5.3.3 presents the hyper-parameters of the doc2vec model and an analysis of the chapters found in the study reports. Finally, the representation vectors obtained with the trained doc2vev model are used to find similar missions, and the results for a case study are detailed in Section 5.3.4.

5.3.1 Background

Document embedding methods

Bag-of-words and Topic Modelling are classic document representation methods. With the BOW approach, summarised in Figure 5.8, a document is mapped to a vector of length n, corresponding to its vocabulary size. Each vector element of index i corresponds to the frequency of a word w in the document. This method could be seen as the document-level equivalent of OHE. The comparison between vectors of documents is then the equivalent to comparing the frequency lexica of the documents. Depending on the vocabulary size, this method yields sparse representations which increase the computation cost. Topic Modelling, introduced in Section 5.2, represents documents as topic distributions. Similar documents should cover similar topics. Both of these methods lack to take into consideration the word's context.

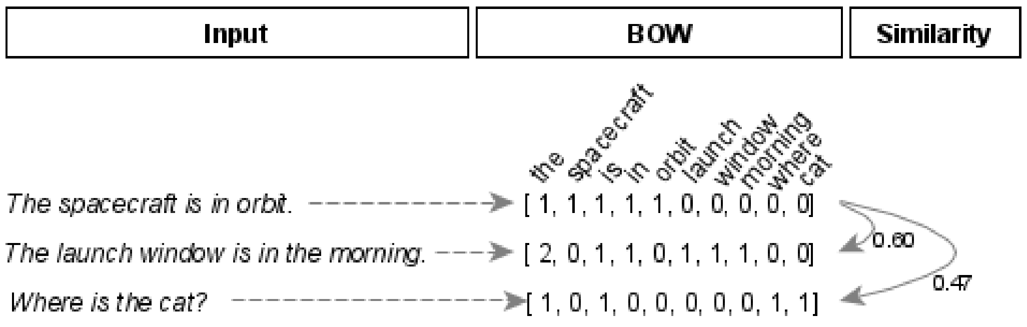


Figure 5.8: Bag-Of-Word example, the vectors are compared with the cosine similarity. The first sentence is found to be more similar to the second sentence than to the third random sentence.

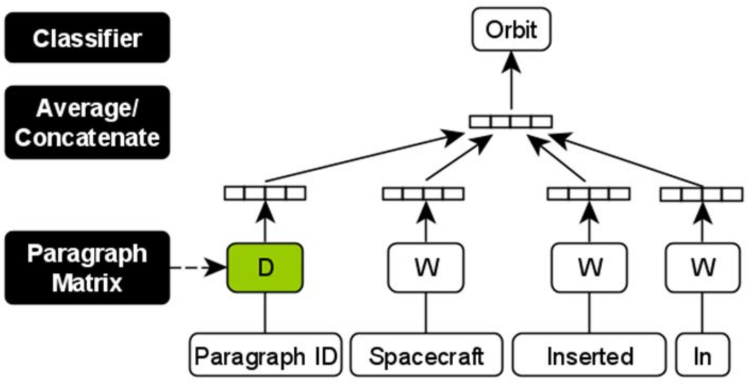
Similarly to the trends seen for word embedding, predicting methods have grown more popular than count-based methods. Le and Mikolov introduced in [184] the Paragraph Vector model, also called doc2vec, an unsupervised framework to learn the distributed vector representations of pieces of text. Doc2vec is de facto a document-level extension of the word2vec model presented by Mikolov et al. in [123, 125]. The target paragraph can be of any size, from a sentence to a document.

Several authors build upon the word2vec architecture to learn sentence-levels embedding. Renter et al. in [185] combine and average the vectors of the words find in a document to learn the document vector. Pagliardini et al. propose Sent2Vec in [186], combining the CBOW architecture of word2vec, multi-words embedding and vectors averaging. Kiros et al. took a different approach with Skip-Through vectors in [187]. Their method is based on Recurrent Neural Networks and takes into account the word order. A more recent model of SentenceBERT presented in [188] is based on a transformer architecture to be presented in Section 4.5.

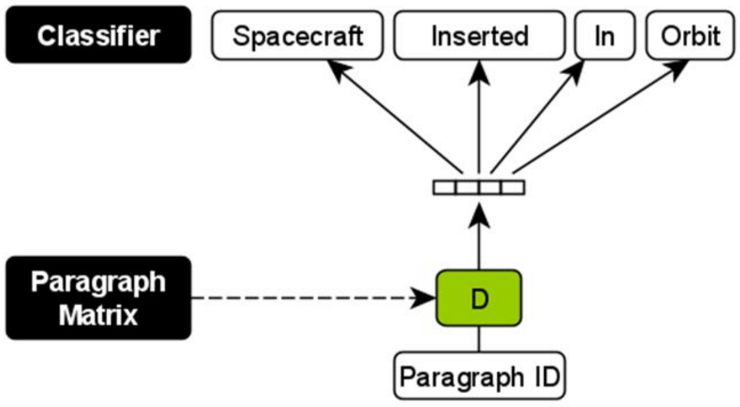
Doc2vec architecture

The doc2vec model architecture is inspired from word2vec. As a matter of fact, Le and Mikolov introduce two different architecture for their Paragraph vector model: the Distributed Memory (DM) version based on the word2vec CBOW architecture, and the Distributed Bag of Words (DBOW) version based on the Skip-gram architecture. Both are respectively illustrated in Figure 5.9a and 5.9b.

In the DM architecture, each paragraph is mapped to a unique vector stored in the matrix D, and each word embedding is stored in a matrix W. Both paragraph and words vectors are used to predict the target word. While a paragraph vector only contributes to the prediction of words found in the paragraph only, the word vectors are shared across all paragraphs. The paragraph vector acts as the memory of the general context, of the paragraph, hence the architecture's name. In the DBOW architecture, a text window is sampled, then a random word from this window, and the model is trained to predict the missing word given the paragraph vector. Both architectures use the hierarchical softmax based on a binary Huffman tree.



(a) DM architecture



(b) DBOW architecture facility

Figure 5.9: Paragraph Vector architectures derived from [184]

Le and Mikolov found in [184] that the DM architecture is consistently better than DBOW for sentiment analysis and information retrieval tasks. However, following contradicting results found in the Literature [189], Lau and Baldwin further compare both architecture and reach the conclusion that DBOW generally outperform the DM architecture. Thus, the DBOW architecture is adopted for this study.

Past relevant applications

Similar applications are found in the legal [190, 191], patent [192], cybersecurity [193], and even Persian poetry [194] fields. The approach is akin to the one suggested below, to discover similar documents, the representation vectors are learned with a doc2vec model and then compared with a cosine similarity.

No doc2vec model trained on a corpus of space systems documents was encountered in the Literature, this work is thus considered as the first application of document embedding to space systems.

5.3.2 Approach

The goal is to quantify the similarity between the feasibility reports provided by the ESA CDF team. These reports are one of the main text sources described in Section 3.2.1. A doc2vec model is trained on a corpus of 27,016 ECSS requirements presented in Section 3.2.2. The model is then used to embed the feasibility reports and chapters from the reports. The model is trained on this smaller corpus rather than the large, main, corpus to focus on highly technical knowledge, the requirements. Embedding the chapters instead of the full documents enable a deeper analysis of the heritage, comparing the architecture choice at subsystem level. The representation vectors of the documents at report or chapter-level are then compared through a cosine similarity measurement. The documents with the highest cosine similarity are the most similar. The approach is summarised in Figure 5.10.

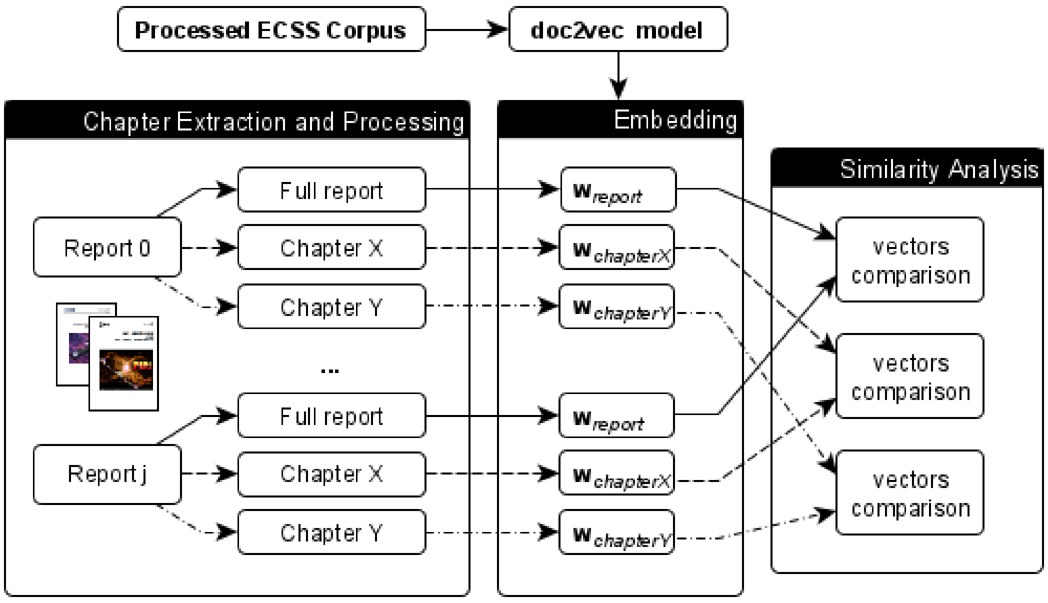


Figure 5.10: Methodology for Heritage Analysis with doc2vec

5.3.3 Methodology

Doc2vec model hyperparameters

The hyper-parameters, displayed in Table 5.14, are set accordingly to the suggestions of Lau and Baldwin [189] who empirically defined optimal doc2vec hyperparameters setting for general applications. Although Le and Mikolov initially reported that the DM architecture outperformed the DBOW architecture in [184], other research have reported conflicting results [189]. The model is trained with the open-source Gensim Python library [132] as the word2vec model.

Table 5.14: Hyper-parameters for doc2vec model training

Parameter	**Setting**	**Parameter Description**
Vector Size	300	Dimension of the representation vectors
Epochs	400	Number of training iterations
Mode	DBOW	DBOW or DM mode
Minimum Count	1	Minimum word frequency in corpus threshold
Window	15	Left/right context window size
Subsampling	10^5	Threshold to downsample high-frequency words
Negative Sampling	5	Number of negative word samples

The model is trained on the ECSS requirements introduced in Section 3.2.2. The requirements are processed with the domain-specific NLP pipeline presented in Section 3.2.3. The corpus is divided into a training set (80%) and a testing set (20%). Each requirement is considered a document. Following training, a 'sanity-check' reveals that the model would associate each document/requirement from the training set to itself with an accuracy of 0.99. Treating the testing set as unseen documents, the average cosine similarity of a document with itself is around 0.98. The output representation vectors are compared with the cosine similarity measure presented in Section 4.4.3.

Chapter Selection

To obtain the representation vectors of various report sections, the chapters must first be extracted. Although the reports are based on a similar template, they do not cover exactly the same topics or use the same titles for their chapters. Figure 5.11 displays the distribution of the top 15 chapters found in the table of contents of the 55 feasibility reports provided by ESA. The table of contents were extracted by identifying capital letter words with a regular expression (regex) pattern. As shown on Figure 5.11, the *Executive Summary*, *Introduction* and *Mission Analysis* sections are consistently found in the reports. Figure 5.11 however does not take into account the naming variations found in the reports. For instance, the *Mission Objectives* chapter is called *Study Objectives*, *Objectives* or *Study Objectives and approach* in a few reports. These variations prevent the automatic extraction of chapters and a human annotator is necessary to validate the chapter titles. The occurrence of key chapters such as *Systems* and *Mission Objectives* respectively increases from 75% to 96% and from 67% to 90% when taking into account all title variations.

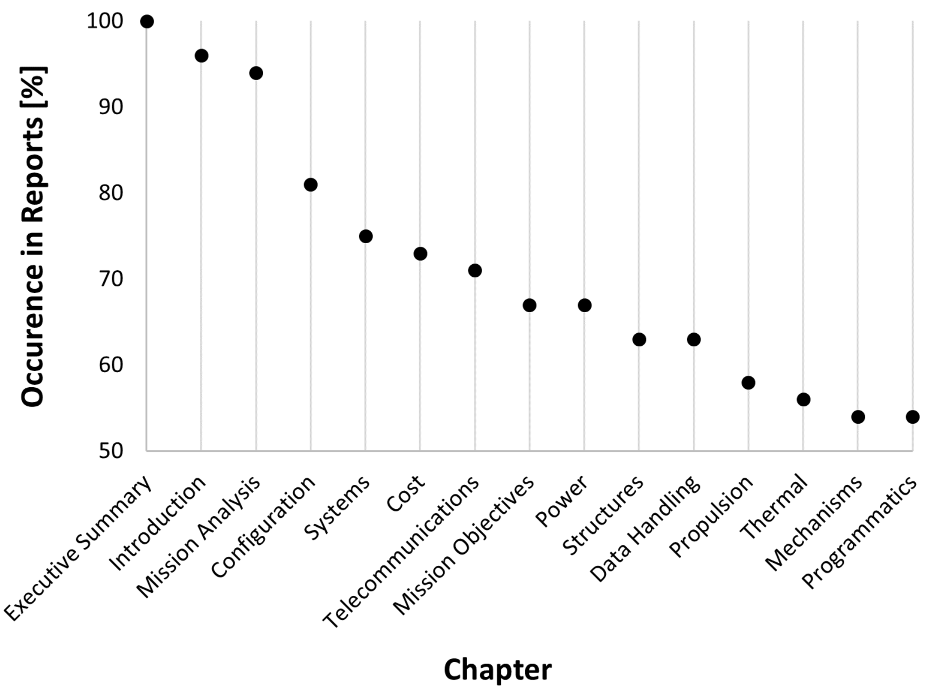


Figure 5.11: Occurrence of identical chapter titles found in 55 feasibility reports.

For this case study, the embedding will focus on three system-level chapters (*Executive Summary*, *Mission Objectives* and *Systems*), and on *Mission Analysis*. These chapters were chosen as they corresponds to key sections summarising the mission's architecture and design drivers. The orbit selection is also an essential criterion decided on at the early stages of the design process, and unlikely to change.

5.3.4 Results & Discussion

Due to the confidential nature of some CDF reports, this section will reference public documents, for instance, assessment study or definition study reports, instead of feasibility study reports. Confidential missions for which no information is publicly available will be anonymised and renamed as "Mission {A, B, ..., Z}".

First, each of the available 55 reports is mapped to a representation vector with the doc2vec model. The similarity of each vector with respect to the remaining 54 studies is computed with the cosine similarity. The results are displayed as an anonymous heatmap in Figure 5.12 where each line and column corresponds to a mission's representation vector. The results then focus on a case study with the ATHENA mission study.

ATHENA Case Study

ATHENA, standing for Advanced Telescope for High-ENergy Astrophysics, is an X-ray telescope mission selected in June 2014 as part of the ESA's Cosmic Vision 2015-25 programme. The latter is the current planning cycle for ESA's space science missions. The mission's science objectives are to map hot gas structures in the Universe and study their physical properties as well as search for supermassive black holes. The mission is due to be launched in the early 2030s and will be placed in an orbit around the second Lagrange point of the Sun-Earth system L2. The CDF team ran a study for the ATHENA mission in November 2014. This mission is chosen for the case study as (i) its CDF report, [195], is publicly available (without sensitive items such as cost data) [1], and (ii) the report contains information on previous similar mission that support this analysis.

[1]`https://www.cosmos.esa.int/web/athena/study-documents`

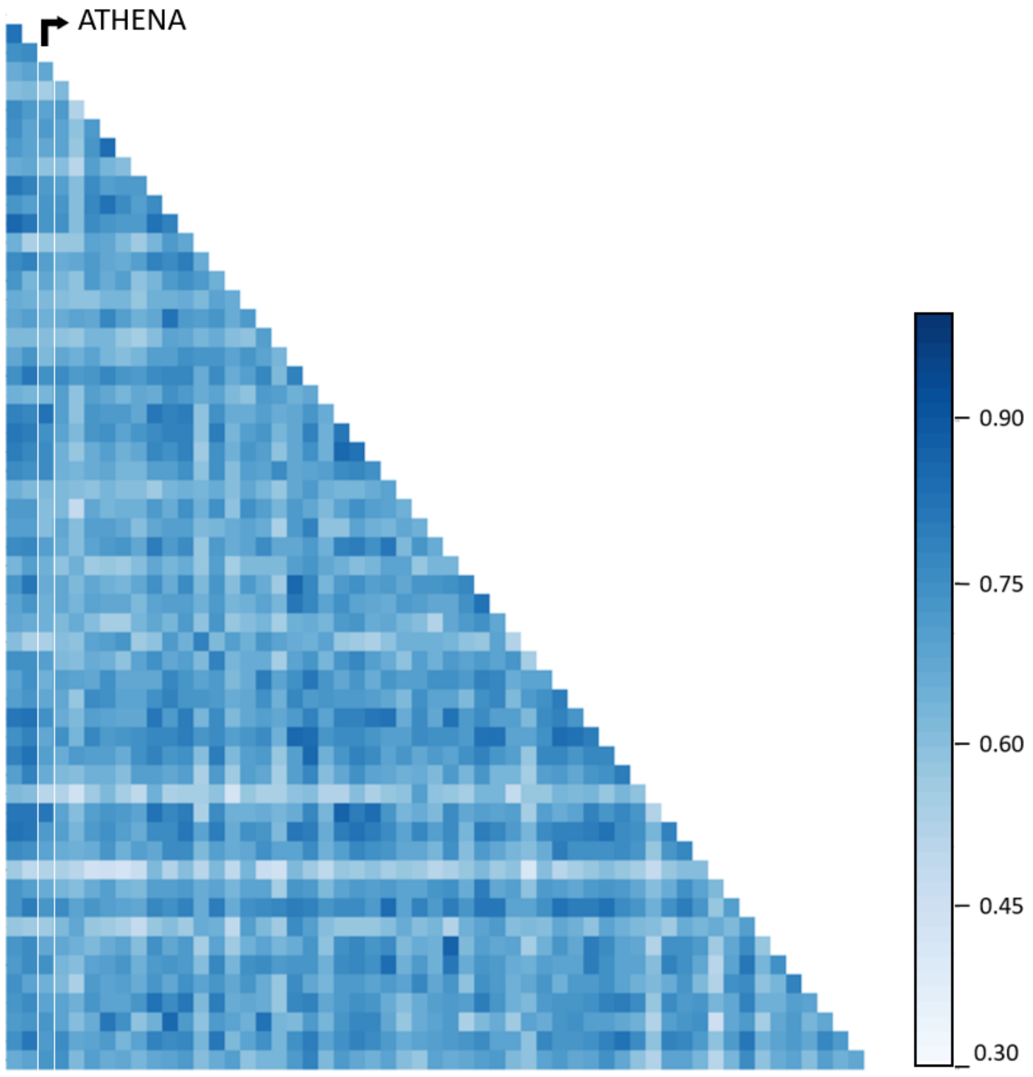


Figure 5.12: Anonymised heatmap of reports' similarities. Each column and each line correspond to a mission. The more intense the colour of their intersection, the higher their similarity.

Several past missions are mentioned in the ATHENA report. The XEUS [196], IXO [197] and ATHENA_L1 missions are highlighted as main predecessors as they also fly a X-ray telescope. The Herschel[2], Planck[3], Gaia[4], Euclid [198] and PLATO [199] missions have a similar L2 orbit. Finally, the spacecraft configuration design drivers

[2]`https://sci.esa.int/web/herschel`
[3]`https://www.cosmos.esa.int/web/planck`
[4]`https://sci.esa.int/web/gaia`

are similar to the XMM-Newton[5] and Chandra[6] missions. Of the mission mentioned, only the reports of the XEUS, IXO and PLATO missions are part of the corpus and thus taken into consideration for this heritage analysis.

Report-level embedding results

The full ATHENA report is mapped to a representation vector with the doc2vec model and compared to the other reports' representation vectors. The average cosine similarity is of 0.67 with a standard deviation of 0.07. The 10 reports most similar to the ATHENA mission according to the cosine similarity measurement are listed in Table 5.15. Most of the missions were studied after the ATHENA mission (post November 2014). The IXO and XEUS missions are respectively ranked at the 6h and 9th position. A human validation shows that the highest ranked missions either share a similar payloads (X-ray imaging and/or telescopes), orbit (L2 Lagrange Point) or programme (Cosmic Vision 2015-2025) with the ATHENA mission. The PLATO mission which has a similar orbit as ATHENA and was identified as a relevant past mission is however ranked 20th with a cosine similarity of 0.71.

Most of the missions highlighted by the doc2vec embedding and cosine similarity analysis carry a telescope, either for X-ray imaging such as *Mission A*, SMILE [200], IXO, XIPE [201] and XEUS or for other imaging in visible and/or infra-red such as ARIEL [202] and SPICA [203]. Apart from the SMILE, XIPE and Phobos-SR [204] missions which are respectively in Highly Elliptical Orbit, Low Earth Orbit or inter-planetary trajectories, all suggested missions are either placed in a Halo Lagrange point L2 orbit or at least start their trajectory from L2 as the M-Argo mission [205]. Finally, several missions are involved in the ESA Cosmic Vision 2015-2025 Programme, either as current (ARIEL, SMILE) or former candidates (SPICA, IXO, XIPE).

Although these initial results are encouraging as they allow to quickly identify potential similar missions, the report-level embedding approach still requires an expert's eye to understand how the missions are similar. The next step is thus to embed chapters

[5]`https://www.cosmos.esa.int/web/xmm-newton`

[6]`https://www.esa.int/Science_Exploration/Human_and_Robotic_Exploration/Chandra_X-Ray_Observatory`

Table 5.15: Comparison of reports vector representations obtained with the doc2vec model.

Rank	Mission	Cosine Similarity	Year	Manual Similarity Assessment		
				Payload	Orbit	Programme
1	*Mission A*	0.82	12/2014	X	X	-
2	ARIEL	0.81	2015	X	X	X
3	SMILE	0.76	2015	X	-	X
4	SPICA	0.76	2018	X	X	X
5	QPPF	0.75	2018	-	X	-
6	IXO	0.75	2008	X	X	X
7	MARGO	0.74	2017	-	X	-
8	XIPE	0.74	2015	X	-	X
9	XEUS	0.74	2004	X	X	-
10	Phobos_SR	0.74	06-2014	-	-	-

rather than full reports, to refine the automatic heritage analysis, by highlighting how the missions are similar.

Chapter-level embedding results

The *Executive Summary*, *Mission Objective*, *Systems*, and *Mission Analysis* chapters are extracted from each CDF reports, embedded with the same doc2vec model and compared with the cosine similarity analysis. The *Executive Summary* usually contains a complete description of the mission including background information, scientific justification, context, and systems overview. The *Mission Objective* chapter focuses on the mission's goal, while the *Systems* chapter summarises the system requirements and design drivers. Finally, the *Mission Analysis* chapter addresses the selected orbital parameters, station-keeping, and the orbit trade-off.

The top 10 missions with the highest cosine similarities for each chapter of interest are highlighted in Figure 5.13. The missions which are found similar to ATHENA after human validation are highlighted in green, the others in red. The analysis at chapter level provides a different perspective on heritage analysis. The PLATO mission, a known similar mission, appears in the top 10 for the *Mission Objectives* and *Mission Analysis*. Other candidates of the Cosmic Vision 2015-2025 such as Laplace [206], LISA [207], CHEOPS [208], THOR [209] and LOFT [210] appear in this comparison. The results are however mixed, with accuracy as high as 80% for the *Mission Objective* chapter, and as low as 50% for the *Systems* chapter. For instance for the *Mission Analysis* chapter, *Mission A* and SPICA do have similar orbits and operations as ATHENA's. There is a noticeable step in the cosine similarity values between the 2nd and 3rd suggested missions. Then the 3rd, 4th, 5th and 6th missions respectively have Earth-Moon, interplanetary and Low Earth Orbit trajectories which are very different from ATHENA's orbit.

5.3.5 Conclusions

The results of this case study are encouraging. The approach based on the doc2vec model and cosine similarity allow the quick retrieval of potential relevant past missions. It is recommended to learn the representation of chapters rather than full reports to provide a detailed heritage analysis.

The method accuracy however varies and requires human validation. To improve the heritage analysis outputs, the metadata parameters used by the CDF team could be combined with the doc2vec approach. Relying on contextualised embedding such as BERT rather than doc2vec could increase the quality of the vectors. To enhance the quality of the representation vectors, the authors of the reports could also be compared to remove potential author bias. This bias could be caused by the author's writing style. Finally, to expand the range of past missions assessed, contents from online missions databases such as ESA EoPortal Directory[7] could be integrated to the comparison.

[7] `https://directory.eoportal.org/web/eoportal/satellite-missions/`

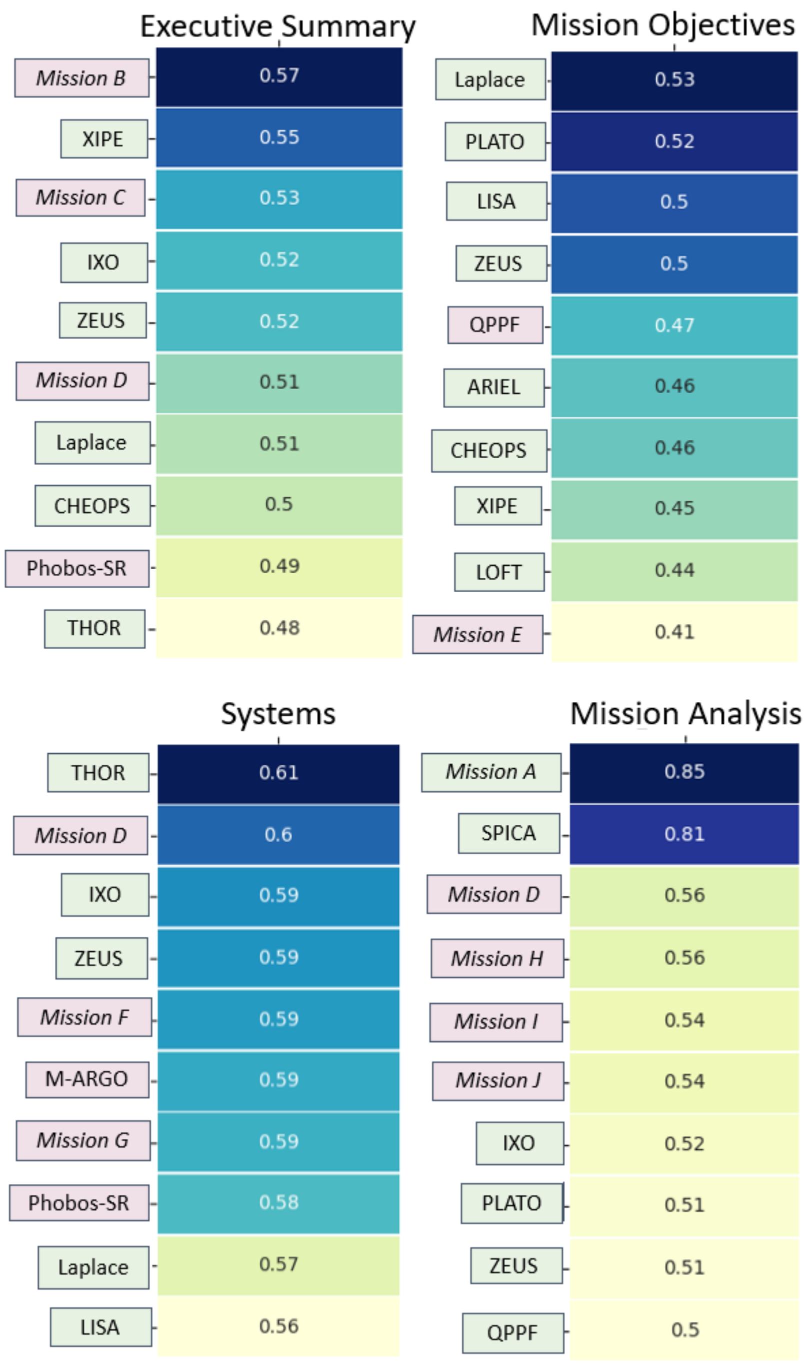


Figure 5.13: Outputs of the chapters comparison for the ATHENA case study. The highest the cosine similarity value, the more similar the missions' chapter are.

5.4 Chapter Summary

Two new domain-specific models, spaceLDA and a space doc2vec, were developed in this chapter. The domain adaptation of these text mining methods demonstrated the potential for these methods to enhance knowledge management and reuse at the early stages of space mission design by automating tasks usually done manually by engineers. These stand-alone applications can contribute to increase the range of queries that could be handled by the DEA as suggested in Figure 1.1. The SpaceTransformers models presented in Section 4.5 can be fine-tuned to additional downstream applications such as Question-Answering or Similarity Analysis, therefore also contributing to task automation.

Chapter 6

From Engineering Models to Knowledge Graph

6.1 Chapter Overview

The technology trade-off was initiated in Berquand et al., "Artificial Intelligence for the Early Design Phases of Space Missions" at the 2019 IEEE Aerospace conference [9]. The approach, methodology, and mass budget application are derived from Berquand, A. & Riccardi, A., "From Engineering Models to Knowledge Graph: Delivering New Insights Into Models" presented at the SECESA 2020 conference [12]. The background, methodology, and heritage application were refined in the frame of an OSIP study "System Engineering Models Meet Knowledge Graph" (ESA Contract No. 4000133311/ 20/NL/GLC) led by Valera, S. (ESA) and Riccardi, A. (University of Strathclyde).

Engineering Models (EMs) are the second main outputs of Concurrent Engineering studies. This type of data is considered as semi-structured rather than structured data as it contains information, for instance requirements, in natural language. EMs contain key information on the spacecraft design, yet these models are today not easily reused, queried, let alone compared. On the other hand, a new type of data structure called Knowledge Graph (KG) is getting an increasing momentum in both the academic and industrial fields. KGs are particularly useful as they can handle data diversity, from high-quality complete data to sparse and incomplete data, with high scalability and flexibility. Furthermore, KGs have reasoning and inference capabilities through their schema structuring the graph information. They can thus infer hidden knowledge from explicit facts. To summarise, KGs could help engineers overcome data silos, interconnect information, provide a big-picture perspective, and infer new knowledge that would have remained hidden otherwise. KG is also the suggested database format for the DEA as seen in Figure 1.1.

This chapter explores how the migration of stand-alone Engineering Models (EMs) to a common KG could enhance the data linkage, reusability and interpretability of these models, further contributing to knowledge management and reuse at the early stages of space mission design. The contributions of this chapter are summarised as follow:

1. The potential for the integration of a KG based on EMs into the Concurrent Engineering process is reflected upon.
2. A trade-off of available off-the-shelf KG tools is provided.
3. Novel mapping rules and migration pipelines from UML to TypeQL are defined.
4. The relevance of a KG to infer hidden knowledge from the EMs is demonstrated through an automatic mass budget generation application.
5. The potential for a KG combined with a NLP layer is demonstrated through a heritage analysis application.

Section 6.2 investigates how enhanced access to Engineering Models content could contribute to the current Concurrent Engineering process. Section 6.3 introduces UML concepts and examines different database types, with an emphasis on SQL and graph databases. A trade-off of available Knowledge Graph tools leads to the selection of the Vaticle TypeDB database. Section 6.4 summarises the approach followed to migrate the Engineering Models to a Knowledge Graph. The mapping rules, from UML to TypeDB, required for the migration are detailed in Section 6.5. Two case studies identified in Section 6.2 are implemented in 6.6. Finally, Section 6.7 discusses the results. The source code for [12] is available at `github.com/strath-ace/smart-nlp`.

6.2 Motivation

As described in Section 2.1.2, and summarised in Figure 2.7, a feasibility study following the Concurrent Engineering approach is usually divided in three parts: the preparation, the study itself, and the post-study. The design process is iterative, with several consecutive design iterations and various design options for each iteration. A feasibility study has several outputs: a study report, an Engineering Model (EM), and, sometimes, a record of lessons learned. Chapters 4 and 5 focused on the first type of output, unstructured data, while this chapter addresses the reuse of the EMs. Based on the expert survey findings presented in Section 2.2, Figure 6.1 suggests how a database populated with previous studies' Engineering Models could support the current design process. The database queries focus on the preparation and the study phase, as the post-study action would consist in migrating the latest EM to the KG. The queries are focused on both inter-model and inter-iterations comparisons. To successfully address the queries, the database is enriched with a NLP layer.

The suggested queries of interest for the preparation phase are the following:

1. **Heritage Analysis:** Based on the metadata and requirements contained in the EMs, similar missions can be clustered with an approach akin to the one presented in Section 5.3 relying on doc2vec.
2. **Estimation of initial design parameters:** The quality of the input parameters is key to enabling a faster design convergence. A common database of EMs makes its possible to compare past previous architectures of similar missions, and helps kick-start the design.

The suggested queries of interest for the study phase are the following:

1. **Budget Evolution:** The systems engineers manually keep track of the mass and power budgets throughout the design study. These could be inferred automatically from the EMs. New insights could also be derived from the evolution of mass and power budgets accross the different iterations of an EM.
2. **Metadata Evolution:** The metadata assigned to each iteration can be extracted and compared, providing further insights on the design evolution.

3. **Architecture Evolution:** This type of query provides insights at subsystem level, by comparing the equipment assigned to each subsystem per iteration. For instance, the evolution of heater or radiator surfaces between consecutive iterations can be inferred.

4. **Comparison with past architectures:** The subsystem architecture can be compared to past missions' architectures. For instance, by inferring the average area of solar panels used in past similar missions.

Current tool for the modelling of the EMs, the OCDT, CDP4 and COMET do not allow to compare the EMs of different missions nor to compare the different iterations of a same EM. Although the most recent tool, COMET, offers reports generations templates, none of the tools integrate text mining or NLP capacities, key enablers of some above queries.

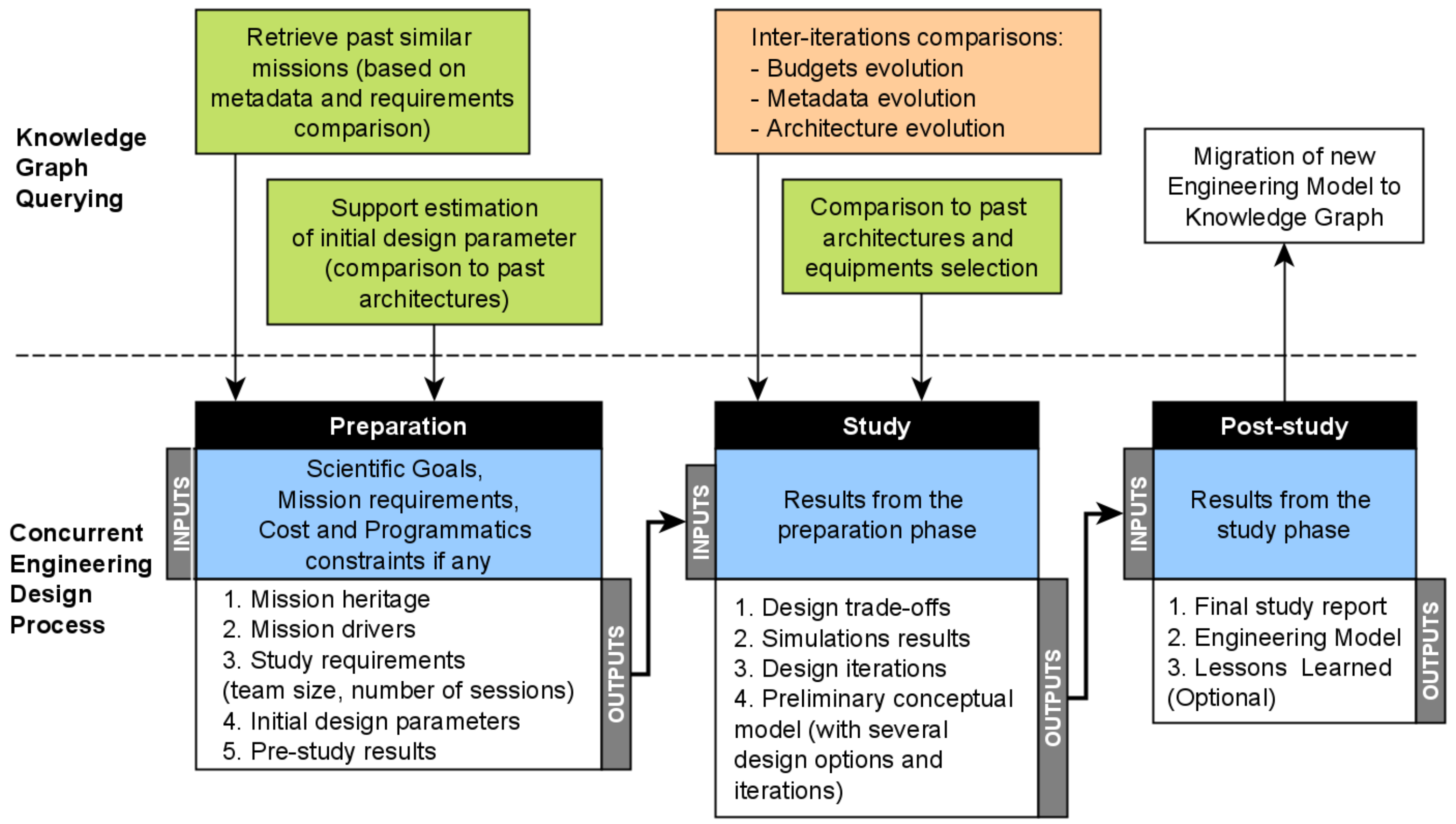

Knowledge Graph Querying
Retrieve past similar missions (based on metadata and requirements comparison)
Support estimation of initial design parameter (comparison to past architectures)
Inter-iterations comparisons:
- Budgets evolution
- Metadata evolution
- Architecture evolution
Comparison to past architectures and equipments selection
Migration of new Engineering Model to Knowledge Graph
Concurrent Engineering Design Process
Preparation
INPUTS
Scientific Goals, Mission requirements, Cost and Programmatics constraints if any
1. Mission heritage
2. Mission drivers
3. Study requirements (team size, number of sessions)
4. Initial design parameters
5. Pre-study results
OUTPUTS
Study
INPUTS
Results from the preparation phase
1. Design trade-offs
2. Simulations results
3. Design iterations
4. Preliminary conceptual model (with several design options and iterations)
OUTPUTS
Post-study
INPUTS
Results from the study phase
1. Final study report
2. Engineering Model
3. Lessons Learned (Optional)
OUTPUTS

Figure 6.1: Integration of the Knowledge Graph querying in the Concurrent Engineering design process. The green boxes represent inter-model queries while the orange box represents the inter-iterations queries.

6.3 Background on Information Modelling & Knowledge Graph

Information modelling is the process of creating a data model for information to be stored in a database. The data model represents the data objects and their associations through relationships and rules. As defined by Valera in [211], a model is a combination of (i) a schema, structuring the regulations for a universe of discourse, and (ii) a population, the data captured. A conceptual schema declares an ontology of the concepts, focusing on the semantics and all concepts relevant to a universe of discourse.

Figure 6.2 illustrates how a domain-specific model is built from a generic model. Each model, generic and domain-specific has its own schema and population. The generic model corresponds to one of the many languages used to specify a domain specific model, for instance, UML. The domain-specific model represents a universe of discourse. The population of the generic model corresponds to the schema of the domain-specific model. For instance, *Spacecraft* would be a level 2 entity, *Sentinel-1* (a satellite) the level 1, the population of the *Spacecraft* entity; and the level 3 would be a UML class. In the frame of the ECSS-based EMs, the generic model (level 3) consists in the building blocks of the Unified Modelling Language (UML), populated with the objects from the ECSS-E-TM-10-25A class diagram (level 2) such as *Element*, *Iteration*. The domain-specific model structured with the latter is then populated with the content of the EMs (level 1), for instance, *Spacecraft* or *Iteration1*. This section investigates current databases and tool options to structure and host the migrated EMs based on an UML class diagram.

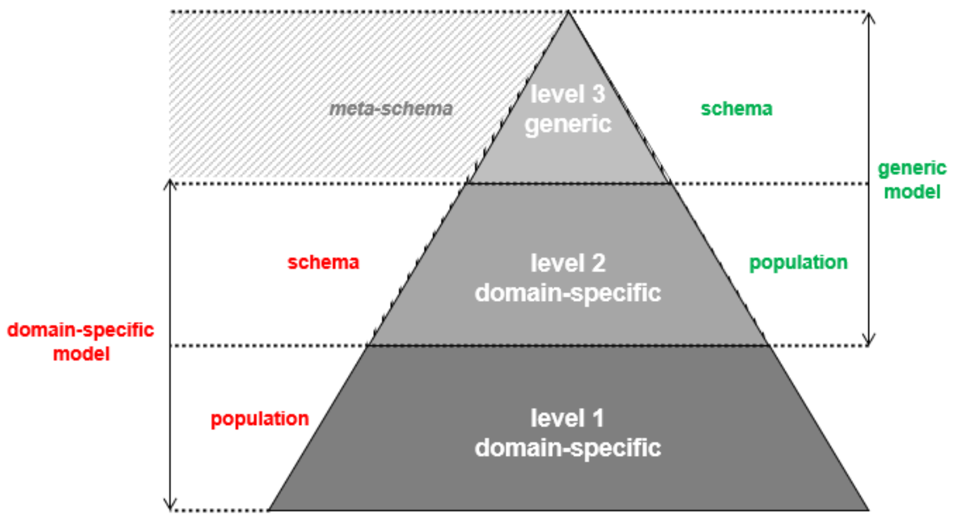


Figure 6.2: Generic vs Domain-Specific Modelling. Courtesy of S. Valera (ESA, TEC-SWT).

First, some background on the UML language and its building blocks, the starting point of the migration, is provided. Common types of database are then described and compared, with an emphasis on graph databases. The Knowledge Graph technology is eventually chosen for the migration, as it integrates semantics, enabling reasoning and the inference of hidden knowledge. The Vaticle TypeDB tool is selected following a trade-off of available KG tools. The level 3 of the pyramid will thus require a mapping from UML to TypeQL concepts, the level 2 a migration of the UML class diagram to a TypeDB schema, and the level 1 will contain the Engineering Models.

6.3.1 Unified Modeling Language

Unified Modeling Language (UML) is a standardised Object-Oriented (OO) modelling language for specifying, visualising, constructing and documenting software systems. As described in [212], UML was invented by Booch, Jacobson, and Rumbaugh in the late 90s, answering a call issued by the Object Management Group (OMG) to develop the specifications for a uniform modelling language. It was adopted as a standard by OMG in 1997. A UML model is represented schematically by diagrams. There are two main categories of diagrams: Structure and Behaviour [212]. Structure diagrams provides a static view of the model architecture. The class diagram is an example of such diagram, modelling the architecture of a system. Behaviour diagrams emphasise the dynamic behaviour of the system, highlighting the exchanges between the objects. A sequence diagram is an example of behavioural diagram representing the sequence of actions followed by various objects to complete a task.

The conceptual model of the EMs to be migrated is based on a class diagram. As described on the official UML webpage[1], the building blocks of a class diagram are:

- **Class:** A class describes a set of objects sharing the same features, constraints and semantics. A class is a type of classifier, an abstract metaclass describing set of instances with common features. Classes can be abstract or concrete.
- **Interface:** An interface is another type of classifier, declaring a set of features and obligations to be implemented not by the interface itself but by another classifier. The interface is not instantiable but is fulfilled by an instance of an instantiable classifier.
- **Features:** The features represents a structural or behavioural characteristics of a classifier. For instance, a property is a structural feature and an operation is a behavioural feature.
- **Property:** The property is a structural feature representing an attribute or the end of an association. A property can be derived. A derived property is produced or computed from other information. For instance, an "*age*" property would be

[1]`https://www.uml.org/`

derived from a "*birthday date*". The attributes of a class are instances of property owned by the class. Attributes can be inherited.

- **Operation:** Operations are a behavioural feature that may be owned by a class, an interface or a data type. A method is the implementation of an operation. It specifies the procedure associated to an operation.
- **Relationship:** A relationship links one or more related classes. The relationship can thus be binary or n-ary. The different types of UML relationships are underlined in Figure 6.3, and illustrated in the example Figure 6.4.

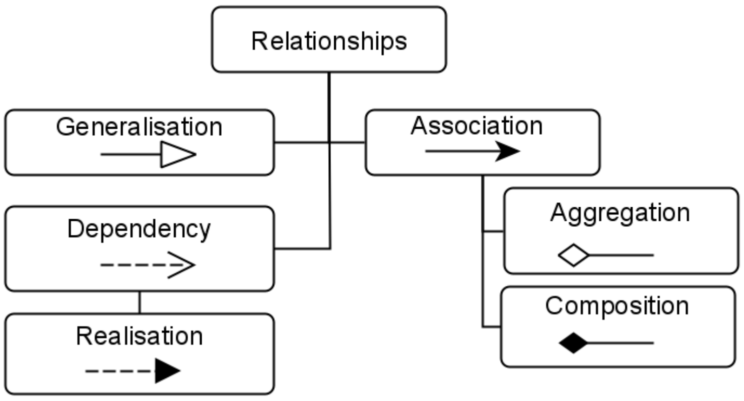


Figure 6.3: Overview of the main UML relationships

Figure 6.4 displays a basic example of an UML class diagram. Multiplicity is specified to indicate n-ary relationships by a 0..∗ or a 1..∗, respectively meaning from zero or one to infinity. An association relationship connects, for instance, the *EngineeringModel* to at least one *Participant* class. The aggregation is a sub-class of the association, representing a "*part of*" relationship. The *Person* class contains at least one *EmailAddress.* In the case of a composition relationship, such as the one between the *Iteration* and the *Option* classes, the existence of a class depends on the other. For instance if the *Iteration* disappears so does the *Option.* The Generalisation or Inheritance relationship links a general classifier to a more specific classifier. The *Participant* class is thus a sub-type of a *Person* class and it inherits the features of its parent class. A dependency relationship is a directed relationship indicating that one or a set of UML elements depend on another model element for specification or implementation. For instance, the *domainOfExpertise* attribute of the *Participant* class depends on the values dictated by the *Domain* enumeration. Finally, the realisation is a special type of dependency, implementing a client-supplier relation, where an object realise a behaviour dictated by another object. For instance, the *EngineeringModel* class implements the *EngineeringModelSetUp*.

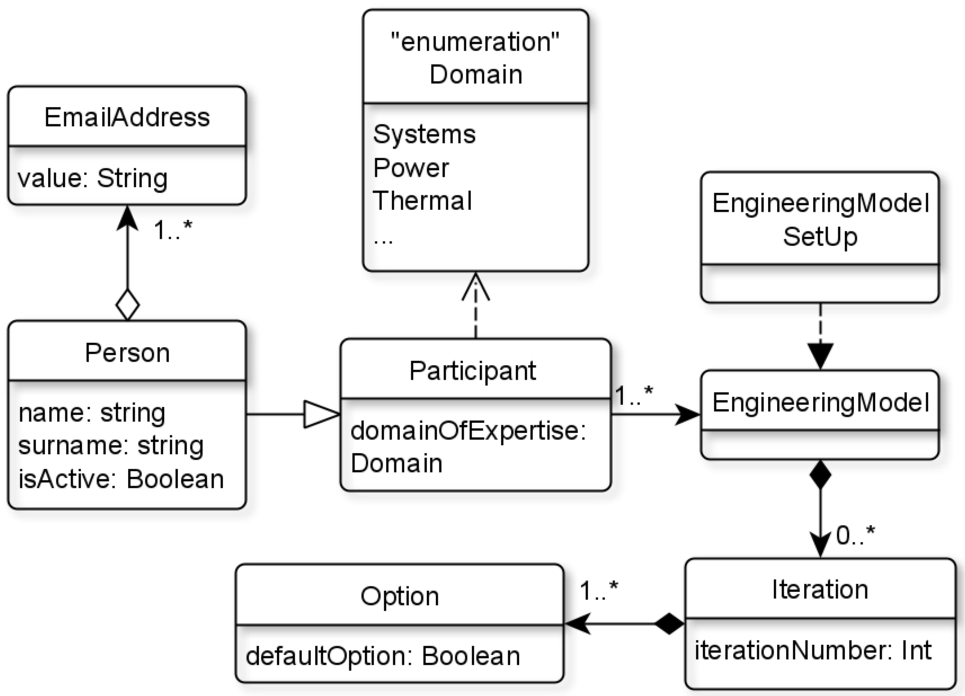


Figure 6.4: Example of an UML class diagram

The conceptual model of a UML diagram aims to be a software blueprint, not to model a universe of discourse. As underlined by Mkhinini et al. in [213], although the field of ontologies and UML share similarities, the limited semantic expressivity of UML leads to significant loss of information. Romero et al. concur by saying that UML defines semantics imprecisely, and thus extend their UML model with additional semantics [214]. The migrated conceptual model built from a UML class diagram will thus inherits the UML semantic limitations.

6.3.2 Relational vs Graph Databases

Graph databases have been increasingly popular in the past years as shown in Figure 6.5. This section provides a thorough comparison between relational and non relational databases, with an emphasis on graph database. This comparison eventually justifies the choice of graph databases for this chapter.

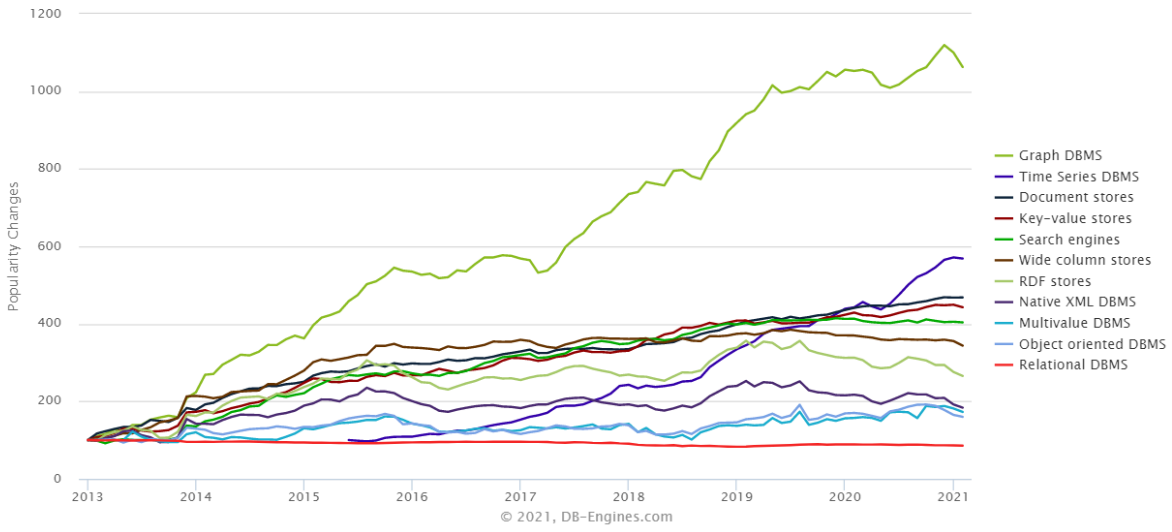


Figure 6.5: Historical trend of database systems' popularity. The popularity changes are computed monthly based on the averaged ranking of the best three systems per category. Source: DB-Engines.com

Relational Databases

Relational Modelling was first introduced by E.F.Codd, a researcher at IBM, in [215]. Relational models target highly structured data, organising data in tables and rows. The common querying language of relational databases is SQL, Structured Querying Language. The data contained in the tables are linked by primary and foreign key pairs, a key being a unique identifier. Relational databases implement Relational Modelling, structuring data into tables, which can be joined to extract insights. From their first use in the 70s, SQL databases are still popular today.

Non-Relational Databases

Non-Relational Databases, also called NoSQL, appeared in the early 2000s, providing new ways of storing, searching and retrieving data other than in tables with relationships. With the rise of Big Data, large amount of unstructured, semi-structured and structured data required storage solutions. As underlined by Li and Manoharan [216], NoSQL databases appeared as an ideal solution as they claimed to be more flexible (schema-free) and faster than SQL, and they could handle heterogeneous data.

Graph Databases

A graph database is a type of NoSQL database based on Entity-Relationship (ER) modelling. The latter models were first suggested by Peter Chen in 1976 [217]. ER diagrams rely on three building blocks: entities (i.e., real word objects), attributes and relationships. With their simple construct, these models are extremely intuitive. As underlined by Harrison [218] and Robinson et al. [219], graph databases are purpose-built to store and navigate relationships. Connectivity is therefore a key feature of graphs, and the relationship between data is seen as equally important as the data itself. Graphs are also flexible and naturally additive. As described by Robinson et al. in [219], there are three main types of graph databases:

- **RDF Graph:** Resource Description Framework (RDF) graphs are based on RDF triples. RDF is a standard model for data interchange proposed for the Semantic Web. As described in [220], RDF was first defined as a standard in 1999 by the World Wide Web Consortium (W3C), to capture the metadata of web resources and interlink resources in a formal, machine-understandable language. The Web Ontology Language (OWL), the ontology language recommended by W3C, is partly based on RDF. An RDF graph is a directed graph, with labelled nodes and directed arcs. Each arc is a binary relationship between two nodes, or resources. Nodes and relationships are identified by a Uniform Resource Identifier (URI). A RDF statement is called a triple, with an object-predicate-object structure as shown on Figure 6.6. RDF graphs integrate semantics unlike property graphs without schema.

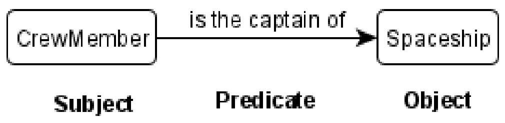


Figure 6.6: Example of a RDF triple

- **(Labelled) Property Graph:** In a property graph, relationships are binary and directed, they have a start and an end node. Both edges and nodes can have properties, which as underlined in [218], allow the property graph to provide a richer model than RDF. Property Graphs do not require a schema, therefore might not bear semantics. The property graph is the basis of Neo4j[2], one of the most popular graph database.
- **Hypergraph:** Hypergraphs are a subcategory of property graphs allowing n-ary relationships. This type of relationship can connect any numbers of nodes, and is therefore useful for models requiring many-to-many relationships [219].

Comparison of SQL & Graph Databases

Based on the Literature Review findings, the relational and graph databases are compared in Table 6.1. The trade-off takes into consideration the following parameters:

1. **Data Types:** This parameter indicates the types of data assimilated in the database.
2. **Structure:** This parameter refers to the modelling blocks used by the database.
3. **Compliance to standards:** This parameter underlines whether or not the database type follows standards.
4. **Knowledge Representation:** This parameter underlines which data elements the database focuses on.
5. **Reasoning Engine Performance and Query Response Time:** The query latency is key for query performances and deep analytic.

[2]`https://neo4j.com/`

6. **Use Cases:** This last parameter identifies the use cases best adapted to each database type.

Table 6.1: Databases Type Trade off

	Relational Database	**Graph Database**
Data Types	highly structured data	heteregenous data
Structure	Tables, relationships are established by primary and foreign keys.	Graph: nodes connected by edges
Compliance to standards	ISO standard	RDF: W3C standard, Property/Hypergraph: No
Knowledge Representation	Focus on data representation	Focus on relationship representation
Query Response Time	High latency at large scale. Query latency increase with amount of data stored due to "join" queries	High performance for complex deep analytic, and for complex transactions
Use Cases	Transaction-focused use cases: online transactions, accounting	Relationship-heavy use cases, when seeking hidden connections: fraud detection, recommender systems, social networks

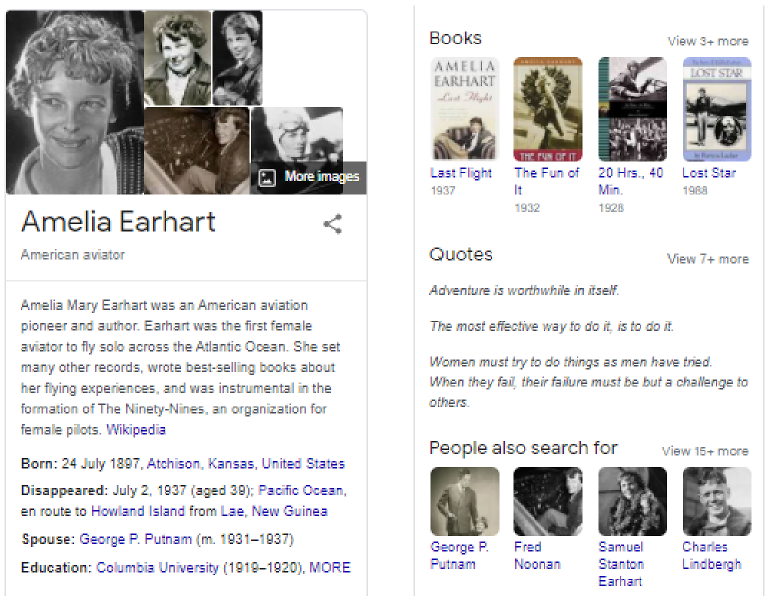
The trade-off outputs reinforce the choice of graph databases over relational databases for the EMs migration. Graph databases are purpose-built to store and navigate relationships. The query performances of relational databases is also inferior to graph databases due to the complexity of compute-heavy joins queries. Graph databases are therefore more adapted to use cases involving highly interconnected data. As the purpose of the migration is to query the EMs rather than modify or add content to them, a graph database appears as the most suitable option.

6.3.3 Knowledge Graphs

This section first explores the origins of Knowledge Graphs and attempts to define them. The evolution of KGs, from targeting web resources to becoming Enterprise KGs adapted to entreprise information management, is briefly discussed. Finally, available modelling tools are compared to select the tool best adapted for the migration of EMs.

Origins

Knowledge Graphs became popular after Google released its graph in 2012. According to a former Google Senior Vice President, Amit Singhal, Google had produced an *intelligent model that understands real-world entities and their relationships to one another: things, not strings*" [221], thus a model capable of far more than mere keyword matching. A Google search result now comes along with a knowledge summary, containing key facts and suggesting related information. Figure 6.7 displays the knowledge summary for an "*Amelia Earhart*" query. Along with an overview of her accomplishments, the knowledge summary links to books, quotes and even related personalities.



Amelia Earhart

American aviator

Amelia Mary Earhart was an American aviation pioneer and author. Earhart was the first female aviator to fly solo across the Atlantic Ocean. She set many other records, wrote best-selling books about her flying experiences, and was instrumental in the formation of The Ninety-Nines, an organization for female pilots. Wikipedia

Born: 24 July 1897, Atchison, Kansas, United States

Disappeared: July 2, 1937 (aged 39); Pacific Ocean, en route to Howland Island from Lae, New Guinea

Spouse: George P. Putnam (m. 1931–1937)

Education: Columbia University (1919–1920), MORE

Books — View 3+ more

Last Flight 1937

The Fun of It 1932

20 Hrs., 40 Min. 1928

Lost Star 1988

Quotes — View 7+ more

Adventure is worthwhile in itself.

The most effective way to do it, is to do it.

Women must try to do things as men have tried. When they fail, their failure must be but a challenge to others.

People also search for — View 15+ more

George P. Putnam

Fred Noonan

Samuel Stanton Earhart

Charles Lindbergh

Figure 6.7: Google Knowledge Summary obtained for an "*Amelia Earhart*" query

Knowledge Graphs were however not invented by Google. As underlined by Fensel et al. in [222], KGs were originally based on Semantic Networks, networks of linked concepts. These were first mentioned in the 1960s, as a knowledge representation framework. The difference between KGs and Semantic Networks became blurry when, in the late 1980s, two Dutch universities initiated a "Knowledge Graph" project, de facto, a Semantic Network, restricting edges to a set of potential relations. Fensel et al. do underline that these earlier concepts were narrow compared to the millions of facts now covered by KGs.

Definition

Several definitions for KG are found across the Literature. Fensel et al. compile various survey's outputs in [222] to propose the following conceptual definition: "*Knowledge Graphs are very large semantic nets that integrate various and heterogeneous information sources to represent knowledge about certain domains of discourse*". Ehrlinger and Wöß then propose the following: "*A knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge*" in [223]. Finally, Gomez-Perez et al., in [224], define a KG as "*a set of interconnected typed entities and their attributes*" having "*an ontology as its schema defining the vocabulary used in the knowledge graph*".

All definitions encountered in the Literature seem to revolve around two key elements: (i) a KG is structured as a graph, using edges (i.e., relations) to interconnect nodes (i.e., entities, either real world objects or abstract concepts), (ii) a KG bears formal semantics defined by an ontology.

The key to differentiate a KG from a plain graph database is thus in the semantics. Ontologies are used to describe a universe of discourse. An ontology can therefore act as the schema of a KG, defining its entities, attributes and relationships. Semantics furthermore enable the inference of new facts.

From Knowledge Graph to Enterprise Knowledge Graphs

The success of the Google's KG set a new trend in information management. While the Google graph initially focused on web searches and data, companies quickly understood that this technology could be adapted to enterprise information management [225]. The GAFA tech giants (Google, Amazon, Facebook, and Apple) have all adopted KGs to integrate and make sense of their diverse data at a large scale [222, 226]. These graphs are usually called Enterprise Knowledge Graphs (EKG), as they are internal to the company and retains proprietary information. EKGs are the key to breaking data silos, connecting data to provide a big picture view and inferring new knowledge that would have remained hidden otherwise. To cite a few examples among well known platforms:

- To connect members to their companies and skills, suggest new job opportunities, and perform business and consumer analytics, LinkedIn implemented it own KG. In [227], the LinkedIn KG is said to connect, among other entities, 450M members, 190M job listing and 9M companies.
- Airbnb, hosting over 5 million rental homes is targeting to become an end-to-end travel platform. Therefore, to offer more than rental house listings, Airbnb has developed its own taxonomy and KG to enhance the traveling experience of its users. By inferring experiences (i.e., activities, events) and location perks (i.e., landmarks proximity, restaurants options) with the available houses, the platform can now deliver travel insights to its users [228] .
- To encode semantics knowledge about the products sold on its platform, and understand the context of a buyers' query, eBay is developing a Product KG called Beam [229].

In the space field, KGs have also gained some momentum. NASA collaborated with Stardog[3] to develop a KG for Product Lifecycle Management (PLM) and Product Data Management (PDM), connecting data silos to facilitate the engineering of complex systems [230, 231]. Another NASA team, [232], used MongoDB[4] and Neo4j[5] to migrate a Lessons Learned (LL) database to a KG, identifying patterns and leading to a quicker recall of relevant LLs during a search. Finally, GraphAware and Cambridge Intelligence recently produced a proof-of-concept for ESA, monitoring the satellite technology market [233, 234]. With the help of domain experts to annotate entities, GraphAware mapped the current satellite ecosystem into a Neo4j graph. To support the visualisation, the data was exported into a KG with the KeyLines product from Cambridge Intelligence; an example is displayed in Figure 6.8.

The increasing demand for EKGs has resulted in the emergence of several consulting companies such as Stardog and GraphAware (Neo4j consultancy) offering EKGs packages. Off-the shelf tools to develop Knowledge Graphs are also available.

[3]`https://www.stardog.com/`
[4]`https://www.mongodb.com/`
[5]`https://neo4j.com/`

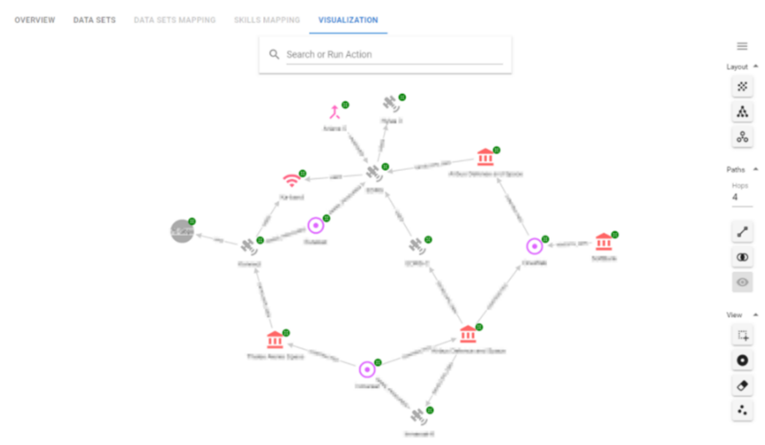


Figure 6.8: Visualisation of the space ecosystem provided by GraphAware in [233], blurred by GraphAware.

Tools Trade-Off

Nine KG modelling tools: Vaticle TypeDB (formely known as Grakn), Neo4j, GraphDB, StarDog, Apache Jena, AllegroGraph, JanusGraph (formely known as Titan), TigerGraph, and AnzoGraph DB are compared. The nine tools were chosen based on their popularity and the fact that they can be schema-based. As previously defined, a graph database without semantics does not qualify as a Knowledge Graph. Therefore, popular graph databases such as Arango-DB, Microsoft Azure Cosmos DB, OrientDB, Amazon Neptune, DGraph, or FaunaDB are not included in the comparison as they are schema free. The goal of the trade-off is to identify the most appropriate tool for the migration of the EMs.

Semantic Web technologies require the user to learn several languages: RDF, RDFS, OWL, SPARQL for querying and SHACL for inference. This approach results in a high barrier to entry that was not adapted to the study timeline. Tools independent from the W3C, like TypeDB, reduce the complexity linked to Semantic Web Standards, while maintaining a high degree of expressivity [235]. With TypeDB, the ontology, query and reasoning languages are all written with one language: TypeQL. This language requires much less complexity to model and query highly interconnected data than SQL as demonstrated in [236] where 151-line SQL query is down-scaled to 4 lines in

TypeQL. Thus, tools based on RDF graphs are considered in the below trade-off, but will be penalised.

To compare the tools, the following parameters are used:

1. **Portability:** The tool will be considered as portable if it is compatible with the common operative systems, e.g., Linux, macOS and Windows.
2. **Learning Curve:** Learning curve for developers and end-users with no background knowledge in any language/modelling framework. Learning curve for graphs based on RDF triples have been set to "Long" due to the multiple technologies a user needs to master for schema definition, population, query and reasoning (RDF, RDFS, OWL, SPARQL).
3. **Scalability**: This parameter identifies whether the tool can grow and manage an increased demand, or has any limits in its number of nodes or relationships.
4. **Inference Engine:** This parameter assesses whether a reasoner is integrated to the tool or if it needs to be built on top. An integrated reasoner saves on time and system complexity.
5. **Visualisation:** This parameter judges if a visualisation interface is integrated into the tool, again saving on time and system complexity.
6. **Support**: The tool support is considered strong when, in addition of documentation and tutorials, there is an active and open exchange between the developers and the users, for instance through a forum or a social media channel. A dynamic community suggests interest in the tool and a potential long term viability. In addition, it is useful when developing to have an easy access to previous issues and solutions. The tool support will be considered average when documentation and tutorials are provided, but there is no open platform for the community to exchange. Without documentations or tutorials, the tool support will be considered as weak.
7. **Licence**: The tool could either be open-source or available through a commercial licence. Open-source is the preferred option.

The tools trade-off is displayed in Table 6.2. Additional table lines (i.e., generic modelling language, developer, initial release, query language, and Python support) provide background information on each tool. A Python API is here considered as an asset based on subjective coding preferences.

Table 6.2: Knowledge Graph Modelling Tools Trade-Off

	AllegroGraph	**AnzoGraph**	**Apache Jena**	**TypeDB**	**GraphDB**	**JanusGraph**	**Neo4j**	**Stardog**	**TigerGraph**
Generic modelling language	RDF Graph	RDF Graph	RDF Graph	Hypergraph	RDF Graph	Graph	Property Graph	RDF Graph	Parallel Graph
Developer	Franz Inc.	Cambridge Semantics	Apache Software Foundation	Vaticle	Ontotext	Linux Foundation	Neo4j Inc	Stardog-Union	TigerGraph Inc.
Initial Release	2004	2018	2000	2016	2000	2017	2007	2010	2017
Query Language	SPARQL	SPARQL*	SPARQL	TypeQL	SPARQL	Gremlin	Cypher	SPARQL	GSQL
Supports Python	Yes	Yes	No, Java	Yes	Yes	Yes	Yes	Yes	No, only Java & C++
Portability	Yes	No, Linux	Yes	Yes	Yes	Yes	Yes	Yes	No, Linux
Learning Curve[1]	Long	Long	Long	Short	Long	Average	Short	Long	Short
Scalability	Known to handle 1 Trillion triples	Unspecified	Unspecified	2^{64} nodes	Unspecified	2^{60} edges and half as many vertices	Limit on 34Bn for both nodes and relations	Unspecified	Unspecified
Inference Engine	Built-in	Built-in	Inference Engines API	Built-in	Built-in	Not integrated	Not integrated	Built-in	Unknown
Visualisation	AGWebView Webviewer	Graphileon Visualiser	Visualiser API	Integrated in Deskop Application	With Ogma JS	Supports tools	Integrated in Deskop Application	Tableau & Linkurious	GraphStudio
Support	Average	Average	Strong	Strong	Average	Strong	Strong	Strong	Strong
License	Commercial	Limited Free Edition	Open Source	Open Source	Commercial	Open Source	Open Source	Commercial	Commercial

From the above trade-off table, TypeDB, Neo4j and TigerGraph appear as the most likely candidates. TigerGraph was however discarded for its lack of portability and for its commercial licence. Neo4j is a well-established tool, with a strong community and a user-friendly query language, Cypher. TypeDB also has a very strong community, and a language, TypeQL, with a low-entry barrier. However, TypeDB has a decisive advantage on both Neo4j and JanusGraph: an integrated reasoner. Although reasoning services can be built on top of Neo4j and JanusGraph, the fact that TypeDB provides an integrated solution considerably reduces the system complexity. As a result, the TypeDB database from Vaticle is selected to support to migration of the Engineering Models.

6.3.4 Vaticle TypeDB Overview

Vaticle TypeDB[6], formerly known as Grakn, is an open source KG development tool. TypeDB is based on a hypergraph, and thus allows the modelling of n-ary relationships. The tool implements its own schema and query language, TypeQL. This section provides an overview of the TypeDB concepts and reasoner, as well as front-end interfaces. An example then illustrates the definition of the schema and rules, as well as the insertion of the data layer.

[6]`https://vaticle.com/`

TypeDB concepts

As defined in the tool documentation[7], there are three main types of schema concepts in TypeDB:

- **Entity:** The entity is a thing with a distinct existence in the domain. An entity can be abstract. As in UML, an abstract type cannot be instantiated. Concrete types can inherit the attributes and roles of an abstract type.
- **Relation:** The relation connects entities together. Since TypeDB is a hypergraph it allows n-ary relationships. The entities involved in a relationship play a role in it. The role that an entity plays in a relationship, e.g.,"*contains*", "*refersTo*", "*isContained*", "*isReferred*", has to be specified in the entity definition. Each relation requires at least one role. A role cannot exist without a relation. One entity type can play several roles from the same relation, different entities can play the same role. A relation can be abstract, but the types playing a role in that relation must also be abstract. Finally, relations may also own attributes.
- **Attribute:** Attributes are data attached to entities or relations. An attribute can be abstract. The attributes can only have one type, and can only represent an attribute. A same attribute can however be owned by different concepts types.

Table 6.3 summarises the TypeQL keywords used in version 2.0.1 to build the schema layer of the graph. If the schema is not valid then it cannot be loaded to a TypeDB database.

[7] `https://docs.vaticle.com/docs/general/quickstart`

Table 6.3: TypeQL Schema Keywords Summary

Keyword	Use	Example
define	initiate a statement to define schema-type element (e.g., entity, attribute, relation)	define mission sub entity;
undefine	initiate a statement to delete schema-type elements (e.g., entity, attribute, relation)	undefine mission sub entity;
sub	Declare a subtype, and induce inheritance. Multiple inheritances are not allowed in TypeDB, a choice of the developers. (the *spacecraft* entity now inherits all attributes and roles from the *mission* entity.	spacecraft sub mission;
abstract	Declare that a type cannot have instances (abstract)	mission sub entity, abstract;
owns	Assigns an attribute	mission sub entity, owns name;
key	Follows an attribute to assign a unique attribute to an entity or a relation. (No instances of *name* will have the same value).	mission sub entity, owns name @key;
plays	Assign a role to an entity	spacecraft sub mission, plays downlink:emitter;
relates	Declares a role within a relation	downlink sub relation, relates emitter, relates receiver;

Inference

The TypeDB reasoner enables the reasoning over data to infer new knowledge. The inference in a TypeDB knowledge graph is either rule-based or type-based. Type-based reasoning is achieved through the modelling of type hierarchies in entities, attributes and relations. If a *pear* and an *apple* are subtypes of a *fruit* entity, then to get all subtypes of *fruit*, one only needs to query for the *fruit* instances. TypeDB also enables basic compute queries to retrieve statistical information (i.e., mean and count), find the shortest path in-between two nodes, identify clusters within the KG and central nodes. Rule-based inference is further discussed below based on the TypeDB documentation.

The rules cannot be used to define a new schema element (i.e., define a new entity, attribute or relation). A rule can only be used to insert a new instance of an entity, an attribute or a relation already defined in the schema. Thus to infer a new relationships, the new roles must first be defined in the schema.

A TypeDB rule is divided in two parts. On the left had side (LHS), the "*when*" part contains the statements that will be verified. When these statements are validated, the consequence(s) contained in the right hand side (RHD), the "*then*" side, are applied. The "*when*" side of the rule is thus a conjunctive pattern while the rule's result, the "*then*" side, is atomic, meaning only one fact is inferred. Rules can be deleted. The format for defining a rule is:

```
rule ruleName:
    when
    { ## the condition(s)
    }
    then
    {   ## the consequence
    };
```

The inference process is performed at query (run) time. The inferred facts are therefore not stored in the KG. When the query is launched, the defined rules are inspected. If the patterns defined in the rules are found, the rule consequence (i.e., instantiate a new relationship) is executed. The rules are applied across the full Knowledge Graph. The rule consequence is only applied for the time of the given transaction, and is not stored in the graph. Which means that if a rule suddenly becomes invalid, no incorrect information is stored in the graph.

TypeDB Front-end

There are three different interfaces to read from and write to a TypeDB database:

- **TypeDB client**: A Java, Node.js or Python TypeDB clients allow to remotely perform database operations.

- **TypeDB console**: The console interface allows to access a TypeDB KG from a console environment.
- **TypeDB Workbase**: The Workbase is a graphical interface to visualise, write in, and query KGs. All the TypeDB KG visualisation shown in this chapter are screenshots of the Workbase.

The TypeDB Core and Workbase are open-source and can be downloaded from the Vaticle website. A complete documentation is also available on their website. Most of the scripts supporting the migration of the EMs are here developed in Python and thus rely on the Python API to interact with the database. Visualisations of TypeDB schema are done through the Workbase.

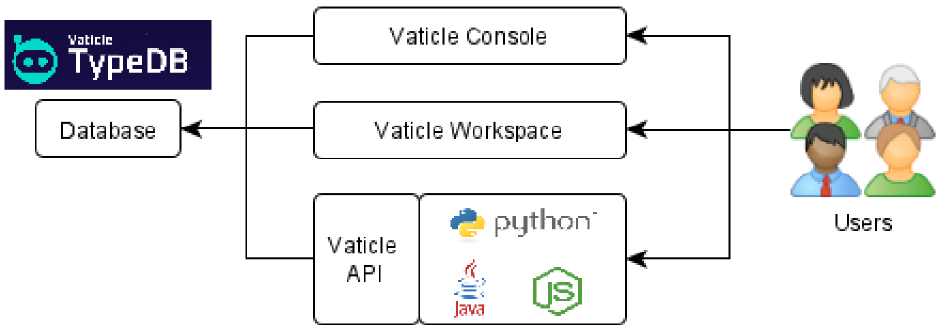
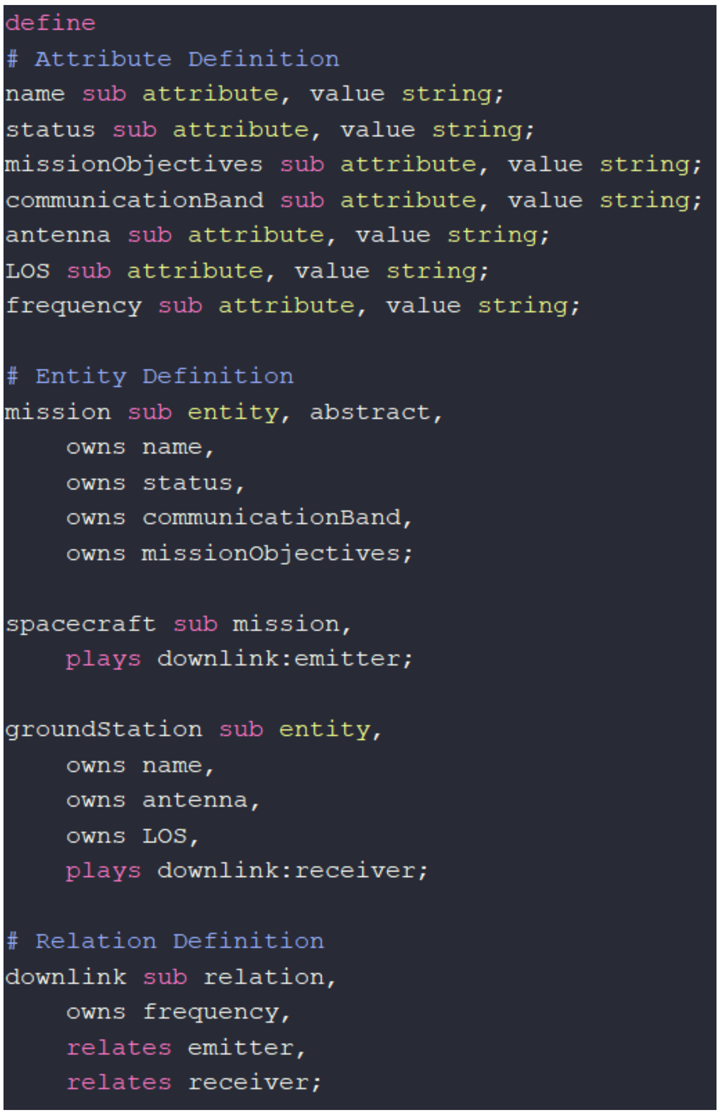


Figure 6.9: Schema of TypeDB Front-end

Schema, Data Layer and Inference Example

To illustrate the information modelling with TypeDB, a simple schema layer example is defined, then populated with data. Finally, rules are defined to infer a new relation and instantiate an attribute.

Three entities are defined in the schema layer shown on Figure 6.10: *mission*, *spacecraft*, and *groundStation*. The *spacecraft* entity inherits the attributes of the *mission* entity. A *spacecraft* entity thus has a name, a mission status, a communication band and some mission objectives. The *downlink* relationship links a *spacecraft* to a *groundStation*, with the *spacecraft* playing the role of the *emitter*, and the *groundStation* the role of the *receiver*. The *groundStation* entity has a name, a receiver antenna band (*antenna*), and an attribute indicating whether or not the station is visible by the spacecraft (*LOS* standing for Line Of Sight). The resulting schema can be visualised with the TypeDB Workbase as shown in Figure 6.11. The entities are represented by purple rectangles, relationships by diamond shapes and attributes by circles.



```
define
# Attribute Definition
name sub attribute, value string;
status sub attribute, value string;
missionObjectives sub attribute, value string;
communicationBand sub attribute, value string;
antenna sub attribute, value string;
LOS sub attribute, value string;
frequency sub attribute, value string;

# Entity Definition
mission sub entity, abstract,
    owns name,
    owns status,
    owns communicationBand,
    owns missionObjectives;

spacecraft sub mission,
    plays downlink:emitter;

groundStation sub entity,
    owns name,
    owns antenna,
    owns LOS,
    plays downlink:receiver;

# Relation Definition
downlink sub relation,
    owns frequency,
    relates emitter,
    relates receiver;
```

Figure 6.10: Example of a TypeQL schema layer for a downlink example

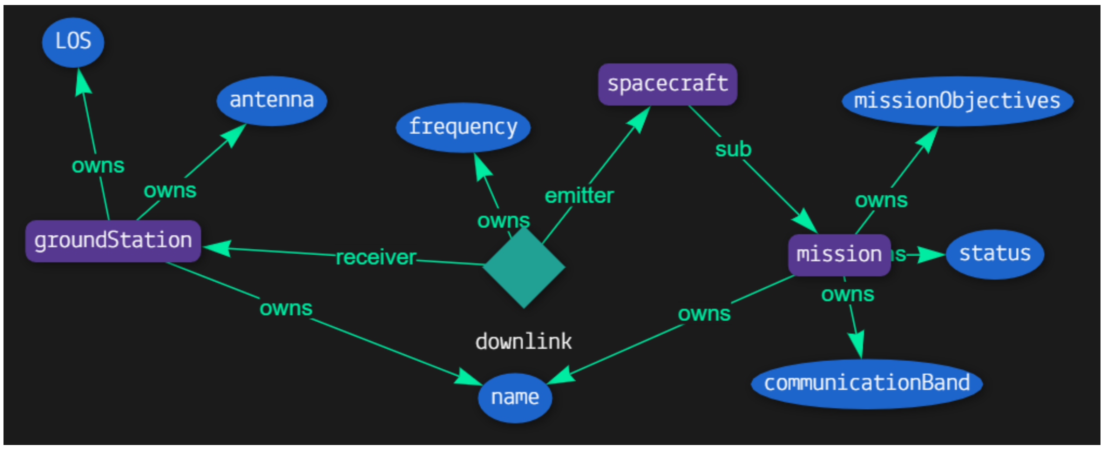
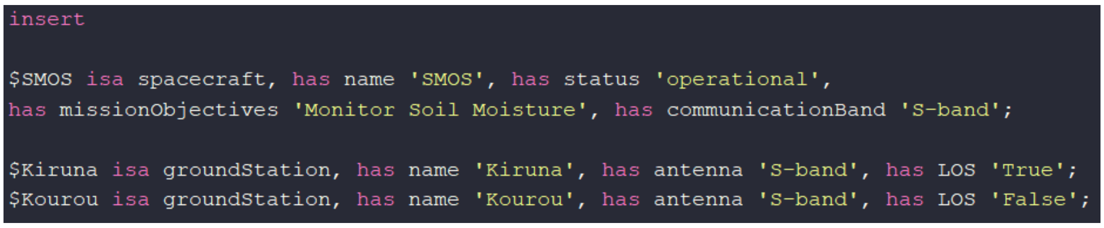


Figure 6.11: Visualisation of the schema via the TypeDB Workbase

The schema layer can then be populated with data with the keyword *insert*. A data layer, shown in Figure 6.12, is populated with one spacecraft and two ground-stations. The spacecraft SMOS is an operational mission communicating within the S-band range (frequency of 2-4 GHz). Both ground-stations, Kiruna and Kourou, have a S-band antenna and could therefore receive the data transmitted by SMOS. Only the Kiruna station is however visible by SMOS. The populated graph can be visualised through the TypeDB Workbase as shown on Figure 6.13.

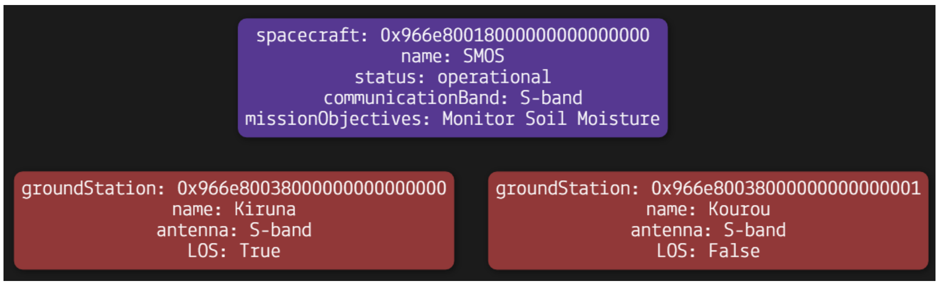


```
insert

$SMOS isa spacecraft, has name 'SMOS', has status 'operational',
has missionObjectives 'Monitor Soil Moisture', has communicationBand 'S-band';

$Kiruna isa groundStation, has name 'Kiruna', has antenna 'S-band', has LOS 'True';
$Kourou isa groundStation, has name 'Kourou', has antenna 'S-band', has LOS 'False';
```

Figure 6.12: Example of a TypeQL data layer



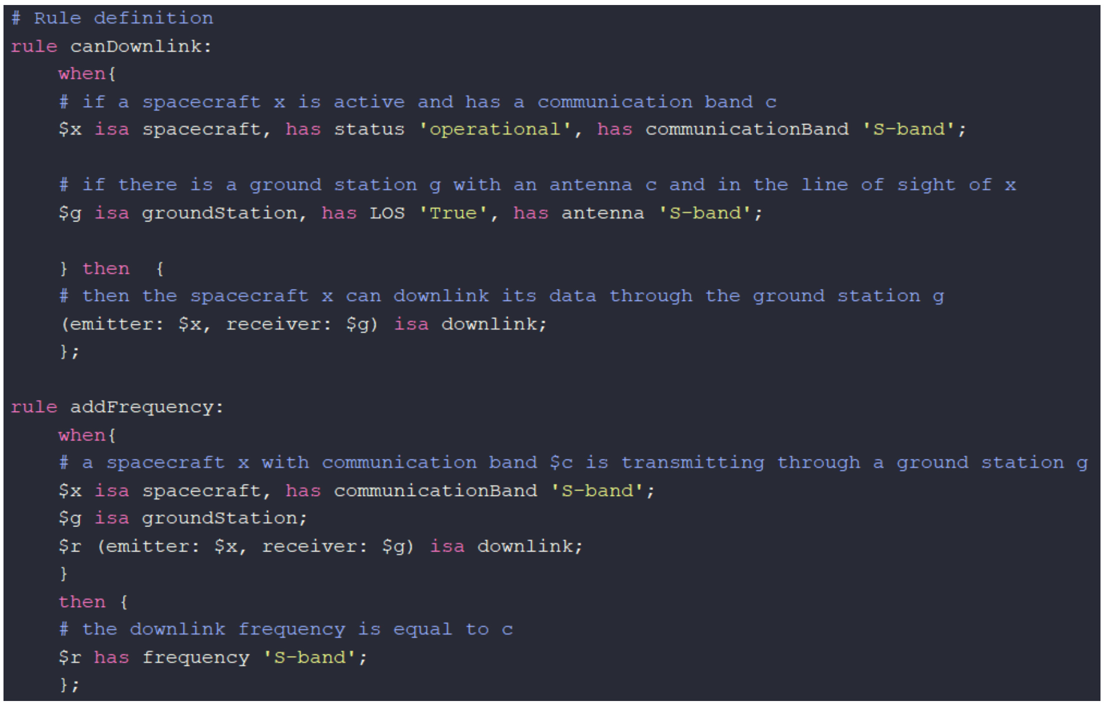
```
spacecraft: 0x966e80018000000000000000
name: SMOS
status: operational
communicationBand: S-band
missionObjectives: Monitor Soil Moisture

groundStation: 0x966e80038000000000000000
name: Kiruna
antenna: S-band
LOS: True

groundStation: 0x966e80038000000000000001
name: Kourou
antenna: S-band
LOS: False
```

Figure 6.13: Visualisation of the populated graph via the TypeDB Workbase

Now, a human would easily deduce that the SMOS spacecraft should downlink (transmit) its data through the Kiruna station as it cannot see the Kourou station. For the inference engine to reach the same conclusion, two rules, displayed in Figure 6.14, are written in TypeQL to (i) infer whether or not a spacecraft can downlink through a ground station and (ii) in which frequency this transmission is done. Note that the schema, Figure 6.10, already defined a *downlink* relation between the *spacecraft* and the *groundStation*. As shown on Figure 6.15, a relation is correctly inferred only between the SMOS spacecraft and the Kiruna station, with a S-band frequency value.



```
# Rule definition
rule canDownlink:
    when{
    # if a spacecraft x is active and has a communication band c
    $x isa spacecraft, has status 'operational', has communicationBand 'S-band';

    # if there is a ground station g with an antenna c and in the line of sight of x
    $g isa groundStation, has LOS 'True', has antenna 'S-band';

    } then  {
    # then the spacecraft x can downlink its data through the ground station g
    (emitter: $x, receiver: $g) isa downlink;
    };

rule addFrequency:
    when{
    # a spacecraft x with communication band $c is transmitting through a ground station g
    $x isa spacecraft, has communicationBand 'S-band';
    $g isa groundStation;
    $r (emitter: $x, receiver: $g) isa downlink;
    }
    then {
    # the downlink frequency is equal to c
    $r has frequency 'S-band';
    };
```

Figure 6.14: Example of a TypeQL rule

6.3.5 Background Summary

This Literature Review placed the Knowledge Graph in the landscape of information modelling techniques. Knowledge Graphs are graph databases bearing formal semantics. They therefore inherit the attributes of graph databases and are adapted to relationship-heavy use cases with heterogeneous data. As KGs bear semantics they are built either from RDF graphs or from property graphs defined with a schema. The

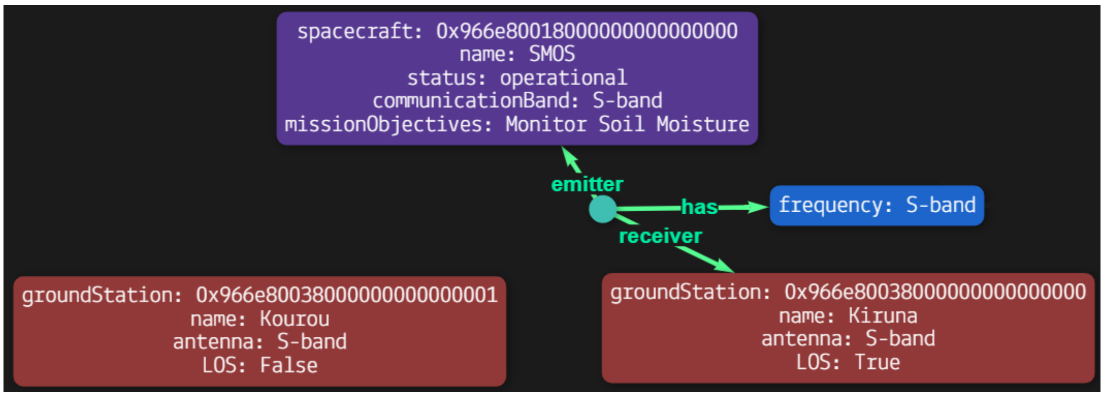

spacecraft: 0x966e8001800000000000000
name: SMOS
status: operational
communicationBand: S-band
missionObjectives: Monitor Soil Moisture
emitter
has
frequency: S-band
receiver
groundStation: 0x966e8003800000000000001
name: Kourou
antenna: S-band
LOS: False
groundStation: 0x966e8003800000000000000
name: Kiruna
antenna: S-band
LOS: True

Figure 6.15: Inference result visualised with the TypeDB Workbase

schema, similar to an ontology, structures the semantics of the graph, defining the entities, attributes and relationships allowed in the KG. The ontology enables the rule-based reasoning. Several KG modelling tools are available on the market. Following a trade-off in between those tools, TypeDB from Vaticle is selected.

6.4 Approach

As seen in Section 3.3, the EMs are structured according to a System Engineering Information Model (SEIM) defined in the ECSS-E-TM-10-25A technical memorandum and available as a UML model. This SEIM defines the allowed classes, properties, and relationships. On the other hand, the previous section established that the preferred database for storing the Engineering Models and inferring new insights from them would be a Knowledge Graph (KG). Furthermore the trade-off between available KG tools suggests the use of the Vaticle TypeDB database.

To migrate the EMs to a TypeDB KG, the structure of the EM first needs to be mapped from the UML model to a TypeDB schema. Thus, the structure of the EMs will be understandable by the KG. The UML model and TypeDB building blocks as well as the mapping rules are presented in the following section. Then, automatic migration pipelines export the content of the EMs to the KG. The mapping of the UML model defines the so-called schema layer of the KG, while the population with the EMs forms the KG's data layer. The approach, summarised in Figure 6.16, is tailored to EMs based on the ECSS-E-TM-10-25A SEIM.

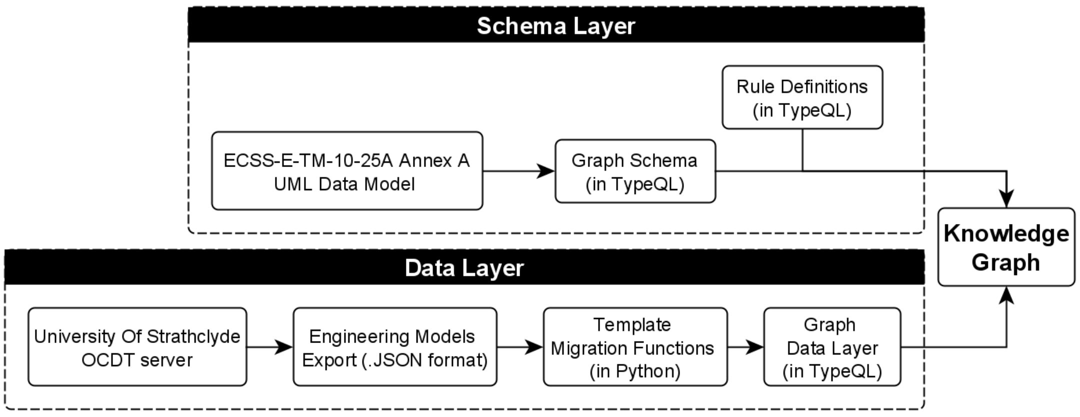

Schema Layer
Rule Definitions (in TypeQL)
ECSS-E-TM-10-25A Annex A UML Data Model
Graph Schema (in TypeQL)
Knowledge Graph
Data Layer
University Of Strathclyde OCDT server
Engineering Models Export (.JSON format)
Template Migration Functions (in Python)
Graph Data Layer (in TypeQL)

Figure 6.16: Approach to migrate Engineering Models to a TypeDB Knowledge Graph

6.5 Methodology: The Migration Building Blocks

This section introduces a novel methodology to map UML class diagram to TypeQL concepts. The mapping is illustrated with the migration of the *Person* class. Metadata extracted from the reports is added to the Knowledge Graph to enrich the models. The mapping rules presented here are later applied to migrate the ECSS-E-TM-10-25A class diagram to a TypeDB schema, and the ECSS-based Engineering Models to a TypeDB data layer.

6.5.1 Mapping Rules

The following migration rules were defined with the support of Hans Peter de Koning (ESA) and Sam Gerené (RHEA). UML classes, properties, and relations are respectively mapped to TypeQL entities, attributes, and relation.

Data Types Equivalence

There are 5 data types available in TypeQL: *long*, *double*, *string*, *boolean*, and *date-time*. The *datetime* type is a date or a date-time. The standard UML primitive types include *Boolean*, *Integer*, *String*, *Unlimited Natural* and *Real*. Other types found in the class diagrams are mapped to *string* by default. Table 6.4 summarises the data types mapping from UML to TypeQL.

Enumeration types are additional data types in UML which can only take pre-defined values. For instance, *ClassKind* is an enumeration datatype that lists all possible classes names. Each class has a *ClassKind* property, which can only take a value that corresponds to a class from the enumeration list, such as *ElementUsage* or *Iteration*. Since the migration is read-only, there is no need to store in TypeDB all the values that the enumeration type could take. The enumeration types are simply migrated as attributes of types string. For instance, the property *classKind* is migrated as an attribute, and instantiated with the value found in the engineering model. The enumeration types mapped from the ECSS UML model to a TypeQL attribute of type *string* are: *ClassKind*, *LogLevelKind*, *ParticipantAccessRightKind*, *PersonAc-*

Table 6.4: Mapping of UML Basic Data Types to TypeQL

UML	**TypeQL**
Boolean	boolean
Date	datetime
DateTime	datetime
Integer	double
LongInteger	long
Real	double
String	string
TimeOfDay	datetime
FileContentType	string
LanguageCode	string
ParameterFormulaType	string
ParameterValueType	string
Sha1HashType	string
Uri	string
Uuid	string

cessRightKind, *ActualFiniteStateKind*, *BooleanOperatorKind*, *InterfaceEndKind*, *ParameterSwitchKind*, *RelationalOperatorKind*, *RulesVerificationStatusKind*, *EngineeringModelKind*, *LogarithmBaseKind*, *NumberSetKind*, *StudyPhaseKind*, *VcardEmailAddressKind*, *VcardTelephoneNumberKind*, and *RuleVerificationStatusKind*.

From UML class to TypeQL entities

UML classes can be abstract, so can TypeQL entities. An abstract UML class maps to an abstract TypeQL entity, and a concrete class maps to a concrete TypeQL class.

The ECSS-E-TM-10-25 user manual available on the OCDT community git[8] indicates that classes may also be "Mixin", an OO programming concept. A Mixin is, according to the manual, *a class that can be "mixed-in" from the side during code generation.* This type of class is similar to UML interfaces: they contains methods for use by other classes without being the parent class. There is no equivalent to a Mixin class in TypeQL. Instead, the features of the Mixin class become attributes or roles of the entity interacting with the Mixin class. This solution is preferable to defining an abstract class for the Mixin class as TypeQL allows only one inheritance. These rules are summarised in Figure 6.17.

[8] `https://ocdt.esa.int/`

For instance, in the ECSS UML model, the *Person* class is a subclass (child class) of *Thing*, *ShortNamedThing*, *NamedThing*, and *DeprecatableThing*. While *Person* is a concrete class, the other three classes (ShortNamedThing, NamedThing, and DeprecableThing) are actually abstract Mixin classes. In the TypeQL schema, the *Person* entity will directly own the attributes and roles that were defined by these Mixin classes, and only inherits, via the TypeQL "sub" keyword, from the *Thing* entity.

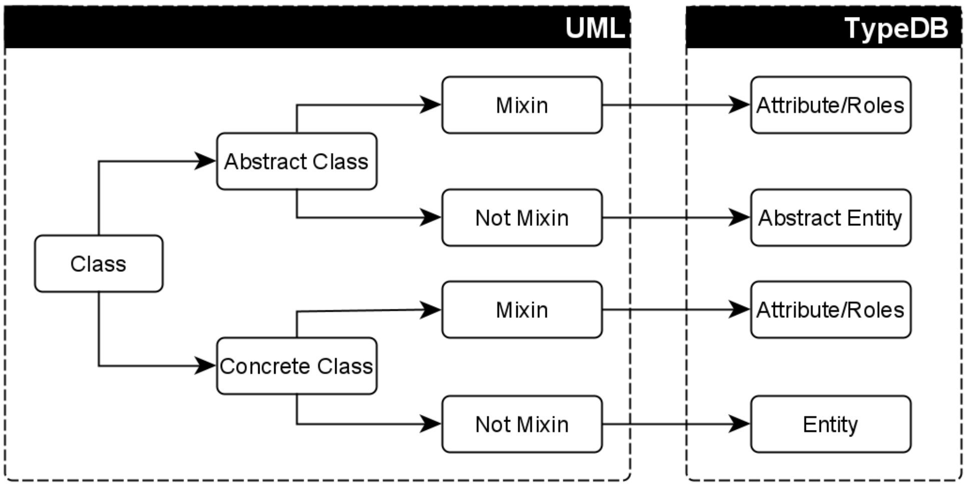


Figure 6.17: Migration of UML class to TypeQL entities

From UML Property to TypeQL Attribute or Role

A UML property can be an attribute (value type) or the end of a relation (reference type). The property is either directly owned by the class or inherited from another class. Since TypeDB allows inheritance, the migration of properties is rather straightforward:

- If the property is owned by the class and is an attribute, then it is defined as a TypeDB attribute of the entity.
- If the property is owned and refers to the end of an association relation, then it is mapped to a TypeDB role.
- If the property is inherited then no action is needed as it is defined in the parent entity. One crucial caveat, however, is for properties inherited from Mixin classes. These must be redefined as owned by the entity.

The above rules are summarised in Figure 6.18. In UML, a property can also be derived. The result of the derivation will be mapped to TypeQL but not the derivation capability itself. For instance, if the *age* attribute is derived from the *birthday* attribute, then if the *birthday* value is modified in the TypeQL schema, the *age* will not be derived again and will retain its previous value. Since the intent of this migration is not to alter the content of the EMs but extract new insights from them, this feature loss is deemed acceptable.

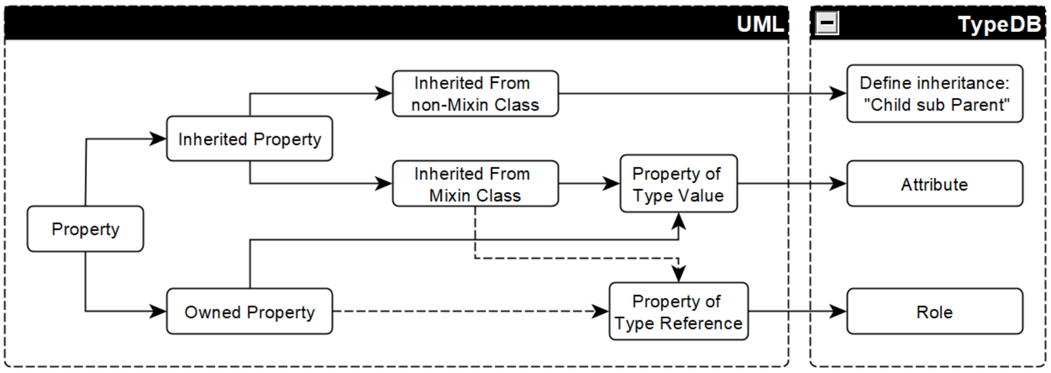

UML
TypeDB
Property
Inherited Property
Owned Property
Inherited From non-Mixin Class
Inherited From Mixin Class
Property of Type Value
Property of Type Reference
Define inheritance: "Child sub Parent"
Attribute
Role

Figure 6.18: Migration of UML properties, either attribute or references, to TypeDB attributes or roles

From UML Relationships to TypeQL Relations

Although TypeDB allows the definition of n-ary relations (being an hypergraph) and the assignment of attributes to relations, it does not match the richness of relationships found in UML. As shown on Figure 6.19, most of the UML relationships are mapped as binary or n-ary TypeQL relations. The Inheritance or Generalisation relationship can be mapped with the inheritance keyword of *sub* in TypeQL.

However, a naming convention is implemented in the migration to differentiate from the two most common relationships found in the ECSS model: containment (composition) and reference (directed association). As can be seen in Figure 6.20, the relations between the classes are either a directed association denoted as "*refers*" or a composition denoted as "*contains*". The differentiation is superficially represented by a different naming convention in the TypeDB schema. Containment relations are named *Containment_parameterName* and reference relations, *Reference_parameterName*. A containment relationship and a reference relationship between *entity1* and *entity2* would thus respectively be defined as:

```
entity1 sub entity, plays Containment_{parameterName}:contains;
entity1 sub entity, plays Containment_{parameterName}:isContained;

Containment_{parameterName} sub relation,
    relates contains,
    relates isContained;

entity1 sub entity, plays Reference_{parameterName}:refersTo;
entity1 sub entity, plays Reference_{parameterName}:isReferredBy;

Reference_{parameterName} sub relation,
        relates refersTo,
        relates isReferredBy;
```

Note on multiplicity: In UML, class attributes may take more than one value or a class might be associated to more than one class. For instance, a Person class may contain more than one email address. The multiplicity is not mapped to the TypeQL schema. A Person entity will simply be related to one email address entity, and the multiplicity information is lost in the migrated conceptual model. This is not a show-stopper as the multiplicity is only relevant for the data layer, when the entities are instantiated. Then there will be as many attributes and relationships inserted as was found in the EM. But the multiplicity information will be lost in the migration and cannot be mapped back from TypeQL to UML.

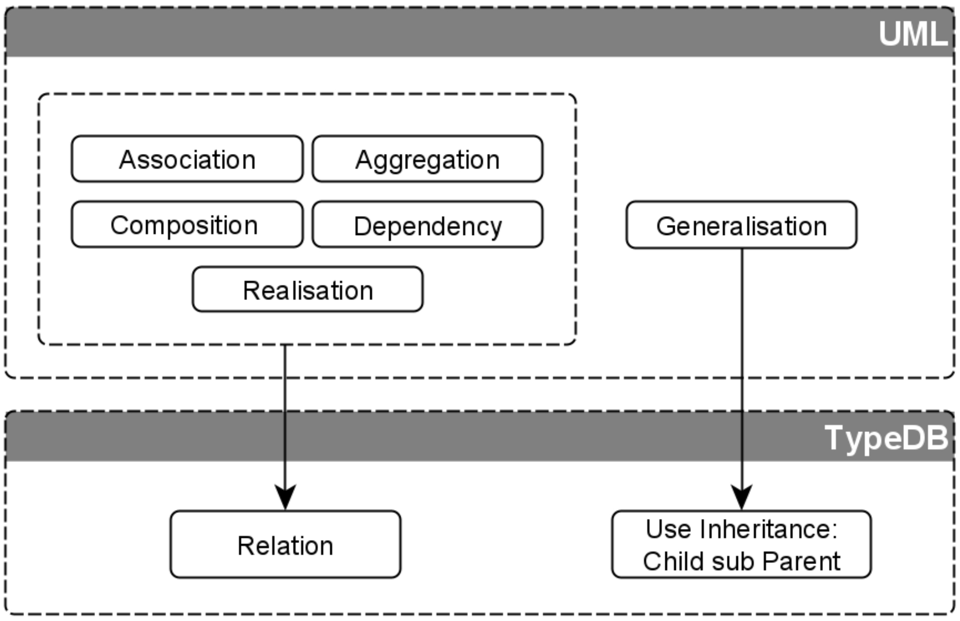


Figure 6.19: Migration of UML relationships to TypeDB relations

TypeDB Schema Generation

The migration is done with the version 0.9.1 of *xmi_verter*, available on the ESA gitlab[9], and with the version 2.0.1 of TypeDB downloaded from the Vaticle webpage[10]. The *xmi_verter* package was developed by ESA, it reads an XMI file of the ECSS UML model and generates several implementation technology representations, for instance in *C#* and SQL. A new script, *xmi2typeql.py*, is written in Python to generate a representation in TypeQL, based on the mapping rules. The *xmi_verter.py* script verifies the compliance with the ECSS-E-TM-10-25 data modelling rules. The TypeDB schema generated with the *xmi_verter.py* and *xmi2typeql.py* scripts can then be defined in a TypeDB database.

TypeDB Data Layer Generation

The Engineering Models are exported as .JSON files from the CDP4. Entity templates are developed in Python for each of the 126 UML (non-mixin) classes found in the

[9]`https://gitlab.esa.int/`
[10]`https://vaticle.com/`

exported models. These templates generate the TypeQL insert queries, from the .JSON files, to populate the TypeDB database, inserting all attributes, roles, and relations. The migration templates are available at `github.com/strath-ace/smart-nlp`.

6.5.2 Migration Validation

The 126 different class kinds found in the UML data model are all mapped to a TypeDB entity. A verification is manually done by a human annotator. To demonstrate the validity of the migration, the example of the *Person* class is taken. This class represents a physical person that can participate to a concurrent engineering activity. Figure 6.20 displays an extract of the UML class diagram focusing on the *Person* class. For clarity, an extract of the ECSS-E-TM-10-25A User Manual is also provided in Figure 6.21.

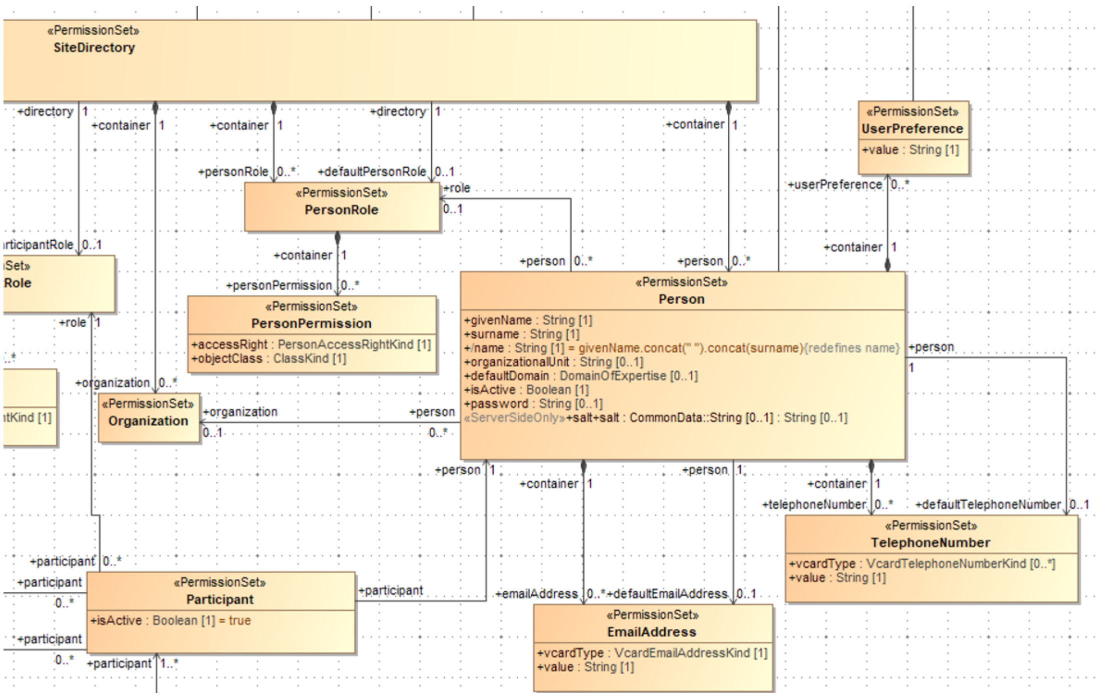
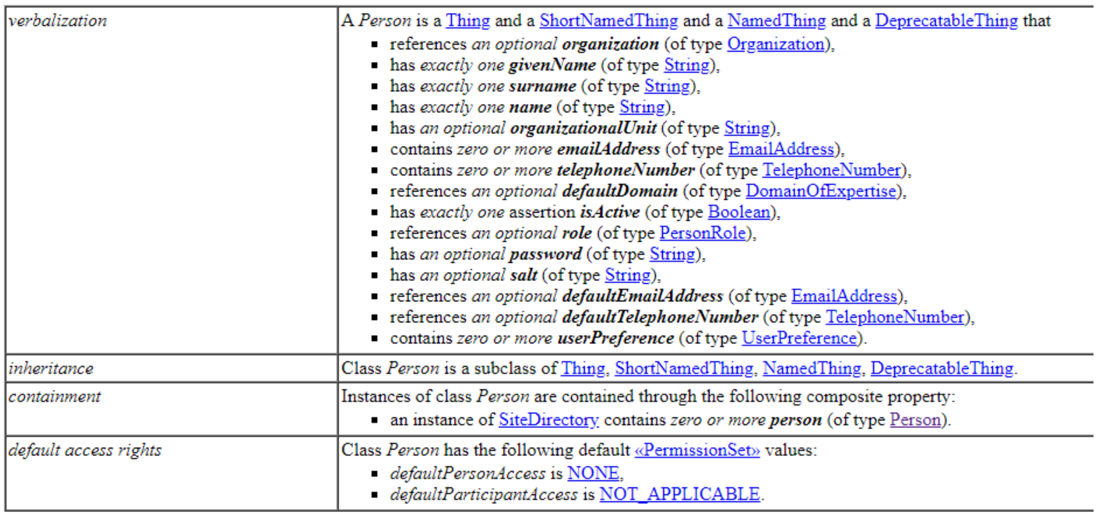

«PermissionSet» SiteDirectory
«PermissionSet» UserPreference
+value : String [1]
«PermissionSet» PersonRole
«PermissionSet» PersonPermission
+accessRight : PersonAccessRightKind [1]
+objectClass : ClassKind [1]
«PermissionSet» Person
+givenName : String [1]
+surname : String [1]
+/name : String [1] = givenName.concat(" ").concat(surname){redefines name}
+organizationalUnit : String [0..1]
+defaultDomain : DomainOfExpertise [0..1]
+isActive : Boolean [1]
+password : String [0..1]
«ServerSideOnly»+salt+salt : CommonData::String [0..1] : String [0..1]
«PermissionSet» Organization
«PermissionSet» TelephoneNumber
+vcardType : VcardTelephoneNumberKind [0..*]
+value : String [1]
«PermissionSet» EmailAddress
+vcardType : VcardEmailAddressKind [1]
+value : String [1]
«PermissionSet» Participant
+isActive : Boolean [1] = true
+directory
+container
+personRole
+defaultPersonRole
+role
+person
+userPreference
+personPermission
+organization
+participant
+emailAddress
+defaultEmailAddress
+telephoneNumber
+defaultTelephoneNumber

Figure 6.20: Extract of the UML class model, focused on the Person class

Based on Figure 6.20 and Figure 6.21, the following information on the *Person* class is deduced:

- **Inheritance:** The *Person* inherits from the *Thing*, *ShortNamedThing*, *NamedThing*, and *DeprecableThing* classes. Since the three last classes are mixin, their prop-



verbalization	A *Person* is a Thing and a ShortNamedThing and a NamedThing and a DeprecatableThing that ▪ references *an optional* ***organization*** (of type Organization), ▪ has *exactly one* ***givenName*** (of type String), ▪ has *exactly one* ***surname*** (of type String), ▪ has *exactly one* ***name*** (of type String), ▪ has *an optional* ***organizationalUnit*** (of type String), ▪ contains *zero or more* ***emailAddress*** (of type EmailAddress), ▪ contains *zero or more* ***telephoneNumber*** (of type TelephoneNumber), ▪ references *an optional* ***defaultDomain*** (of type DomainOfExpertise), ▪ has *exactly one* assertion ***isActive*** (of type Boolean), ▪ references *an optional* ***role*** (of type PersonRole), ▪ has *an optional* ***password*** (of type String), ▪ has *an optional* ***salt*** (of type String), ▪ references *an optional* ***defaultEmailAddress*** (of type EmailAddress), ▪ references *an optional* ***defaultTelephoneNumber*** (of type TelephoneNumber), ▪ contains *zero or more* ***userPreference*** (of type UserPreference).
inheritance	Class *Person* is a subclass of Thing, ShortNamedThing, NamedThing, DeprecatableThing.
containment	Instances of class *Person* are contained through the following composite property: ▪ an instance of SiteDirectory contains *zero or more* ***person*** (of type Person).
default access rights	Class *Person* has the following default «PermissionSet» values: ▪ *defaultPersonAccess* is NONE, ▪ *defaultParticipantAccess* is NOT_APPLICABLE.

Figure 6.21: Extract of the ECSS-E-TM-10-25 User Manual

erties are re-defined as owned parameters in the *Person* class. The *Person* class should only inherit from the *Thing* class.

- **Attributes:** The attributes of value type owned by the *Person* are shown in Table 6.5. Those should be defined as attributes of the *Person* entity in TypeQL. The attributes inherited by the *Thing* class (*iid*, *revisionNumber* and *classKind*) should not be re-defined in the *Person* TypeQL entity.
- **Relations:** According to the User Manual, the relations that should be re-defined in the *Person* entity are summarised in Table 6.6. When a property is the end of the relationship, a role is also created in the other entity involved in the relation.

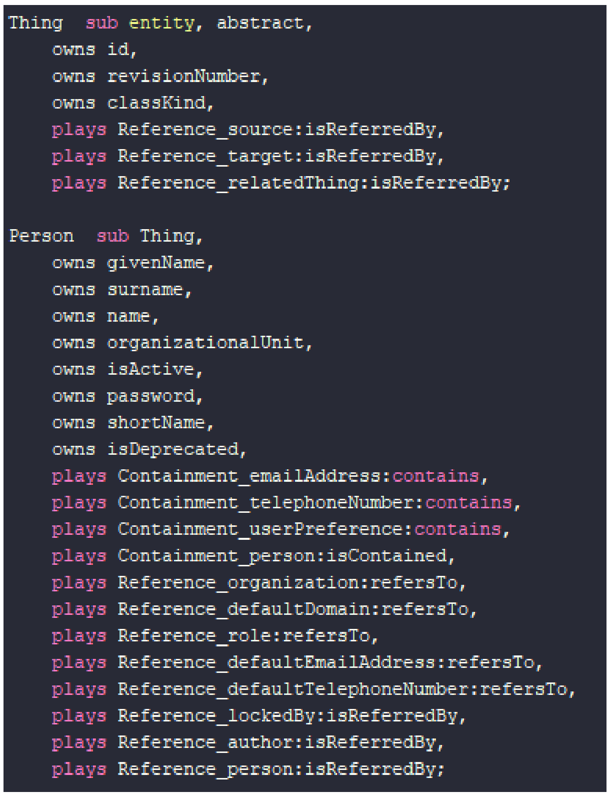
Table 6.5: Attributes owned and inherited from Mixin by the Person class

Attribute	Type	Inheritance
shortName	String[1..1]	ShortNamedThing
isDeprecated	Boolean[1..1]	DeprecatedThing
givenName	String[1..1]	owned
surname	String[1..1]	owned
organisationalUnit	String[0..1]	owned
isActive	Boolean[1..1]	owned
password	String[0..1]	owned

Table 6.6: Relations owned and inherited from Mixin by the Person class

Relation Type	Relation Name	Class Type	Inheritance
Containment	emailAddress	EmailAddress [0..*]	owned
	telephoneNumber	TelephoneNumber [0..*]	owned
	userPreference	UserPreference [0..*]	owned
Reference	organisation	Organisation [0..1]	owned
	defaultDomain	DomainOfExpertise [0..1]	owned
	role	PersonRole [0..1]	owned
	defaultEmailAddress	EmailAddress [0..1]	owned
	defaultTelephoneNumber	TelephoneNumber [0..1]	owned

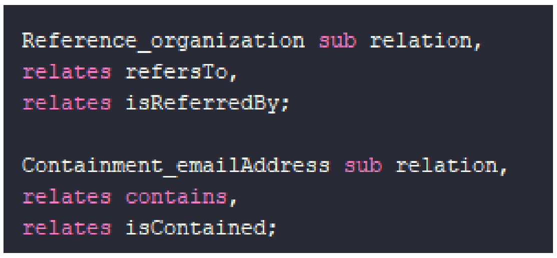
As can be seen on Figure 6.22, the inheritance, attributes and relationships are correctly mapped from the *Person* UML class to the TypeDB *Person* entity. The attributes types are defined based on the data type equivalence presented in Section 6.5.1. Relations and the roles played by the other entities are generated simultaneously as show in Figure 6.23.



```
Thing  sub entity, abstract,
    owns id,
    owns revisionNumber,
    owns classKind,
    plays Reference_source:isReferredBy,
    plays Reference_target:isReferredBy,
    plays Reference_relatedThing:isReferredBy;

Person  sub Thing,
    owns givenName,
    owns surname,
    owns name,
    owns organizationalUnit,
    owns isActive,
    owns password,
    owns shortName,
    owns isDeprecated,
    plays Containment_emailAddress:contains,
    plays Containment_telephoneNumber:contains,
    plays Containment_userPreference:contains,
    plays Containment_person:isContained,
    plays Reference_organization:refersTo,
    plays Reference_defaultDomain:refersTo,
    plays Reference_role:refersTo,
    plays Reference_defaultEmailAddress:refersTo,
    plays Reference_defaultTelephoneNumber:refersTo,
    plays Reference_lockedBy:isReferredBy,
    plays Reference_author:isReferredBy,
    plays Reference_person:isReferredBy;
```

Figure 6.22: *Person* entity as defined in the TypeQL schema. The *Thing* entity is provided as information for the inheritance.



```
Reference_organization sub relation,
relates refersTo,
relates isReferredBy;

Containment_emailAddress sub relation,
relates contains,
relates isContained;
```

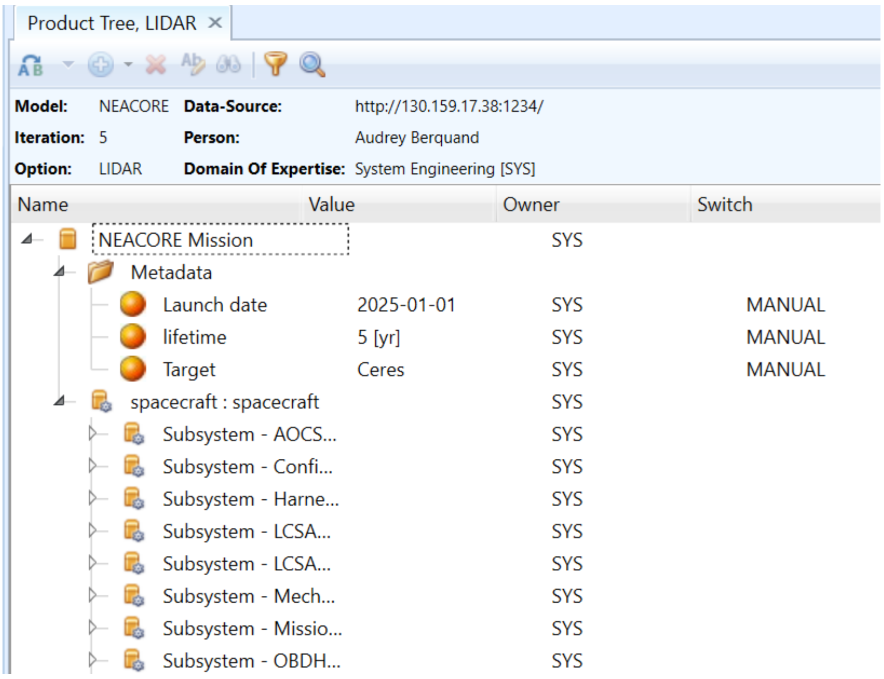
Figure 6.23: Sample of rules related to *Person* entity

6.5.3 Integration of Additional Metadata

The CDF team is currently deploying a study portal where past feasibility reports are identified by a set of metadata. These parameters cover the main characteristics of a mission, including, for instance, the system requirements, the mission objectives and target system. They also encompass main design information such as the orbit, payload and propulsion types, and information on the design sessions, for instance, the participants.

The metadata is not necessarily integrated to the EMs yet. As this study was done in collaboration with ESA, and although no CDF EMs were migrated, the TypeDB schema is slightly modified to accommodate the metadata set. The integration of the metadata to the schema is meant to be the less disruptive as possible, thus relying on concepts already defined in the ECSS-E-TM-10-25A technical memorandum to avoid modifying the Engineering Models themselves.

The suggested approach is the equivalent of defining a parameter group containing the metadata at the root of the iteration product tree, as shown in Figure 6.24. The set of parameters is an instantiation of the *ParameterGroup* entity. Parameters from the group are stored as instantiation of the *Parameter* class, and linked to a *ParameterType* and a *ParameterValueSet*. For instance *lifetime* is a *ParameterType* and *5* the published *ParameterValueSet*. The *Team Composition* metadata is already represented by the *Participant* entity, as well as the *System Requirements* by the *Requirement* entity. Rules are thus defined to link the Metadata entity to those two pre-existing entities. To avoid modifying the EMs, the metadata to be inserted are manually stored in Python dictionaries and inserted into the graph already populated with EMs. The methodology is summarised in Figure 6.25.



Product Tree, LIDAR

Model:	NEACORE	**Data-Source:**	http://130.159.17.38:1234/
Iteration:	5	**Person:**	Audrey Berquand
Option:	LIDAR	**Domain Of Expertise:**	System Engineering [SYS]

Name	Value	Owner	Switch
NEACORE Mission		SYS	
Metadata			
Launch date	2025-01-01	SYS	MANUAL
lifetime	5 [yr]	SYS	MANUAL
Target	Ceres	SYS	MANUAL
spacecraft : spacecraft		SYS	
Subsystem - AOCS...		SYS	
Subsystem - Confi...		SYS	
Subsystem - Harne...		SYS	
Subsystem - LCSA...		SYS	
Subsystem - LCSA...		SYS	
Subsystem - Mech...		SYS	
Subsystem - Missio...		SYS	
Subsystem - OBDH...		SYS	

Figure 6.24: Adding a Metadata parameter group to the NEACORE CubeSat Engineering Model in CDP4.

The only necessary alteration to the original schema is the re-definition of the *ParameterGroup* entity to allow it to play two new roles, relating it to the *Participant* and *Requirement* entities. Two TypeQL rules are defined to link a *ParameterGroup* named “Metadata” to all *Participant* and *Requirement* entities instances.

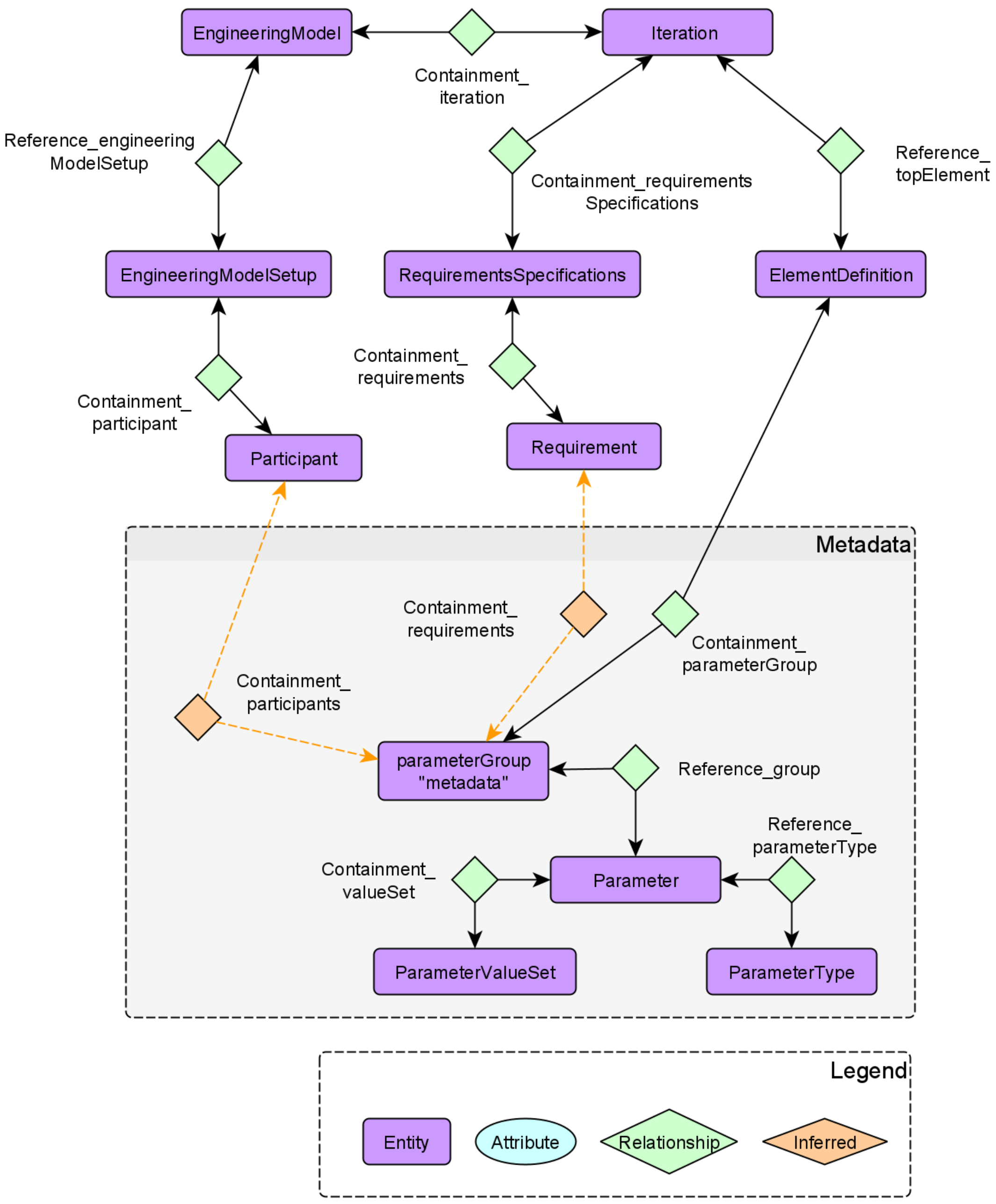


Figure 6.25: Methodology to add the metadata content on top of the ECSS schema

6.6 Applications

The migration is applied to the two EMs belonging to the University of Strathclyde and introduced in Section 3.3.2, the NEACORE and STRATHcube missions. One iteration from each mission is thus migrated to a same TypeDB Knowledge Graph. The STRATHcube iteration contains three design options, based either on a CubeSat 2U or 3U platform, and with or without a propulsion system. The propulsion system can only be accommodated on a 3U platform. The NEACORE iteration contains two design options, depending on which payloads were flown, either a LIDAR and a camera, or a spectrometer and a camera. The population of the KG is done automatically from the EM's exports via Python pipelines. It takes around 15 minutes to insert the data from each EM into the TypeDB database. The NEACORE iteration contains 22,183 nodes and edges (18% entities, 40% attributes, and 42% relations). The STRATHcube iteration contains 24,039 nodes and edges (18% entities, 41% attributes, and 41% relations).

6.6.1 Rule-Based Inference: Automatic Mass Budget Generation

As introduced in Section 6.2, the computation of the mass budget is usually done manually, often by means of excel spreadsheets, by the system engineers. The total spacecraft mass is a crucial parameter influencing the launcher selection and mission cost. In this case study, a mass budget is automatically inferred for each design options found in an EM iteration. Generating the budget automatically alleviates the experts' workload and avoids human mistakes. New insights could also be derived from the evolution of mass and power budgets across the different iterations of an EM.

Approach

There is no relationship to express that an element is contained within the mass budget of a design option in the ECSS-E-TM-10-25A standard. However, this relationship can be inferred. The inference is done in four steps: (i) a design iteration is found in the KG, (ii) the mass parameters of equipment found in the iteration's product tree are extracted, (iii) the design option(s) the parameter belong to is/are identified, and (iv) a new relation is created to include the parameter into the relevant design options' mass budget.

Methodology

A new relationship, *includedInMassBudget*, is defined in the TypeDB schema to relate a design option to a parameter value. To ensure that only parameters identified as a mass are retained, the *Parameter* entity must refer to a *SimpleQuantityKind* entity with an attribute name "*mass*". A *Parameter* entity does not contain a value attribute, instead it relates to a *ParameterValueSet* entity which indicates the parameter type. Since the parameter value may change depending on the design option, the relationship *includedInMassBudget* is created between the *Option* entity and the *ParameterValueSet* entity rather than at the parameter level.

The *Parameter* entity does contain a Boolean attribute *isOptionDependent* indicating whether or not the parameter varies per design option. When a parameter is option-dependent, it is usually linked to the option it depends on by a *refers_actualOption* relation. When a parameter is option-independent, it does not necessarily mean that it should be linked to all options' mass budgets. The parameter might indirectly be excluded from a design option by a *refers_excludeOption* relation. This logic is summarised by a decision tree shown in Figure 6.26, and is illustrated in Figures 6.27 and 6.28. Eventually, two TypeQL rules, simplified in Table 6.7, are applied to the KG to infer the *includedInMassBudget* relationship.

Inference Outputs:

Two inference examples are respectively visualised in Figure 6.29 and Figure 6.31 through the TypeDB Interface for the STRATHcube mission. For clarity, only the relevant nodes and edges are shown.

In Figure 6.29, three new relationships, appearing in purple, have successfully been inferred between each option-dependent parameter and the corresponding option they referred to. The mass parameter with id *V467104* refers to deployable solar cells. The

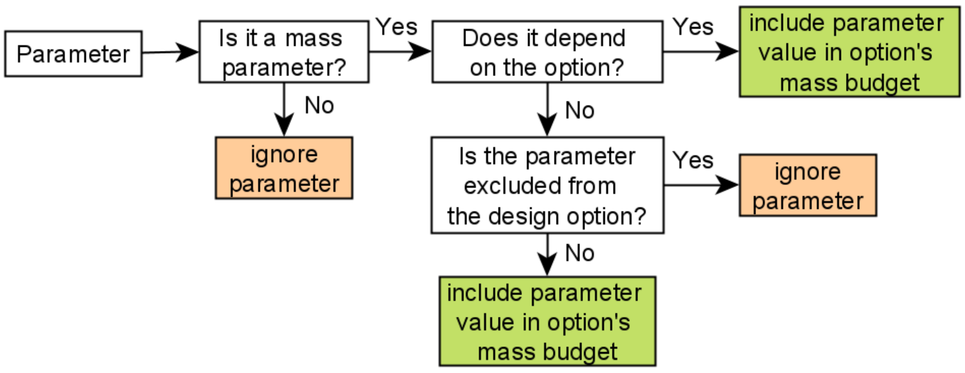

Parameter
Is it a mass parameter?
Yes
No
ignore parameter
Does it depend on the option?
Yes
include parameter value in option's mass budget
No
Is the parameter excluded from the design option?
Yes
ignore parameter
No
include parameter value in option's mass budget

Figure 6.26: Decision tree to infer an *includedInMassBudget* relationship between a *Parameter* and an *Option*

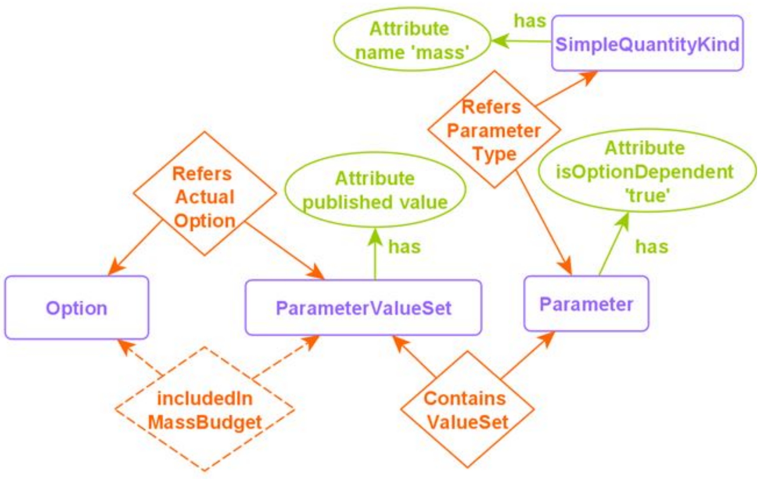


Figure 6.27: Inferring an *includedInMassBudget* relationship for an option dependent parameter. The entities are rectangles, the relationship diamonds and the attributes circles. Dotted shapes are inferred.

parameter is option dependent, meaning its values vary depending on the design option, and it indeed has two different values, either 57g or 107g. The definition of the solar cells element as input in the CDP4 is shown on Figure 6.30. As shown on Figure 6.29, the parameter values with ids *V41381960* and *V995504* were respectively linked to the second and third options by a *refers_actualOption* relation. The parameter value with id *V*438320 is linked by a similar relation to the first design option. This case, with an option-dependent parameter, corresponds to the first inference rule. As can be seen

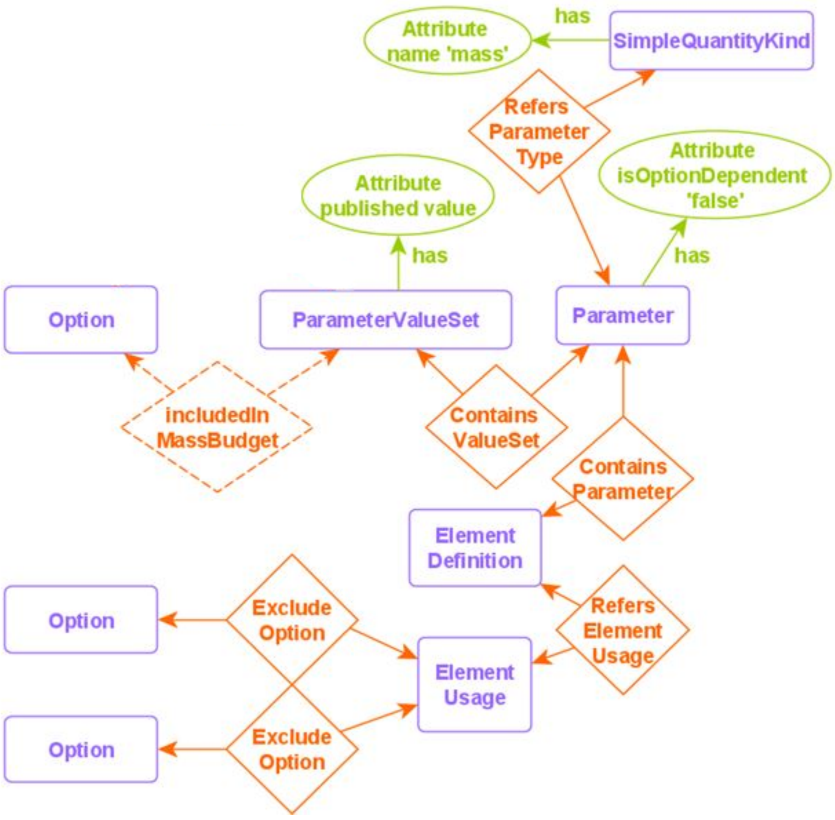


Figure 6.28: Inferring an *includedInMassBudget* relationship when the parameter is option independent. The entities are rectangles, the relationship diamonds and the attributes circles. Dotted shapes are inferred.

on the figure, the reasoner successfully links each parameter value set to the design options it refers to.

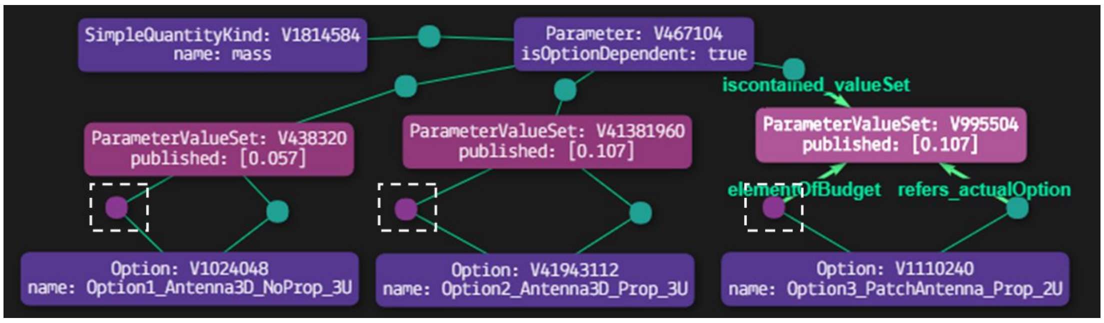
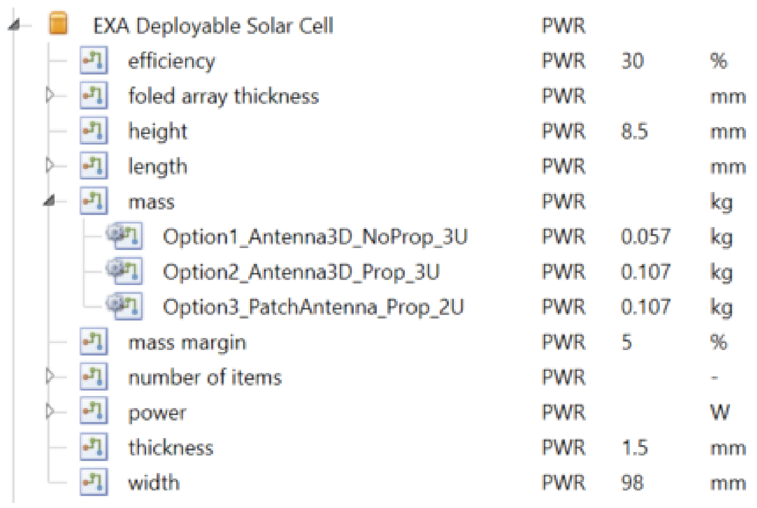


Figure 6.29: Inference outcomes from Rule 1 visualised with the TypeDB Workbase. The three inferred edges are denoted by purple circles and framed in manually added boxes.

Table 6.7: Pseudo-code of Rules

Rule 1: The Parameter is option dependent **when:** 1. There is a ParameterValueSet, contained by an option dependent Parameter, 2. The same Parameter refers to a SimpleQuantityKind with name “mass”, 3. There is an element of class Option which the element of ParameterValueSet refers to as ActualOption., **then:** The element of class ParameterValueSet is included in the Option’s mass budget. **Rule 2: The Parameter is option independent** **when:** 1. There is a ParameterValueSet, contained by an option independent Parameter, 2. The same Parameter Type refers to a SimpleQuantityKind with name “mass”, 3. The ElementUsage associated with the same Parameter through an ElementDefinition does not exclude the Option., **then:** The element of class ParameterValueSet is included in the Option’s mass budget.



EXA Deployable Solar Cell	PWR		
efficiency	PWR	30	%
foled array thickness	PWR		mm
height	PWR	8.5	mm
length	PWR		mm
mass	PWR		kg
Option1_Antenna3D_NoProp_3U	PWR	0.057	kg
Option2_Antenna3D_Prop_3U	PWR	0.107	kg
Option3_PatchAntenna_Prop_2U	PWR	0.107	kg
mass margin	PWR	5	%
number of items	PWR		-
power	PWR		W
thickness	PWR	1.5	mm
width	PWR	98	mm

Figure 6.30: Definition of the solar cell mass parameters in CDP4

Figure 6.31 addresses the second inference rule. An element corresponding to a 3U CubeSat structure is not defined as option-dependent. However, it should obviously only be included in the budgets of the first and second design options, both based on 3U platforms. The model does indicate that the element is excluded from the third design option via a *refers_excludeOption* relationship. As can be seen on the figure, relationships were successfully inferred and created between the parameter's mass value and the first and second options.

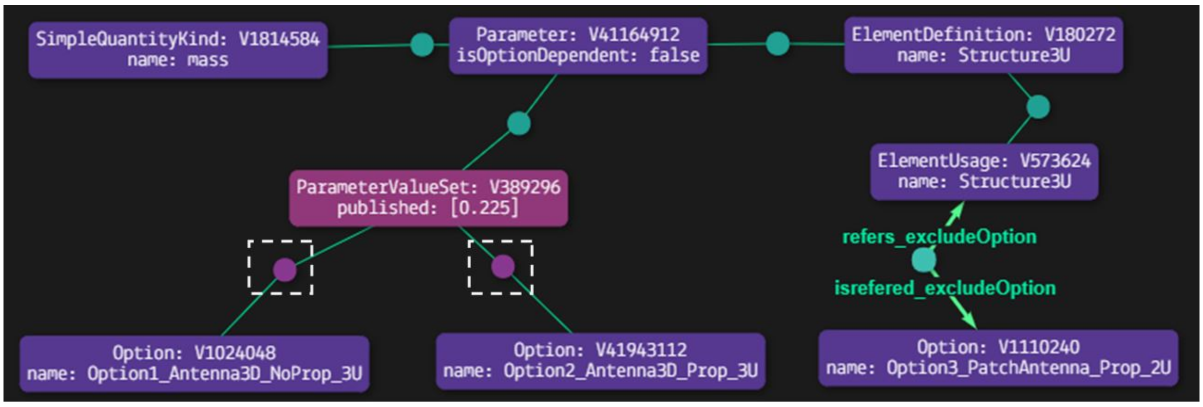


Figure 6.31: Inference outcomes from Rule 2 visualised with the TypeDB Workbase, the two inferred edges are denoted by purple circles and framed in manually added boxes.

Mass budgets:

The inference of these new relationships dramatically decreases the complexity of extracting the mass budget from the EMs. Via the TypeDB Python API, the values related to an option by an *includedInMassBudget* relationship are queried. The number of elements, scale, and mass margins associated with each value are also extracted from the graph with the TypeDB Python API.

Table 6.8 compares the mass budgets of STRATHcube and NEACORE manually computed at the time of the studies, with the budgets inferred from the KG. The slight dissimilarities observed mostly originate from missing mass margins in the original models. Following the ESA CE margin philosophy [16], a mass margin of 20% is assumed by default. However, during manual computation, discussions with the study participants often revealed that the actual mass margin was lower. This analysis also disclosed that the mass parameter of one equipment from STRATHcube's first design option was missing, contributing to the delta mass observed. In the case of the NEACORE's second design option, the comparison exposed an error in the manual computation, as some equipment had been wrongly incorporated into the option's budget. Removing these items from the manual computation decreased the mass delta to 0.3%.

Table 6.8: Comparison of mass budgets manually computed and inferred from the KG, per design option.

Satellite	**Design Option**	**Manual Computation [kg]**	**Inferred from Graph [kg]**	**Δ [%]**
STRATHcube	1	3.78	3.76	0.53
	2	5.03	5.06	-0.60
	3	3.17	3.20	-0.95
NEACORE	1	22.65	22.44	0.93
	2	21.27	20.65	2.91

6.6.2 Combination of NLP and KG: Heritage Analysis

Heritage analysis is one of the first tasks performed by system engineers during the pre-study phase. The goal is to identify past similar missions to help kick-off the new study, and support the initial parameters estimation. The application presented here combines KG and NLP.

Each of the iterations found in the KG is manually enriched with a metadata set as defined in Section 6.5.3. Since the metadata is defined at iteration level, the various design options do not impact the comparison. By extracting and analysing the metadata sets, a similarity score can be computed between each iteration. Iterations with a similarity score above a set threshold are deemed similar. This application is similar to the method presented in Section 5.3 where CDF feasibility study reports are compared with a doc2Vec model and a cosine similarity. The same approach will be used here to compare the metadata expressed as text.

Approach

The approach, summarised in Figure 6.32, can be decomposed in 4 steps: (i) the metadata sets of N iterations are extracted from the populated KG, (ii) each metadata set is compared to the remaining metadata sets either through a term matching or using the doc2vec model and cosine similarity, (iii) $(N*(N-1))/2$ similarity factors are computed from each iteration pair, (iv) $(N*(N-1))/2$ new relationships are inserted in the KG, linking the iterations. This new type of relation is named *Reference_SimilarityFactor*, and has an attribute *simFactor* corresponding to the similarity factor's value. The metadada sets are extracted via the TypeDB Python API. An iteration is not compared to itself.

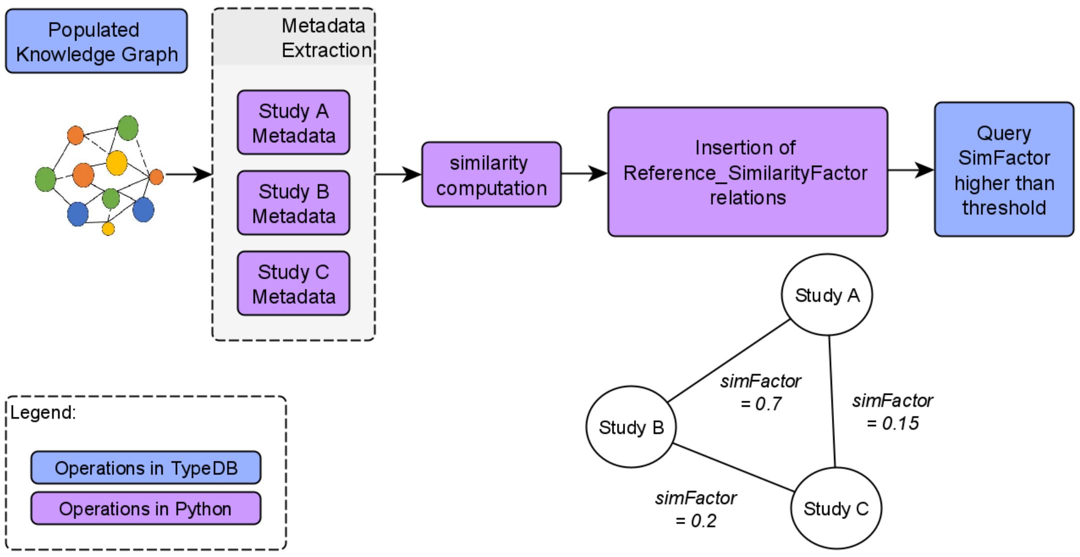


Figure 6.32: Approach to retrieve similar missions from the Knowledge Graph

Methodology

The global similarity factor is computed based on the metadata sets presented in Section 6.5.3. The metadata sets are assigned different weights to balance their contribution based on their relevance to the heritage analysis. For instance, similar *Mission Objectives*, a core parameter, should have a higher impact than similar *Launch Year*. A preliminary hierarchy of the metadata divided in 3 categories (High, Medium, Low) is suggested in Table 6.9. The global similarity score is the combination of each parameter comparison. These scores are combined in a weighted sum based on Equation 6.1.

$$F_{i,j} = \frac{\sum_{l=1}^{3}(\sum_{p=1}^{N_p} f_{p(i,j)} * w_l)}{\sum_{l=1}^{3} N_p * w_l} \tag{6.1}$$

Where $F_{i,j}$ the similarity factor between mission i and mission j, w_l the weight assigned to category l, N_p the number of metadata parameters in category l, and $f_{p(i,j)}$ the similarity of parameter p between mission i and j.

Table 6.9: Metadata Weights Distribution

Weight	**Metadata Type and examples**
High	Core information on the mission: scientific objectives, requirements, background, target system.
Medium	Information on mission design: orbit type, payloads, propulsion type, volume envelope
Low	Programmatics information: launch year, lifetime

Most of the metadata parameter is free text, and will therefore be compared with a doc2vec model and a cosine similarity as previously seen in Section 5.3. The same doc2vec model, trained with the ECSS requirements, is used to generate the representation vectors of the metadata textual parameters. The value of the similarity $f_{p(i,j)}$ will then correspond to the cosine similarity value between parameters' representation vectors. Metadata parameter with inputs limited to a set list, such as the orbit type, are compared with a simple term matching. If the two inputs are identical, then $f_{p(i,j)}$ is set to 1.

Results

The KG populated with the STRATHcube and NEACORE iterations is used again. However, a shallow iteration of a third CubeSat, QARMAN, is also migrated to the KG. QARMAN is a mission from the Von Karman Institute, its metadata set is populated with information found online in [237,238]. The final graph contains 123,896 nodes and edges (9.2% entities, 9.5% attributes, and 81.3% relations).

The QARMAN mission is selected due to its similarities with the STRATHcube mission. QARMAN is a 3U CubeSat deployed from the ISS in February 2020, and is the first CubeSat designed to survive atmospheric reentry. STRATHcube is also a 3U CubeSat, to be deployed from the ISS, with a 3D phased array antenna for space debris detection as primary payload. Its secondary objective is to perform measurements during re-entry using several heat flux/pressure sensors and UV/visual spectrometers. Neither mission has a propulsion system. On the other hand, NEACORE is a radically different mission. It is an interplanetary mission involving up to six 12U CubeSats. NEACORE studies Near-Earth Objects (NEOs) accommodating a camera and either a LIDAR or a spectrometer. The mission is expected to last between 3 and 6 years, with a low-thrust propulsion system. It is thus expected that the value of the similarity factor between STRATHcube and QARMAN is higher than between STRATHcube/NEACORE or QARMAN/NEACORE.

The metadata weights for the High, Medium and Low parameters are respectively set to 0.5, 0.35, and 0.15 after a grid search. As shown on Figure 6.33, three new *Reference_SimilarityFactor* relationships are instantiated with the similarity factors as attributes. The similarity factor between STRATHcube and QARMAN of 0.62 is significantly higher than the similarities between NEACORE and STRATHcube or NEACORE and QARMAN, respectively of 0.21 and 0.23.

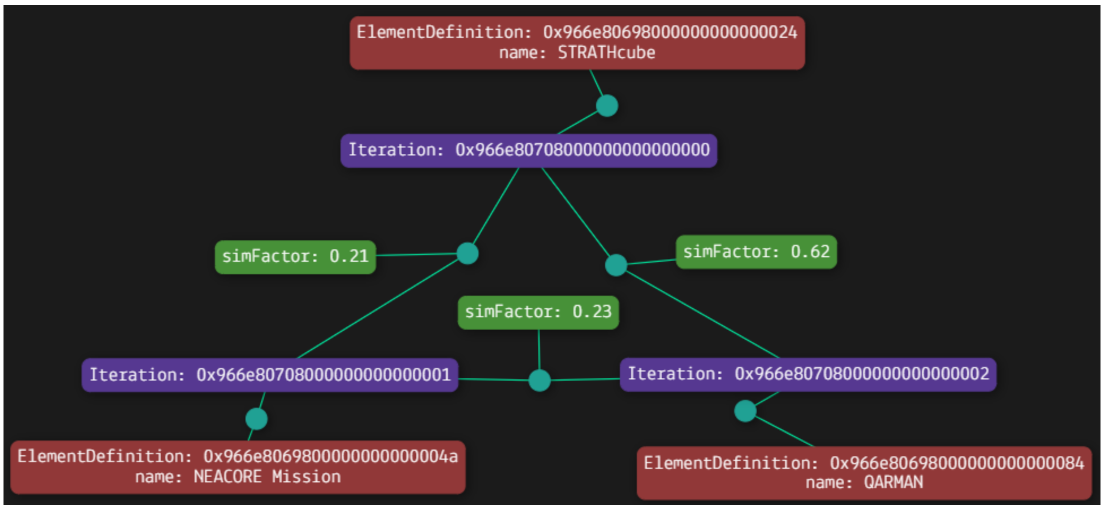

ElementDefinition: 0x966e806980000000000000024
name: STRATHcube
Iteration: 0x966e807080000000000000000
simFactor: 0.21
simFactor: 0.62
simFactor: 0.23
Iteration: 0x966e807080000000000000001
Iteration: 0x966e807080000000000000002
ElementDefinition: 0x966e80698000000000000004a
name: NEACORE Mission
ElementDefinition: 0x966e806980000000000000084
name: QARMAN

Figure 6.33: Visualisation of the similarity relations with the iterations in TypeDB Workbase

6.7 Discussion

The two applications implemented here gave only a glimpse of the potential for Knowledge Graph and text mining to support the concurrent engineering process. Automation of calculation and information retrieval tasks are the way to go to relieve the experts' workload. A future Concurrent Engineering tool, COMET, mentioned in Section 3.3, will automate budgets generation and allow basic rule-based inference. This tool, nor any other, does however not implement a NLP layer, which is key to analyse the unstructured text found in the models. These tools are also not yet capable of comparing models. On the other hand, merging the models into a Knowledge Graph gives a complete flexibility to perform inter and intra-models querying and inferring. In future work, additional rules could be defined to furthermore use the inference potential of the KG reasoner and extract additional insights from the models.

Only a global similarity score is computed in the second application. In future work, more refined scores, for certain parameters of interest or per subsystem, could be computed to provide a more detailed heritage analysis. A similar approach was implemented in Section 5.3, with a heritage analysis based on chapters from feasibility studies and the doc2vec model. An ideal solution would be to eventually merge all data sources related to a mission, including the reports, the models and online content, into a single Knowledge Graph to provide a complete profil of a mission design.

Merging heterogeneous data however first requires achieving semantic interoperability via a space systems ontology. Even merging only the feasibility reports with their respective Engineering Models semi-automatically is not straightforward. As was seen in Section 5.3, the reports do not necessarily follow the template and chapters' names vary as does the naming of equipment in the Engineering Models. For instance, a battery equipment could be named "*bat XYZ*" or "*XYZ battery*". Text mining methods could be once again the key to match similar concepts, by means of Topic Modelling or doc2vec embedding.

6.8 Chapter Summary

This chapter presented the building blocks to merge Engineering Models of space missions into a first Knowledge Graph of EMs. Two applications, an automatic mass budget generation and a heritage analysis, gave a glimpse into the potential of such data structure to support the early design process of space missions. A new migration pipeline was developed to migrate a UML class diagram to a TypeDB database. This chapter has successfully demonstrated the potential of combining KG technology and NLP to enhance the data linkage, reusability, and interpretability of EMs.

In the absence of a space systems ontology to structure the DEA's KG, this chapter explored a first application relying on the EM's conceptual model. The comparison of off-the-shelf KG tools as well as the first rule-based inference applications pave the way for the development of the DEA's KG and reasoner.

Chapter 7

Conclusions

This final chapter reviews the findings and contributions of this thesis, assessing how they address the aims and objectives. Current limitations of the methods applied are discussed. The chapter ends with recommendations for future work and a suggested architecture for a future Design Engineering Assistant.

7.1 Evaluation of Findings and Contributions

This thesis developed several building blocks for a DEA, while exploring how NLP and text mining methods could be tailored to the field of space systems to facilitate knowledge management and reuse at the early stages of space mission design. This work focused on the unstructured and semi-structured data used or generated during feasibility studies applying the Concurrent Engineering approach. Several core objectives were defined in Section 1.2, this section underlines how each of them were fulfilled by the work presented in this thesis.

Objective 1: Review the latest design methodologies and understand the needs of the experts involved in Concurrent Engineering design sessions.

The Literature Review of current design processes confirmed that the Concurrent Engineering approach is, nowadays, one of the most efficient design methods, enhancing the collaboration between subsystem experts and accelerating the convergence towards a sound system solution. 47 ESA experts were interviewed to compare the theory with the practice. The outputs of this survey largely contributed to quantifying, for the first time, the knowledge bottleneck faced by engineers at the early stages of design, and identifying the tasks that could be alleviated by applying NLP and text mining methods.

Objective 2: Build a first collection of unstructured and semi-structured data dedicated to space systems engineering.

A 15GB collection of texts including journal publications, books, reports and Wikipedia pages was collected. A novel NLP pipeline tailored to space systems by adhering to the European standards was developed with the NLTK Python library. The processed documents, with the exception of the ESA reports, constitute a first open-source collection of documents related to space systems. Two CubeSat Engineering Models were generated during internal design challenges led at the University of Strathclyde's Concurrent and Collaborative Design Studio. These two models, each including several design iterations and design options, form the semi-structured data set. This collection of texts form the foundations of the DEA Knowledge Base.

To fulfill objectives 3 and 4, several methods were adapted and trained, from Ontology Learning, Topic Modelling, word and document global embedding to contextualised embedding. Five new domain-specific models were trained on the corpus of unstructured data: (i) the Topic Modelling spaceLDA model, (ii) a word embedding word2vec model, and (iii) the SpaceTransformers family including SpaceBERT, SpaceRoBERTa, and SpaceSciBERT.

Objective 3: Develop statistical, embedding and contextualised methods adapted to space systems to semi-automatically generate the initials layers of a space systems ontology.

The Terms, Synonyms and Concepts layers of the Ontology Learning Layer Cake were, for the first time, semi-automatically extracted from a collection of texts related to space systems using statistical and word embedding methods. The potential of using the Layer Cake to lay the basis of a space systems engineering ontology, required to structure the DEA's KG, and enrich pre-existing domain-specific lexica was demonstrated. Novel domain-specific representation vectors for terms related to space systems were produced with the word2vec model. The SpaceBERT, SpaceRoBERTa and SpaceSciBERT models were further pre-trained from BERT, RoBERTa, and SciBERT on the unstructured corpus excluding the ESA reports. A thorough comparison of the performance of these domain-specific models with respect to their baseline models on a classification task was provided. The SpaceRoBERTa model considerably improved the results on the downstream Concept Recognition task for domain-specific language models.

Objective 4: Develop probabilistic and embedding methods adapted to space systems, and understand their scope of applicability within the Concurrent Engineering design process.

Regarding the automation of tasks usually manually done by systems engineers, two text mining methods were adapted to space systems, demonstrating the positive impact of these methods on the design process. A spaceLDA model was enriched with curated lexical priors and an optimised Weighted Sum. This new approach outperformed the unsupervised LDA model and a literature method for aggregating per-topic word distributions. The Topic Modelling model was successfully applied to a Requirements Management case study, identifying the topics of unseen design requirements. An extension of the word embedding approach, a doc2vec model, was trained to produce document representations of feasibility reports. The comparison of these representation vectors proved to be a new efficient way to perform instantaneous heritage analysis and recall past similar missions

Objective 5: Investigate methods to extract new insights from semi-structured data generated during Concurrent Engineering studies.

To exploit and compare design information stored in Engineering Models (EMs), the models were merged in a single database. Following a database trade-off, a Knowledge Graph (KG) was selected to host the models. A KG is defined as a graph database bearing semantics, and thus enabling reasoning and the inference of hidden knowledge. Novel migration pipelines were developed to enable the mapping of the UML-based models to the selected KG tool. Finally, the potential for enhancing feasibility studies with a KG was demonstrated through two case studies: (i) a mass budget, a task usually manually done by systems engineers, was automatically inferred for each design option, (ii) relying again on the document embedding approach, similar missions were identified based on a set of metadata. The latter case study showed the potential for building a NLP layer on top of a KG.

The work presented in this thesis has fulfilled the initial core objectives. It is one of the first attempt to bridge the gap between the worlds of space systems engineering and text mining, a bridge seldom crossed by previous research. Furthermore, the methodologies presented here could be applied to various engineering fields.

7.2 Limitations

The lexica semi-automatically generated and the topics discovered with spaceLDA are restricted by the size and content of the training corpus. Although various heterogeneous text sources were sought, it is complex to ensure that the resulting corpus does provide a complete overview of space systems. On the other hand, text collection can quickly turn into a never-ending task. The text collection in this thesis is broad enough to cover the fundamentals of space systems and includes the most common sources of information mentioned by experts. Access to data was one of the main challenges faced at the start of this thesis, data silos and the confidential nature of some design information restricted the amount of data that could be collected. Solutions were found to avoid some of these issues, using for instance a NDA and a remote ESA server to access the feasibility reports. The current Transfer Learning trend however gives hope that future models will not require large corpus of documents to be trained.

Access to Engineering Models was even more restricted. Furthermore they had to be based on the ECSS-E-TM-10-25A technical memorandum, meaning they had to be either generated with ESA's OCDT or RHEA's CDP4. Only two models, for the NEACORE and STRATHcube CubeSats, were available following internal Concurrent Engineering studies. It may be argued that this is a small library of models, but the generation of each model required substantial effort. The NEACORE and STRATHcube studies respectively involved teams of 17 and 29 participants, master and PhD students from the University of Strathclyde, who designed each mission over the span of two full week design challenges. Each model contains several iterations and design options which still enable intra-model inference. For the heritage analysis case study, a shallow model for the QARMAN CubeSat was manually created. The purpose of this study was, nevertheless, to showcase a new methodology for EMs comparison rather than performing a through analysis between models. In future work, additional models could be derived from feasibility studies reports.

Limitation in computational power did not allow to train the SpaceTransformer models from scratch. Doing so could have enabled the comparison of the further pre-trained models with not only their baseline but also with pre-trained versions. This approach was however not considered the wisest, as the baseline models are trained on very large corpus with extensive computational power. The size of the text collection used as training corpus, 15GB, represents 10% of the dataset used to train RoBERTa. The text collection size is similar to BERT's (16GB), but the latter model was trained over 4 days on 4 cloud TPUs. As mentioned in the above paragraph, while access to very large text datasets or computational power is limited, the sensible approach is to capitalise on the baseline models rather than reproduce their training.

The different text sources were treated with the same level of confidence for the syntactical and embedding approaches. In both cases, the reliability of the texts content was not crucial as the training goals were to learn the structure of the domain-specific language and extract frequent words. For other applications, for instance, Question Answering or Document Summarisation, coefficients such as Certainty Factors [5] could be introduced to reduce the influence of less reliable sources. Such coefficients could have been used for the heritage analysis case studies, to reflect for instance, the impact of technology obsolescence and diminish the similarities with outdated missions.

7.3 Future Research Directions

A first step would be to merge the unstructured and semi-structured data which were examined separately in this thesis. This task is challenging as the conceptual model of the semi-structured data has to be mapped to the semantics of the texts. To implement semantic interoperability, an ontology of space systems is essential. The development of a standardised European ontology is currently ongoing under the supervision of ESA. Its release will enable the restructuring of all data with respect to a single conceptual model, allowing to merge various data sources and types.

The NLP community is still evaluating the impacts of the recent Transfer Learning and Transformers revolution. This thesis only studied one downstream application, Concept Recognition, fine-tuning the domain-specific models. Additional downstream tasks, such as Relation Extraction, Question Answering or text similarity analysis could be examined in future work. The spaceTransformer models will considerably facilitate future domain-specific fine-tuning. Most available language models are trained on English text. Several research teams have attempted to develop models in other languages such as CamemBERT (French) [239] or AlBERTo (Italian) [240]. A durable solution could come from so-called multilingual models. These models can transfer knowledge across languages as, for instance, Google AI's MUM, standing for Multitask Unified Model, trained across 75 different languages [241]. This could help increase the diversity of text sources, steering towards an international perspective on space systems. To furthermore increase the diversity of knowledge sources, additional data types could be integrated to the models. For instance, a multimodal model combining computer vision and text mining could retrieve information from figures found in the documents. Speech To Text methods could transcribe discussions recorded during feasibility studies, and thus provide additional insights on trade-off arguments which are not always mentioned in the study report.

As underlined by the survey results presented in Section 2.2, experts expressed their reservation with respect to these new methods. AI is often seen as a *black box*, where the inputs and outputs are known but the calculation process itself remains obscure.

Interacting with an incomprehensible system can create discomfort and distrust, which is problematic for AI systems involved in decision making. The implementation of Explainable AI, XAI, in future work should contribute to easing this defiance. XAI increases the transparency of results provided by Machine Learning algorithms, so that they can be interpreted by humans [242].

7.4 Notes on the suggested DEA architecture

The architecture presented in Figure 1.1 does not explicitly integrate the means to quantify and mitigate the uncertainties related to the knowledge extraction. These uncertainties may arise from the reliability of the data sources, the incompleteness or conflict of data stored in the Knowledge Base, the vagueness of the user query, or the ambiguities of natural language itself. Methods such as Certainty Factors should be investigated to diminish the impact of these uncertainties.

This thesis focused on explicit knowledge stored as unstructured and semi-structured data. However, the experience and instincts built by experts during their career constitute precious know-hows that would be worth integrating in the DEA's Knowledge Base. It is thus recommended to build a feedback loop, allowing the virtual assistant to continuously evolve and learn. For example, a user might want to update the launch date of a spacecraft or the specifications of an equipment. A basic feedback could be based on a grading system where the user would rate the quality of an output. With an NLP layer, the user could also add a written comment to complete or append the information provided by the DEA. For instance, both Wolfram Alpha [1] and Google allow their users to comment on a query output. The design of the feedback loop must however be done with great care as it might trigger the injection of additional uncertainties and subjective inputs.

Last but not least, an additional recommendation collected during this thesis and through discussions with experts, would be to ensure that the tool remains easy to use. The DEA should not increase the workload of the experts and remain as low-maintenance as possible, with interactions done in natural language.

***

[1] `https://www.wolframalpha.com/`

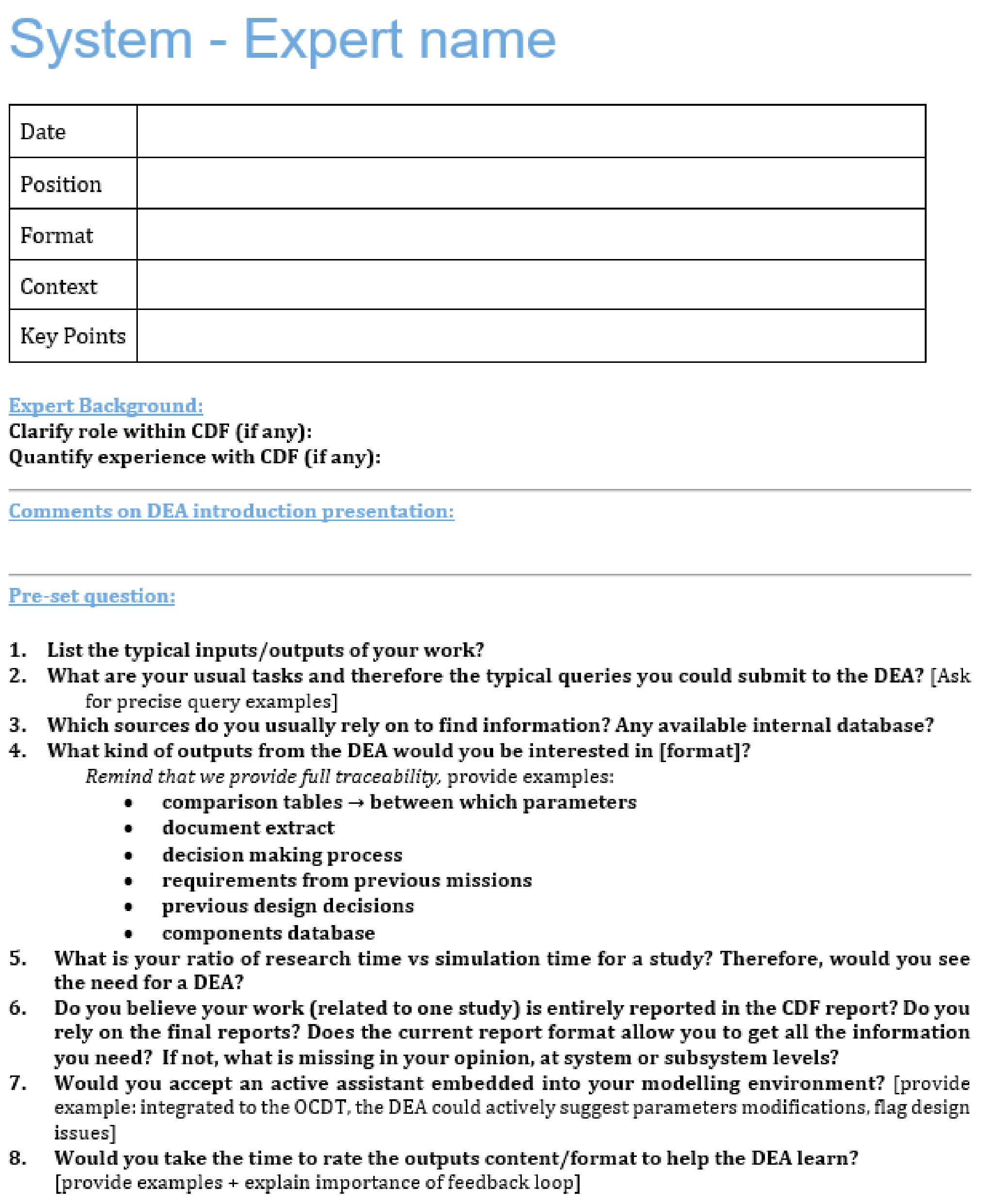
Chapter 8

Appendices

8.1 Appendix 1: Survey Templates

Table 8.1: Survey with ESA expert: questions submitted to experts during the round-table

Section	**Question**	**Format**	**Objective**
Intro	*When we say AI for Space Mission Design, you think:*	Word Cloud	Understand bias or preconceived ideas on the topic, take pulse of audience
Part I: Work habits	*Compared to a study length how long to you spend researching through available information?*	Single answer	Estimate the time spent by experts in information retrieval tasks
	Where do you find the most useful information for your studies?	Ranking	Identify the experts' main sources of information
	Would you rely on an AI to mine information for you?	Single answer	Estimate level of thrust into AI systems
Part II: Human Machine Interaction	*Rate the most useful outputs*	Rank	Understand which outputs content the Users are the most interested in.
	Pick your preferred output format	Rank	Orientate how the results will be displayed via the User interface.
	Pick your favourite interface	Multiple answers	Discuss the integration of the tool as a plug-in into a modelling environment or a knowledge-sharing platform.
	If you could provide feedback on an output, which format would you prefer?	Single answer	Understand how to build a feedback loop
	Which documents and/or database would you like to see included in a Knowledge Graph?	Open answer	Inquire about other data sources



System - Expert name

Date	
Position	
Format	
Context	
Key Points	

Expert Background:
Clarify role within CDF (if any):
Quantify experience with CDF (if any):

Comments on DEA introduction presentation:

Pre-set question:

1. **List the typical inputs/outputs of your work?**
2. **What are your usual tasks and therefore the typical queries you could submit to the DEA?** [Ask for precise query examples]
3. **Which sources do you usually rely on to find information? Any available internal database?**
4. **What kind of outputs from the DEA would you be interested in [format]?**
   Remind that we provide full traceability, provide examples:
   - **comparison tables → between which parameters**
   - **document extract**
   - **decision making process**
   - **requirements from previous missions**
   - **previous design decisions**
   - **components database**
5. **What is your ratio of research time vs simulation time for a study? Therefore, would you see the need for a DEA?**
6. **Do you believe your work (related to one study) is entirely reported in the CDF report? Do you rely on the final reports? Does the current report format allow you to get all the information you need? If not, what is missing in your opinion, at system or subsystem levels?**
7. **Would you accept an active assistant embedded into your modelling environment?** [provide example: integrated to the OCDT, the DEA could actively suggest parameters modifications, flag design issues]
8. **Would you take the time to rate the outputs content/format to help the DEA learn?** [provide examples + explain importance of feedback loop]

Figure 8.1: Interview template

8.2 Appendix 2: Supplementary Background

8.2.1 Artificial Intelligence

Defining AI

AI is not easily restricted to a single definition. In the fundamental textbook of Russel and Norvig [55], several definitions are organised based on four different goals: (i) Thinking Humanly (ii) Acting Humanly (iii) Thinking Rationally, and (iv) Acting Rationally. The *Thinking Humanly* approach is linked to cognitive modeling where an AI system is not based on executing a program but achieving goals. The *Thinking Rationally* approach is on the contrary based on rule and inference from facts. The *Acting Humanly* approach is associated with an AI system capable of mimicking Human interactions and thinking. Finally, the *Acting Rationally* is based on a more logical approach, a “sense-plan-act” execution through logic. A satisfying short definition is perhaps provided by American futurist Raymond Kurzweil in [243] defining AI as

> The art of creating machines that perform functions that require intelligence when performed by people.

A fundamental distinction is usually made between strong (general) and weak (narrow) AI. Strong AI refers to AI exhibiting human-level intelligence, able to autonomously reason, continuously learn and act as a human [3]. Weak AI can only handle one task at a time, and only act as if it is intelligent [55]. At the time of writing, we have not yet achieved Artificial General Intelligence (AGI).

A brief history of AI

The field of Artificial Intelligence research was founded during a 1956 summer conference at Dartmouth University hosted by John McCarthy and Marvin Minsky. By gathering ten researchers for a two-month workshop, they hoped to initiate a collaborative effort and significantly advance the development of AI. During the event, McCarthy is said to have coined the term of *Artificial Intelligence.* Ideas behind AI were not new but the workshop provided a strong common foundation for future research. Already in the 1940s, the Canadian psychologist Donald Hebb replicated the process of neurons in the human brain, his research paving the way for Artificial Neural Networks (ANNs). However at the time, computers did not have the sufficient processing power to sustain these networks [3]. In 1950, Alan Turing presented a test to identify intelligence in a system [244]. In his test, a human questions two terminals, one operated by a machine and another by a human. If the human cannot distinguish the answers provided by the machine from the human's, then the machine has passed the test. The Turing test is still today a reference for evaluating algorithms.

Following the 1956 conference, interest for AI research increased. Early models could prove mathematical theorems, solve calculus problems or even impersonate a psychotherapist [245] [246]. The performances of these early programs, partly restricted by hardware limitations in terms of memory capacity and processor speed, failed to meet the expectations and led to a first winter of AI lasting from 1974 to 1980 [3]. This so-called "AI-winter" corresponds to a period of decrease in funding and slower development. In the 1980s, Japan massively invested in AI and several countries followed their lead [246]. Expert systems became the symbol of the AI renewal. These systems mimic experts reasoning based on rules formalising human thinking as set of "if-then" statements [3]. Unfortunately the growth of these systems once again hit a hardware limit, and a second AI winter descended on the AI community from 1987 to 1994. By the mid-1990s, the growth of computer processing with, notably, the introduction of the first GPUs, along with the development of the internet and the web allowing large data collections, "Big Data", gave another chance to AI. De Mauro et al. in [4] provide one of the first formal definition of "Big Data":

> Big Data is the Information asset characterized by such a High Volume, Velocity and Variety to require specific Technology and Analytical Methods for its transformation into Value.

The growth of AI has since been continuous. Iconic milestones were reached in 1997 when a chess grand master was defeated by IBM's Deep Blue [247], in 2011 when IBM's Watson won the quiz game "Jeopardy" [248], and in 2015 when Google's AlphaGo beat the Chinese Go champion Ke Jie [249].

AI Classification and branches

Methods of AI can be classified in two categories: symbolic and statistical approaches. Symbolic approaches are based on rules and inference. The early history of AI was dominated by symbolic approaches, expert systems being an example of symbolic models. Statistical approaches such as Machine Learning are data-based. The development of statistical approaches was made possible with the development of Big Data and growth in processing power. Today the field of AI is divided in several branches.

Natural Language Processing, the field most relevant to this thesis, is extensively introduced in Section 2.3. This thesis will also leverage several Machine Learning (ML) methods. The latter approaches build a statistical model based on a data set. The learning can be supervised, semi-supervised, unsupervised or reinforced [250]. In supervised learning, the data set is a collection of labelled examples. Supervised training is usually applied to classification and regression problems. With unsupervised learning, the data set is a collection of unlabelled examples. Clustering is a well-known application of unsupervised training where unlabelled data with similar features are clustered. When the data contains both labelled and unlabelled examples, then the training is call semi-supervised. Finally, in reinforcement learning, the models learns to achieve a certain goal based on rewards earned when the optimization is correct. If an algorithm was tasked with the identification of cats in a picture, with supervised learning, the algorithm is given a data set of cat pictures and learns the features of a cat. With unsupervised learning the algorithm is given a data set of animal pictures, and has to learn on its own the different features of each animal. With reinforcement learning the algorithm guesses the features of a cat, when it is right it is rewarded, and fine-tunes its features.

Speech to text and text to speech are the main applications of the Speech recognition field. Speech recognition is a key enabler for digital assistants as Apple's Siri or Amazon's Alexa, able to understand voice command and synthesise vocal answers [251]. The computer vision field focuses on image recognition and machine vision. The development of the latter has been a key enabler to the fields of robotics increasing the autonomy of systems. Matthies et al., [252], summarises how progress in computer vision algorithms increased the performances of the NASA/JPL Mars Exploration Rover (MER). Rule-based experts corresponds to expert systems previously introduced.

8.2.2 Word embedding

The concept of word embedding appeared in the 1950s based on the idea that context plays a significant role in the meaning of a term. One of the founding figure of this school of thoughts, J.R. Firth, is known for declaring that, *"You shall know a word by the company it keeps"* [253]. There are several approaches to word embedding. The most basic method is One-Hot Encoding (OHE), where each word is represented by a sparse vector of 0s and 1 as shown on Figure 8.2. OHE however contains poor information and only support equality testing. Previous classic approaches such as the Hyperspace Analogue to Language (HAL) method and Latent Semantic Analysis (LSA) are based on counting co-occurring words. The HAL model, presented in [254], captures the statistical dependencies between words by counting their co-occurrence in a context window. LSA is the precursor of topic models and LDA addressed in a later section. These set of methods were outperformed in the mid-2000s by neural network-based methods, which "predicted" rather than "counted". As underlined in [255] and [256], these neural network-based methods outperformed the previous co-occurrence-based approaches.

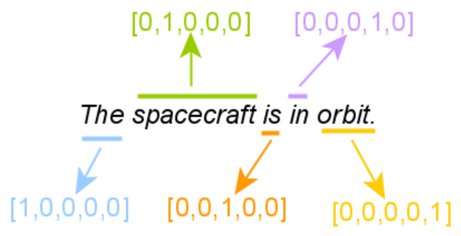


Figure 8.2: Example of One-Hot Encoding

A first major contribution to the predictive methods is the work of Bengio et al., in [124], where the authors refer to word embedding as *distributed representations of words*. The authors present a feed-forward neural network language model (FNNLM), with a linear projection layer and a non-linear hidden layer to train a statistical language model. A language model learns the probability of a sequence of words in a language. The model of Bergio et al. laid the foundation of future word embedding models. Its general building blocks are (i) an embedding layer, also called projection layer, which generates a word embedding by multiplying an index vector with a word embedding matrix, (ii) a hidden layer producing an intermediate representation of the input, and (iii) a softmax layer which produces the output probability distribution. This architecture is summarised in Figure 8.3.

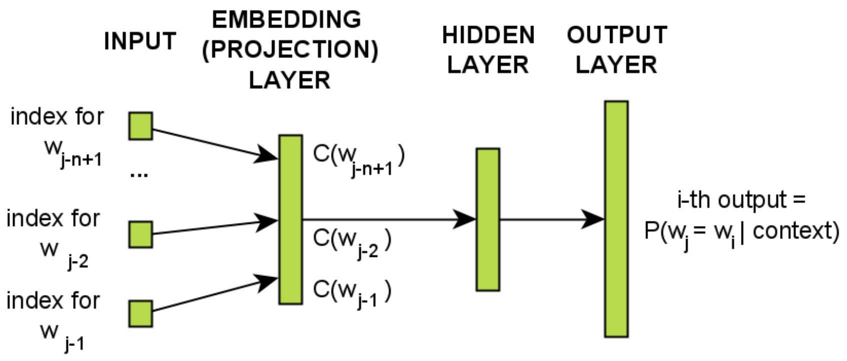


Figure 8.3: Simplified view of a neural network language model where $C(w_i)$ is the i-th word feature vector and $P(w_j = w_i)$ is the output probability of target word w_i being the term w_j based on the previous words. Derived from [124].

While the word embedding in Bengio et al. was a by-product of the language model training, Collobert and Weston in [257] focused on the word representations and demonstrated how the embedding themselves could significantly improve NLP downstream applications such as synonyms predictions, NER or POS tagging. Both approaches from Bengio et al. and Collobert and Weston are however limited by the high computational cost of their hidden and softmax layers. Word embedding really took off when in 2013, Mikolov and his team from Google introduced two novel model architectures, known as word2vec, for generating continuous vector representations of word [123]. With these new architectures, the authors intended to reduce the computational complexity by removing the hidden layer and optimising the softmax layer.

A year after the release of word2vec, a team of researchers from Stanford presented GloVE [258]. The GloVE method follows the count-based rather than predictive approach by factorising a large co-occurence matrix to achieve lower-dimension representation, addressing a pitfall of previous counting methods. A couple of years later, a team from Facebook (including Mikolov) developed fastText [259]. FastText reinterprets the word2vec approach by using sub-words and character levels embedding rather than words. The lower-level embeddings are then combined to form the word embeddings. Word2vec, GloVe and fastText are currently the main embedding methods. The authors of [260] and [127] respectively compare word2vec and GloVE on the generation of word embeddings for the Italian language and for legal Polish terms. The authors agree that word2vec outperforms GloVe for their applications. In more recent work, Sutton and Cristianini compare word2vec, GloVe, and fastText [261] on a concept identification task. Their comparison demonstrates that fastText then GloVe achieve the highest ranking on their training sets. However the authors admit to using different general corpora for the training of the different methods which might have impacted the results. Finally, Yeganova et al [262] also compare these three main embedding methods on a synonym extraction task in the biomedical field. The authors concluded that word2vec model had the highest results for the discovery of synonyms based on the same stem, for instance, *launch/launcher*, while fastText showed better results with different stems synonyms, for instance, *astronaut/cosmonaut*. Predictive approaches such as word2vec and fastText currently seem to have more momentum than counting approaches such as GloVe.

8.2.3 Language Models

The key takeaways presented in the background section are:

1. Attention amplifies the signal from relevant parts of the input sequence, allowing the model to focus on different components of an input sequence to build a word embedding.
2. A Transformer is a stack of encoders followed by a stack of decoders including attention mechanisms.
3. Contextualised embedding is a product of the encoding layer.

4. The BERT architecture is based on the encoder part of the Transformer.
5. Transfer Learning, pre-training followed by fine-tuning, achieves high results on supervised downstream tasks trained with small data set.

Precursor models

The Embeddings from Language Models (ELMo) model proposed by Peters et al. in [263] is seen as the first breakthrough in the generation of contextualised embedding. The vector representations are derived from a bi-directional Long-Short Term Memory (LSTM) neural network model trained on a language modeling task. The bi-directional characteristic of the model ensures that it takes into account both previous and future words. The LSTM is a type of Recurrent Neural Network (RNN). The latter uses an internal state memory to process sequences of inputs. The LSTM can handle longer sequences than RNN. However the computation in a LSTM is linear, unlike the Transformer approach which enables parallelisation and thus a significant decrease in computational time as will be seen in the next section. As underlined by Liu et al. [264], ELMo concatenate the forward and backward LSTMs outputs, independently trained, thus disregarding the interaction between the left and right contexts. On the other hand, BERT is truly bi-directional as its representations take into consideration both sides of the context. Howard and Ruder proposed the Universal Language Model Fine-turning (ULMFit) in [138], unlocking the keys to Transfer Learning for NLP applications. The UMLFit architecture is based on a bi-directional LSTM as ELMo. Contextualised embedding and Transfer Learning however really took off with the adoption of the Transformer architecture and the release of the BERT model in 2018.

Attention mechanism and Transformer architecture

The Transformer architecture was first introduced by Vaswani et al. in the paper "*Attention Is All You Need*" [134]. In this fundamental work, the authors suggested a new architecture for machine translation, an application of sequence-to-sequence models, solely based on attention. Sequence-to-sequence (Seq2Seq) models map an input sequence to an output sequence, and are generally based on an encoder and a decoder combination. Machine translation, speech recognition, and text summarisation are common applications of Seq2Seq models. The Transformer model of Vaswani et al. outperformed classic approaches based on RNNs, LSTMs, and Convolutional Neural Networks (CNNs), establishing new SOTA results on machine translation tasks.

The Transformer architecture as defined by Vaswani et al. is shown on Figure 8.4. The encoder blocks (on the left) each include a multi-head attention sub-layer followed by a Feed-Forward Neural Network. This background focuses on the encoder architecture as it is the backbone of the BERT model. The initial input is a list of word embeddings from the input sequence. Each word embedding flows through the encoders independently and in parallel, the output of an encoder block being the input of the next encoder block. Layer by layer, the encoder stacks build unique contextualised representation of the input tokens. The original Transformer architecture includes 6 encoders and 6 decoders layers, with a hidden dimension of 512. The positional encoding applied to the input embedding enables the model to learn the words order.

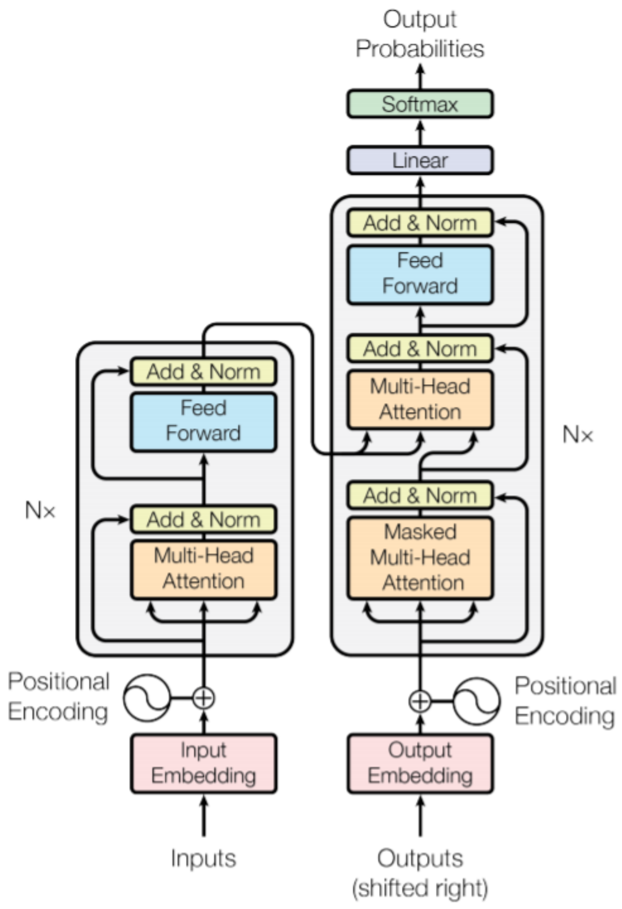


Figure 8.4: The Transformer Architecture as seen in [134]. N indicates the number of encoding and decoding stacks.

Attention is the key to the Transformer architecture success. Bahdanau et al. [265] and Luong et al. [266] pioneered the use of attention for Neural Machine Translation (NMT). Attention amplifies the signal from relevant parts of the input sequence, allowing the model to focus on different components of an input sequence. An attention head will capture a particular type of relationship between words, for instance neighbouring words or *subject-object* relationships. With several attention heads, it is possible to track several features simultaneously, and influence the embedding of a word. Vig introduced in [267] an open-source tool to visualise attention. Figure 8.5 displays the outputs obtained with this tool for different attention heads and encoding blocks in a

BERT model. The basic BERT architecture stacks 12 encoder blocks with 12 attention heads each. Two sentences are encoded: "*The spacecraft was launched last month. It is now orbiting the moon*". The *[CLS]* and *[SEP]* are special tokens added by the BERT tokenizer. The attention is represented as lines connecting the tokens, with the line thickness reflecting the value of the attention score. Each head is identified by a different colour. Sub-figure *(a)* displays all the attention scores of the first encoding layer. Sub-figure *(b)* illustrates how for the token *launched* the attention head in the first layer focuses on the neighbouring words. On the other hand, in sub-figure *(c)* and *(d)*, two attention heads of layer 2 respectively seem to focus on a *subject-verb* relationship and on a *noun-pronoun* relationship.

The above paragraph provides a conceptual definition of attention. Vaswani et al. [134] define the output of the attention head as function of a scaled dot-product as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{8.1}$$

Where Q, K, and V are respectively the Query, Key and Values, and d_k is the dimension of the Key matrix. Each row of these matrices, learned during training, contains the query vector $\mathbf{q}$, a key vector $\mathbf{k}$ and to a value vector $\mathbf{v}$ of an input word embedding. The query/key/value combination is comparable to a retrieval system. The query corresponds to the input embedding to be enhanced with attention and the keys map to the embeddings of all other words in the context window. As explained by Vaswani et al. [134], the output is then computed as the weighted sum of the values, where the values are paired with the keys and the weights are a combination of the query and keys.

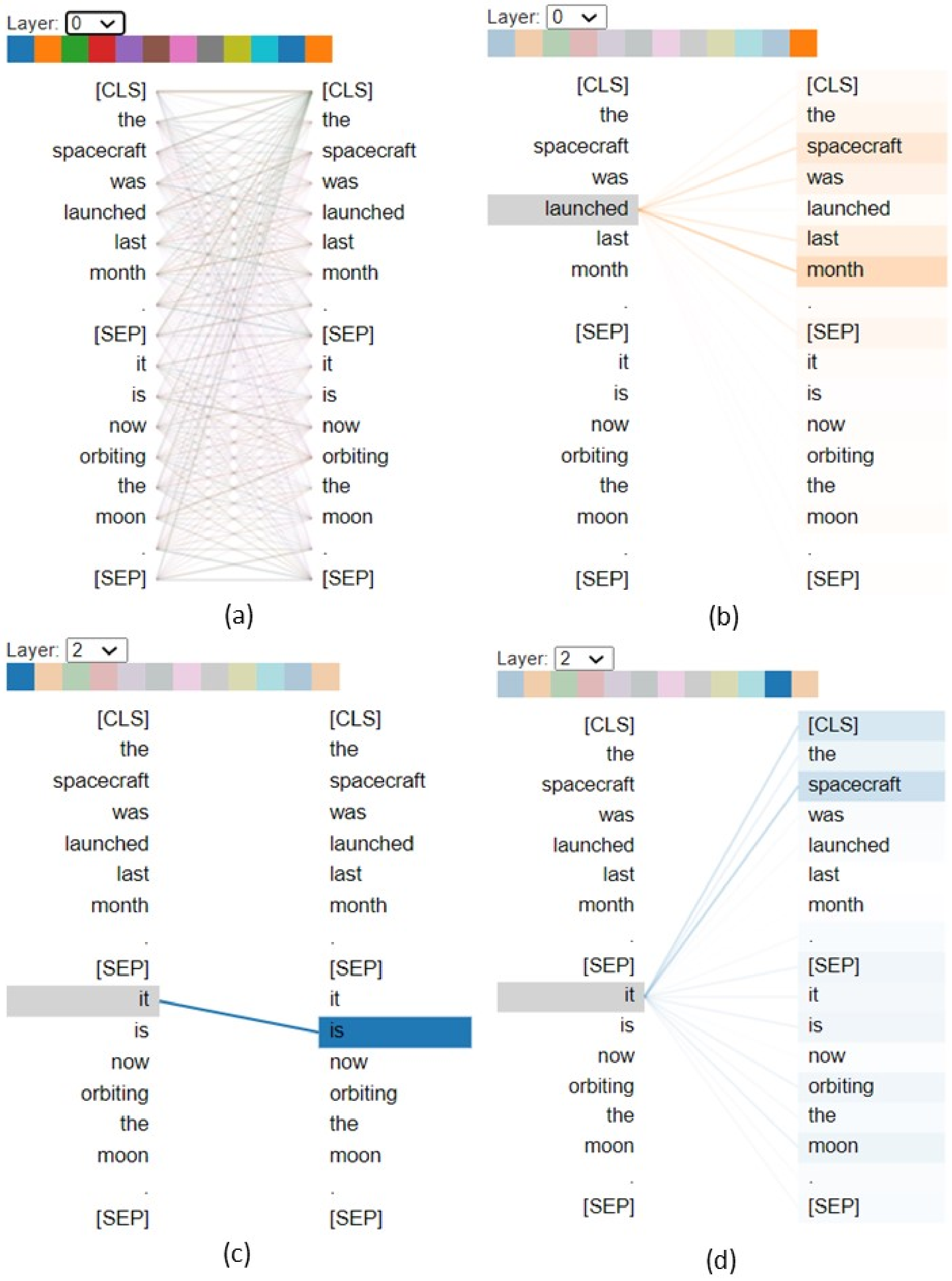

Layer: 0
[CLS] the spacecraft was launched last month . [SEP] it is now orbiting the moon . [SEP]
(a)
Layer: 0
(b)
Layer: 2
(c)
Layer: 2
(d)

Figure 8.5: Examples of attention visualisation generated with the open-source tool from [267]

Figure 8.6 simplifies the attention score calculation for a word w. First, the attention scores are computed between the query vector of the target word and the key vectors of the remaining words. Dividing by the square root of the Key matrix dimension increases the gradient stability [268]. The softmax function normalises the score and ensure that they sum to 1. Then, the Value vector is multiplied by the attention score. Words with high attention scores are thus emphasised. The Value vectors are then summed in a weighted sum, producing the output of the attention layer for the embedding x_1. This type of attention is called "self-attention" as the attention is calculated with respect to one input.

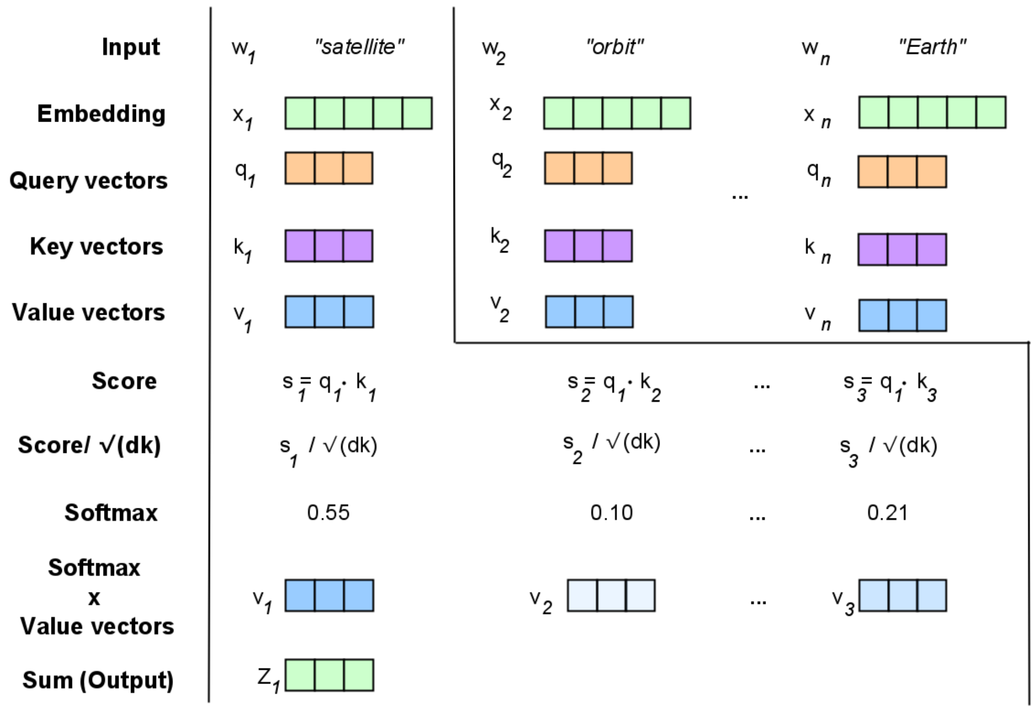


Figure 8.6: Illustration of the attention score calculation for an input embedding w_1, derived from [268]

Vaswani et al. use several attention heads in parallel, hence the name of "*Multi-head Attention*", in each encoder sub-layer. More precisely, they use 8 attention heads, and thus obtain 8 output vectors z for each input embedding. These are concatenated into a single output as the feed-forward sub-layer accepts a single input. The parallelisation in the Transformer significantly decreases the training time.

Pre-trained Language Models

In 2018, Delvin et al. from Google AI Language released the BERT model [77], a multi-layer bidirectional Transformer encoder based on the work of Vaswani et al. [134]. BERT comes in two flavours, BERT-Base and BERT-Large. The configuration of BERT-Base is based on a transformer architecture with 12 encoder layers, each using 12 self-attention heads and hidden dimension of 768. BERT-Large uses 24 encoder blocks, each with 16 attention heads and hidden dimension of 1024. The models respectively have 110 and 340 million parameters.

BERT was trained on two pre-training objectives, Masked Language Model (MLM) and Next Sentence Prediction (NSP), and on a general corpus of 3.3B words. With the MLM language modeling training task, 15% of the input sequence are randomly selected. 80% of these chosen token are replaced by a $< MASK >$ token, 10% by a random token, and 10% are unchanged. The model is then trained to correctly predict the hidden token. The 15% threshold is suggested by Delvin et al. The approach is, broadly speaking, comparable to the target word prediction in word2vec. In the NSP task, the model is trained to predict if two sentences are related, meaning if one sentence is likely to come after the other. The model chooses two sentences A and B, 50% of the time, the sentence B actually follows A, and 50% of the time it is a sentence randomly selected. This training task is notably useful for Question Answering (QA) applications. The training set for these two tasks are directly generated from the unlabelled corpus. When the model randomly masks a token or chooses a sentence, it does not require a labelled corpus, it independently generates a labelled data set to train on. Yann Le Cun, a major figure in the field of Deep Learning, named this type of training self-supervised learning (SSL). In [269], Le Cun and Misra assert that SSL could be the key to bringing AI closer to human-level intelligence as it removes the obstacle of labelled data set and is more similar to human-like learning.

According to Delangue, CEO of the HuggingFace library[1], speaking at 2020 NLP Summit[2], the advent of BERT is a conjuncture of three trends: increased computational power, large available text data set, and Transfer Learning, all allowing Deep Learning for NLP to work. The release of BERT inspired several other models based on a similar architecture or further pre-trained from BERT. Liu et al. from Facebook AI released RoBERTa standing for a Robusty Optimised BERT Pretraining Approach in 2019 [135]. RoBERTa is based on the configurations of BERT-Base and BERT-Large. However it outperformed BERT with a larger corpus of 160 GB, a longer training, a pre-training solely based on MLM, and larger batch sizes of 8k. SciBERT [136] was trained with the same configuration and size as BERT-Base on a scientific corpus of 1.14M papers, mostly focused on the biomedical field. Several variants of the SciBERT model were trained, either further pre-trained or trained from scratch. The SciBERT models were found to outperform BERT-Base on domain-specific tasks.

Finally, the Generative Pre-trained Transformer (GPT) models of OpenAI [78, 270, 271] are based on the decoder stacks of the Transformer architecture. The GPT-3 model has 96 layers with 96 attention heads, with an embedding size of 12,888 resuling in 175 billion parameters. This model is trained on a general corpus with 499 billion tokens on a cluster of V100 GPUs. GPT-3 achieved strong performance in several NLP tasks such as text summarisation and generation, with little to no supervised fine-tuning (zero-shot, one-shot and few-shot settings). The high performances of the model actually led OpenAI to not release the GPT open-source to prevent malicious usage. GPT-3 is still commercially available while BERT is open-source and has thus acquired more momentum in the academic community. Table 8.2 summarises the training configuration for the BERT, RoBERTa, and GTP-3 largest model.

[1]`https://huggingface.co/`
[2]`https://www.nlpsummit.org/`

Table 8.2: Transformer-based Models Parameters Summary

Model	BERT Base	BERT Large	RoBERTa Base	RoBERTa Large	GPT-3
Hidden units	768	1,024	768	1,024	12,888
Number of layers	12	24	12	24	96
Attention heads per layer	12	16	12	16	96
Number of parameters	110M	340M	125M	355M	175B
Corpus Size	16 GB		160 GB		499B tokens

To compare models with a similar configuration and size, the BERT-Base, RoBERTa-Base and SciBERT models are selected as foundations to further pre-train the domain-specific models presented in this thesis.

Subwords tokenization

Word-level tokenization has, so far, been the only type of tokenization discussed in this thesis. As was hinted to in Section 4.4.1 with the fastText model, character and subword tokenizers have been raising some interest in the NLP field. Word-level tokenizers usually assign a unique id to each terms found in a corpus vocabulary. Rare words that are not part of the tokenizer's vocabulary are thus mapped to an *OOV* (Out-Of-Vocabulary) id. Large training corpus yield high vocabulary sizes increasing the computational cost. Character level tokenization attempts to by-pass this issue by splitting terms into characters, thus dramatically reducing the tokenizer vocabulary, and being able to reproduce any words. However, character level tokenization significantly increases the length of input sequences, and does not record semantics as single characters carry few to no meaning. Subword tokenization is the compromise between these two approaches. With this method infrequent words are split into smaller common subwords. For instance, the word *highest* would be divided into two subwords, *high* and *est*. The tokenization of the word *higher* would then involve the same first subword entity *high* and a new entity *er*. Thus the number of OOV is dramatically reduced, as the subwords allow several word combinations.

Both BERT and RoBERTa rely on subword tokenization. As described by Delvin et al., BERT implements the WordPiece embeddings by Wu et al. [137] with a 30K subwords dictionary. As shown in Table 8.3, the WordPiece tokenizer adds two special tokens to a sequence, $[CLS]$ and $[SEP]$. The first special token is a classification token for the NSP training task. The $[SEP]$ token separates sentences. In the example, two words, *redefined* and *proportionally* are divided into subwords. The "##" indicates that a subword should be attached to the previous token.

The RoBERTa model from Liu et al. uses a byte-level Byte-Pair Encoding (BPE) tokenizer proposed by Radford et al. [271], based on the work from Sennrich et al. in [272]. Instead of using "##" to indicate that two subwords follow each other, BPE relies on a special character, $\dot{G}$, to indicated the start of a new word. As shown in Table 8.3, the BPE tokenizer does not have a classification token, as RoBERTa is only trained on a MLM objective. Instead, the tokenizer use a special symbole, $< s >$, to mark the start and the end of a sentence. The BPE tokenize has a larger vocabulary than BERT, around 50K subword units. According to Liu et al. [76], early work did not reveal any significant differences between these two approaches.

Table 8.3: Example of tokenization outputs at word and subword levels

Input	The solar panel area is redefined proportionally to the incident angle.
Word Tokenization	'the', 'solar', 'panel', 'area', 'is', 'redefined', 'proportionally', 'to', 'the', 'incidence', 'angle', '.'
WordPiece	'[CLS]', 'the', 'solar', 'panel', 'area', 'is', 'red', '##efined', 'proportional', '##ly', 'to', 'the', 'incidence', 'angle', '.', '[SEP]'
BPE	'$< s >$', 'The', '$\dot{G}$solar', '$\dot{G}$panel', '$\dot{G}$area', '$\dot{G}$is', '$\dot{G}$red', 'efined', '$\dot{G}$proportion', 'ally', '$\dot{G}$to', '$\dot{G}$the', '$\dot{G}$incidence', '$\dot{G}$angle', '.', '$< /s >$'

8.2.4 Topic Modelling

Topic Modelling (TM) is a statistical approach for discovering the hidden (latent) topics in a collection of documents. If TM was for instance applied to Jules Verne's "*Twenty Thousand Leagues Under the Sea*" novel, some of the discovered topics could include *Submarine*, *Travel*, or *Ocean*. TM has been previously suggested as a preferred method for building text representations by Al-Salemi et al. in [273] and Yun and Geum in [274]. Sriurai [275] also demonstrated that TM was a more efficient method for building feature representations of texts than the Bag-of-Words (BOWs) approach. The latter method was first mentioned by Harris in [276]. With this approach, a document is mapped to a vector of length n, corresponding to its vocabulary size. Each vector element of index i corresponds to the frequency of a word w in the document.

An early method, Latent Semantic Analysis (LSA), also referred to as Latent Semantic Indexing (LSI), was proposed by Deerwester et al. in [277]. Their approach is based on a large term-by-document matrix, where each line corresponds to a document and each column to a term frequency. They reduce this matrix to a representation in a so-called latent semantic space using a Singular-Value Decomposition (SVD). Their assumptions is that similar documents will have similar representations in the latent semantic space. The authors developed this method for automatic indexing and information retrieval applications. Rather than relying on classic term-matching methods, their method compares the latent semantic structure, or topics, of documents.

LSA was however outperformed by a probabilistic approach proposed by Hofmann in [167]. The Probabilistic Latent Semantic Analysis (pLSA), sometimes called Probabilistic Latent Semantic Indexing (pLSI), learns a probability model over documents and words. Each document is expressed as a probability distribution over topics, and each topic as a probability distribution over words. A simple example is shown in Figure 8.7. According to [166], the model is however prone to over-fitting and is not a well defined generative model for new documents.

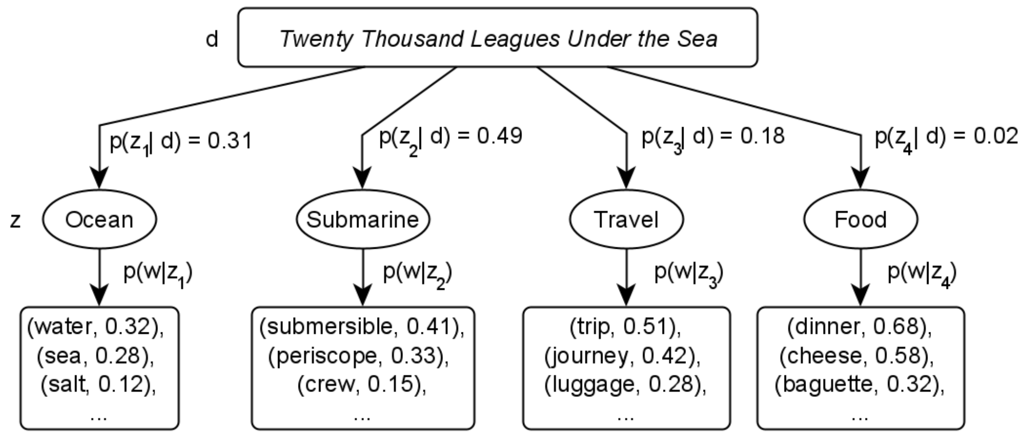
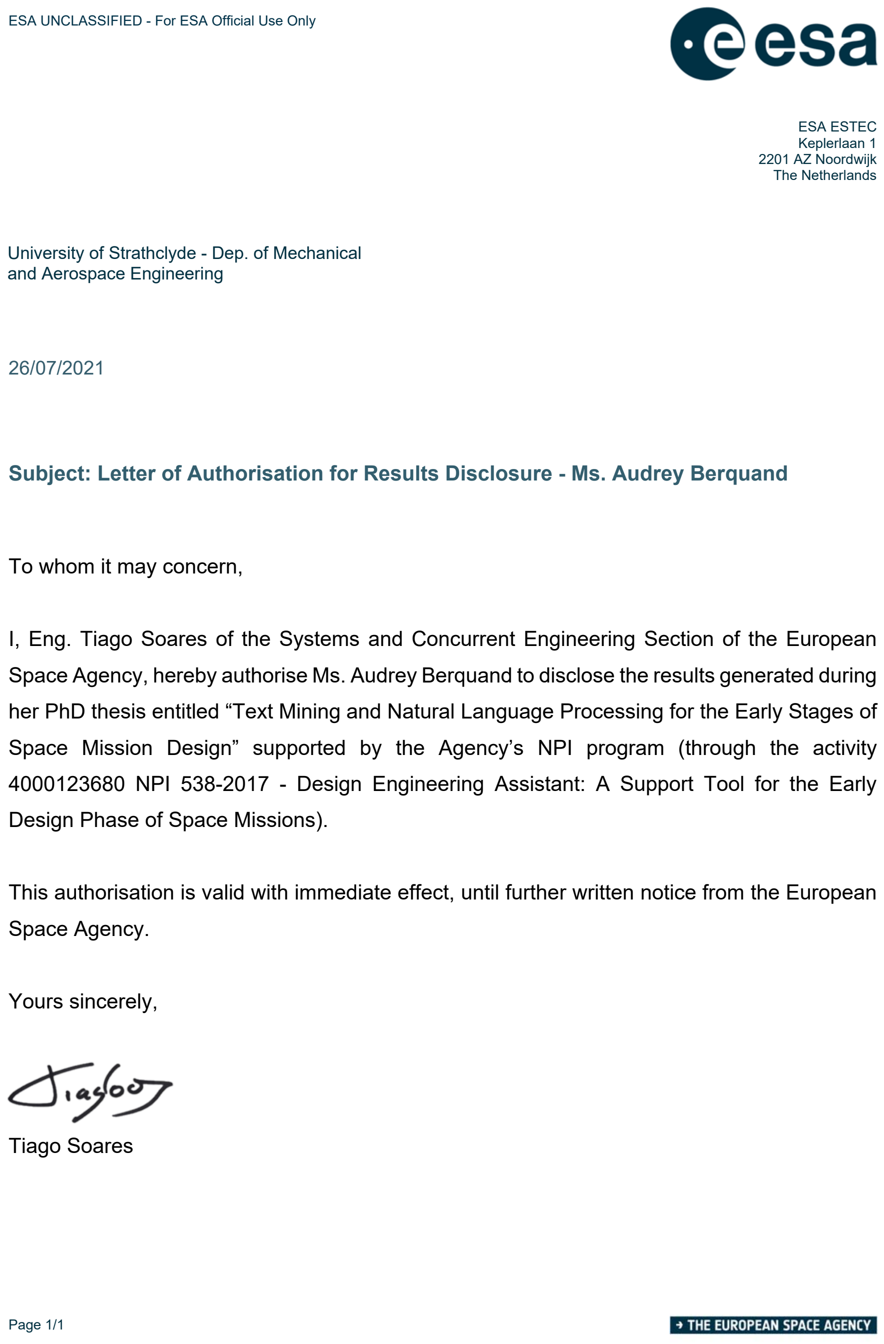

d
Twenty Thousand Leagues Under the Sea
p(z1| d) = 0.31
p(z2| d) = 0.49
p(z3| d) = 0.18
p(z4| d) = 0.02
z
Ocean
Submarine
Travel
Food
p(w|z1)
p(w|z2)
p(w|z3)
p(w|z4)
(water, 0.32), (sea, 0.28), (salt, 0.12), ...
(submersible, 0.41), (periscope, 0.33), (crew, 0.15), ...
(trip, 0.51), (journey, 0.42), (luggage, 0.28), ...
(dinner, 0.68), (cheese, 0.58), (baguette, 0.32), ...

Figure 8.7: Example for the probabilistic Topic Modelling approach, where a document d has follows a topic distribution $p(z_n|d)$, and each topic z has a word distribution $p(w|z)$.

The Latent Dirichlet Allocation (LDA) first introduced by Blei et al. [166] does however generalise well to new (unseen) documents, and was thus selected as baseline in this thesis.

8.3 Appendix 3: ESA Letter of Authorisation for Results Disclosure



ESA UNCLASSIFIED - For ESA Official Use Only

ESA ESTEC
Keplerlaan 1
2201 AZ Noordwijk
The Netherlands

University of Strathclyde - Dep. of Mechanical
and Aerospace Engineering

26/07/2021

Subject: Letter of Authorisation for Results Disclosure - Ms. Audrey Berquand

To whom it may concern,

I, Eng. Tiago Soares of the Systems and Concurrent Engineering Section of the European Space Agency, hereby authorise Ms. Audrey Berquand to disclose the results generated during her PhD thesis entitled "Text Mining and Natural Language Processing for the Early Stages of Space Mission Design" supported by the Agency's NPI program (through the activity 4000123680 NPI 538-2017 - Design Engineering Assistant: A Support Tool for the Early Design Phase of Space Missions).

This authorisation is valid with immediate effect, until further written notice from the European Space Agency.

Yours sincerely,

[illegible]

Tiago Soares

Page 1/1

→ THE EUROPEAN SPACE AGENCY

Bibliography

[1] V. Schroder, R. Regele, J. Sommer, T. Eisenberg, and C. Karrasch, “GNC System Design for the Crew Interactive MObile companiON (CIMON),” in *Proceedings of the 69th International Astronautical Congress (IAC)*, Bremen, Germany, 2018.

[2] P. J. Lucas and L. C. van der Gaag, *Principles of Expert Systems*, 1991. [Online]. Available: https://www.cs.ru.nl/~peterl/proe.pdf

[3] M. Haenlein and A. Kaplan, “A brief history of artificial intelligence: On the past, present, and future of artificial intelligence,” *California Management Review*, vol. 61, no. 4, pp. 5–14, 2019.

[4] A. De Mauro, M. Greco, and M. Grimaldi, “A formal definition of Big Data based on its essential features,” *Library Review*, vol. 65, no. 3, pp. 122–135, 2016.

[5] E. Marakakis, K. Vassilakis, E. Kalivianakis, and S. Micheloyiannis, “Expert System for Epilepsy with Uncertainty 1,” no. December, pp. 19–21, 2005.

[6] A. Arruda, “An Ethical Obligation to Use Artificial Intelligence ? An Examination of the Use of Artificial Intelligence in Law and the Model Rules of Professional Responsibility,” *American Journal of Trial Advocacy*, vol. 40, no. 3, pp. 443–458, 2017.

[7] F. Murdaca, A. Berquand, A. Riccardi, T. Soares, S. Gerené, N. Brauer, and K. Kumar, “Artificial Intelligence for Early Design of Space Missions in Support of Concurrent Engineering Sessions,” *Proceedings of the 8th International Systems & Concurrent Engineering for Space Applications Conference (SECESA 2018)*, pp. 1–8, 2018.

[8] A. Berquand, F. Murdaca, A. Riccardi, T. Soares, S. Gerené, N. Brauer, and K. Kumar, "Towards an Artificial Intelligence based Design Engineering Assistant for the Early Design of Space Missions," *Proceedings of the 69th International Astronautical Congress (IAC 2018)*, 2018.

[9] ——, "Artificial Intelligence for the Early Design Phases of Space Missions," in *2019 IEEE Aerospace Conference*, Big Sky, MT, USA, 2019, pp. 1–20.

[10] A. Berquand, I. McDonald, A. Riccardi, and Y. Moshfeghi, "The automatic categorisation of space mission requirements for the Design Engineering Assistant," in *Proceedings of the 70th International Astronautical Congress (IAC 2019)*, 2019, pp. 1–12.

[11] A. Berquand, Y. Moshfeghi, and A. Riccardi, "Space mission design ontology: extraction of domain-specific entities and concepts similarity analysis," in *AIAA Scitech 2020 Forum*, no. January, Orlando, Florida, USA, 2020.

[12] A. Berquand and A. Riccardi, "From Engineering Models to Knowledge Graph : Delivering New Insights Into Model," in *Proceedings of the 9th International Conference on Systems & Concurrent Engineering for Space Applications (SECESA)*, 2020.

[13] A. Berquand, Y. Moshfeghi, and A. Riccardi, "SpaceLDA: Topic distributions aggregation from a heterogeneous corpus for space systems," *Engineering Applications of Artificial Intelligence*, vol. 102, no. April 2021, p. 104273, 2021. [Online]. Available: https://doi.org/10.1016/j.engappai.2021.104273

[14] A. Berquand, P. Darm, and A. Riccardi, "SpaceTransformers : language modeling for space systems," *IEEE Access*, vol. 9, pp. 133 111–133 122, 2021. [Online]. Available: 10.1109/ACCESS.2021.3115659

[15] M. A. Aguirre, *Introduction to Space Systems: Design and Synthesis.* Springer, 2013.

[16] A. R. L. Tatnall, J. B. Farrow, M. Bandecchi, and C. R. Francis, “Spacecraft System Engineering,” in *Spacecraft Systems Engineering*, 4th ed., P. Fortescue, G. Swinerd, and J. Stark, Eds. John Wiley & Sons, Ltd., 2011, ch. 20, pp. 643–678.

[17] G. Richardson, K. Schmitt, M. Covert, and C. Rogers, “Small Satellite Trends 2009-2013,” in *Proceedings of the AIAA/USU Conference on Small Satellites, Technical Session VII: Opportunities, Trends and Initiatives, SSC15-VII-3*, Utah State University, UT, USA, 2015. [Online]. Available: https://digitalcommons.usu.edu/smallsat/2015/all2015/48/

[18] ECSS, “ECSS Abbreviated Terms,” Tech. Rep., 2017. [Online]. Available: https://ecss.nl/home/ecss-glossary-abbreviations/

[19] W. Kriedte, “ECSS - A Single Set of European Space Standards,” in *Spacecraft Structures, Materials and Mechanical Testing*, W. Burke, Ed., Noordwijk, Netherlands, 1996, pp. 321–327.

[20] M. Blythe, M. Saunders, D. Pye, L. Voss, R. Moreland, K. Symons, and L. Bromley, *NASA Space Flight Program and Project Management Handbook*, 2014. [Online]. Available: https://ntrs.nasa.gov/api/citations/20150000400/downloads/20150000400.pdf

[21] ECSS Secretariat, “ECSS-M-ST-10C Rev. 1 - Space project management - Project planning and implementation,” Tech. Rep. March, 2009. [Online]. Available: https://ecss.nl/standard/ecss-m-st-10c-rev-1-project-planning-and-implementation/

[22] ESA, “European Code of Conduct for Space Debris Mitigation,” Tech. Rep. 1, 2004. [Online]. Available: http://www.unoosa.org/documents/pdf/spacelaw/sd/2004-B5-10.pdf

[23] I. M. L. Ferreira, “Enhancing the conceptual design phase of complex engineering systems with an integrated methodology and support tools,” Ph.D. dissertation, Universidade Técnica de Lisboa Instituto Superior Técnico, 2012.

[24] J. E. Folkestad and R. L. Johnson, "Resolving the conflict between design and manufacturing: Integrated Rapid Prototyping and Rapid Tooling (IRPRT)," *Journal of Industrial Technology*, vol. 17, no. 4, 2001.

[25] R. I. Winner, J. P. Pennell, H. E. Bertrand, and M. M. G. Slusarczuk, "The role of concurrent engineering in weapons system acquisition," Institute for Defense Analyses, Tech. Rep., 1988. [Online]. Available: https://apps.dtic.mil/dtic/tr/fulltext/u2/a203615.pdf

[26] R. P. Smith, "The Historical Roots of Concurrent Engineering Fundamentals," *IEEE Transactions on Engineering Management*, vol. 44, no. 1, pp. 67–78, 1997.

[27] M. Bandecchi, B. Melton, B. Gardini, and F. Ongaro, "The ESA / ESTEC Concurrent Design Facility," in *Proceedings of the 2nd European Systems Engineering Conference (EuSEC)*, Munich, Germany, 2000.

[28] A. Pickering, "ESA Concurrent Design Facility Infopack 2017," 2016. [Online]. Available: https://esamultimedia.esa.int/docs/cdf/CDF_infopack_2017.pdf

[29] P. Wognum, R. Jardim-Goncalves, R. de Graaf, F. Lettice, and R. Roy, "Analysis on 10 years of ISPE/CEconf community," in *Proceedings of the 10th ISPE International Conference on Concurrent Engineering (ISPE CE 2003)*, R. Jardim-Goncalves, J. Cha, and A. Steiger-Garcao, Eds. Madeira Island, Portugal: A. A. Balkema Publishers, 2003, pp. 1–7.

[30] D. Juarez, J. Segui, A. Mengual, and S. Ferrándiz, "Concurrent engineering applied to key industrial sectors," *Annals of The University of Oradea. Fascicle of Management and Technological Engineering*, no. 3, pp. 81–84, 2015. [Online]. Available: http://hdl.handle.net/10251/65946

[31] J. L. Smith, "Concurrent Engineering in the Jet Propulsion Laboratory Project Design Center," *SAE Transactions*, vol. 107, pp. 1106–1118, 1998.

[32] ICE LAB, "Concurrent and collaborative design studio - the facility," p. 3, 2021. [Online]. Available: http://icelab.uk/resources/ccds/

[33] A. Braukhane, "The DLR Concurrent Engineering Facility (CEF)," 2011.

[34] A. R. Wilson and A. Berquand, "Concurrent Engineering and Social Distancing 101 : Lessons Learned During a Global Pandemic," in *Proceedings of the 9th International Systems & Concurrent Engineering for Space Applications Conference (SECESA 2020)*, no. October, Digital Event, 2020.

[35] L. Walker, C. Greco, M. Di Carlo, A. Wilson, L. Ricciardi, A. Berquand, and M. Vasile, "Nanospacecraft Exploration of Asteroids by Collision and flyby Reconnaissance (NEACORE)," in *Proceeding of the 2019 IAA Low-Cost Planetary Missions Conference (LCPM)*, Toulouse, France, 2019.

[36] B. Sherwood and D. McCLeese, "JPL innovation foundry," *Acta Astronautica*, vol. 89, pp. 236–247, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.actaastro.2013.04.020

[37] X. Collaud and D. Evans, "The Devil is in the Details : Lessons Learned From Operations for Phase 0 Studies," in *Proceedings of the 8th International Systems & Concurrent Engineering for Space Applications Conference (SECESA 2018)*, Glasgow, UK, 2018, pp. 1–4.

[38] A. Braukhane, V. Maiwald, A. Martelo, D. Quantius, and O. Romberg, "Be Aware of the Squad: Lessons Learnt from 50 Concurrent Engineering Studies for Space Systems," in *Proceedings of the 66th International Astronautical Congress*, Jerusalem, Israel, 2015, p. 8. [Online]. Available: http://elib.dlr.de/98729/1/BRAUKHANE_Andy_paper_IAC-15_D1.3.5x30104_v1.0.pdf

[39] W. J. Verhagen, J. Stjepandić, and N. Wognum, "Chapter 28 - Challenges of CE," in *Concurrent Engineering in the 21st Century*, Josip Stjepandić, Nel Wognum, and Wim J.C. Verhagen, Eds. Springer, 2015, ch. 28. [Online]. Available: https://link.springer.com/content/pdf/10.1007%2F978-3-319-13776-6.pdf

[40] K. Dalkir, *Knowledge management in theory and practice*, 3rd ed. The MIT Press, 2017.

[41] S. L. Kendal and M. Creen, *An Introduction to Knowledge Engineering.* Springer, London, 2007.

[42] Karl E. Misulis and Mark E. Frisse, "Data, Information, and Knowledge," in *Essentials of Clinical Informatics.* Oxford University Press, 2019.

[43] H. Sack and M. Alam, "Knowledge Graphs Lecture 4-Knowledge Representation with Ontologies 4.1 A Brief History of Ontologies Autumn 2020."

[44] R. L. Ackoff, "From data to wisdom," *Journal of Applied Systems Analysis*, vol. 16, pp. 3–9, 1989.

[45] J. Rowley, "The wisdom hierarchy: Representations of the DIKW hierarchy," *Journal of Information Science*, vol. 33, no. 2, pp. 163–180, 4 2007.

[46] IAF-IPMC, "IAF-IPMC Young Professionals Workshop 2018 Workshop Results Report," Bremen, Germany, Tech. Rep. March, 2019. [Online]. Available: https://www.iafastro.org/assets/files/static/ipmc/report/ipmc-yp-workshop-report-2018-final.pdf

[47] A. T. W. Min, R. Sagarna, A. Gupta, Y. S. Ong, and C. K. Goh, "Knowledge Transfer Through Machine Learning in Aircraft Design," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 48–60, 2017.

[48] G. Baldesi, R. M. Dow, R. Houben, and A. Vena, "From Libraries to ESA knowledge and learning centres : Key Features and Status of Implementation," in *Proceeding of the 69th International Astronautical Congress (IAC)*, Bremen, Germany, 2018.

[49] F. Bolger, "Chapter 16 - The Selection of Experts for (Probabilistic) Expert Knowledge Elicitation," in *Elicitation - The Science and Art of Structuring Judgement*, internatio ed., L. C. Dias, A. Morton, and J. Quigley, Eds. Springer, 2018, ch. 16, pp. 393–443.

[50] N. R. Shadbolt and P. R. Smart, “Knowledge Elicitation : Methods , Tools and Techniques,” in *Evaluation of Human Work*, 4th ed., J. R. Wilson and S. Sharples, Eds. Boca Raton, Florida, USA: CRC Press, 2015, ch. 7.

[51] G. Tokadlı and M. C. Dorneich, “Development of Design Requirements for a Cognitive Assistant in Space Missions Beyond Low Earth Orbit,” *Journal of Cognitive Engineering and Decision Making*, vol. 12, no. 2, pp. 131–152, 2018.

[52] M. Flasiński, “Chapter 16: Application Areas of AI Systems,” in *Introduction to Artificial Intelligence.* Springer International Publishing Switzerland, 2016, ch. 16, pp. 223–234. [Online]. Available: http://link.springer.com/10.1007/978-3-319-40022-8

[53] S. Bird, E. Klein, and E. Loper, “Categorizing and Tagging Words,” in *Natural Language Processing with Python.* O’Reilly Media, 2009, ch. 5, pp. 179–220.

[54] J. Krishnan, P. Coronado, and T. Reed, “Seva: A systems engineer’s virtual assistant,” in *Proceedings of the AAAI 2019 Spring Symposium on Combining Machine Learning with Knowledge Engineering (AAAI-MAKE 2019)*, Stanford University, Palo Alto, California, USA, 2019.

[55] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Pearson Education, Inc., 2010, vol. 4.

[56] A. Ashwitha, G. Shruthi, H. R. Shruthi, M. Upadhyaya, A. P. Ray, and T. C. Manjunath, “Sarcasm detection in natural language processing,” *Materials Today: Proceedings*, vol. 37, pp. 3324–3331, 2021. [Online]. Available: https://doi.org/10.1016/j.matpr.2020.09.124

[57] R. A. Potamias, G. Siolas, and A. G. Stafylopatis, “A transformer-based approach to irony and sarcasm detection,” *Neural Computing and Applications*, vol. 32, pp. 17 309–17 320, 2020. [Online]. Available: https://doi.org/10.1007/s00521-020-05102-3

[58] N. Chomsky, *Syntactic structure.* Mouton & Co., 1957.

[59] ——, *Aspects of the Theory of Syntax.* MIT Press, 1965.

[60] C. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing.* Cambridge, MA: MIT Press, 1999. [Online]. Available: https://nlp.stanford.edu/fsnlp/

[61] T. Jo, *Text Mining: Concepts, Implementation, and Big Data Challenge.* Springer International Publishing AG, 2019, vol. 36, no. 2.

[62] S. VijayGaikwad, A. Chaugule, and P. Patil, "Text Mining Methods and Techniques," *International Journal of Computer Applications*, vol. 85, no. 17, pp. 42–45, 2014.

[63] N. Zanini and V. Dhawan, "Text Mining: An introduction to theory and some applications," *Research Matters: A Cambridge Assessment publication*, vol. 19, no. Winter 2015, pp. 38–44, 2015.

[64] S. Fratini, "Artificial Intelligence in ESA," Tech. Rep. 1.1, 2019.

[65] L. Simonini, ""The Manager" User's guide," European Space Agency, Tech. Rep. 1, 2005.

[66] M. Brunelli, "Introduction and Fundamentals," in *Introduction to the Analytic Hierarchy Process.* SpringerBriefs in Operations Research, 2015, ch. 1, pp. 1–15.

[67] A. Viros and D. Selva, "Daphne: A Virtual Assistant for Designing Earth Observation Distributed Spacecraft Missions," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 30–48, 2020.

[68] D. Selva, B. G. Cameron, and E. F. Crawley, "Rule-based system architecting of Earth observing systems: Earth Science Decadal Survey," *Journal of Spacecraft and Rockets*, vol. 51, no. 5, pp. 1505–1521, 2014. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/1.A32656

[69] D. Kaslow, B. Ayres, P. T. Cahill, L. Hart, and R. Yntema, "A model-based systems engineering (MBSE) approach for defining the behaviors of CubeSats,"

in *Proceedings of the 2017 IEEE Aerospace Conference.* Big Sky, MT, USA: IEEE, 2017, pp. 1–14.

[70] S. C. Spangelo, D. Kaslow, C. Delp, B. Cole, L. Anderson, E. Fosse, B. S. Gilbert, L. Hartman, T. Kahn, and J. Cutler, "Applying model based systems engineering (MBSE) to a standard CubeSat," in *Proceedings of the 2012 IEEE Aerospace Conference.* Big Sky, MT, USA: IEEE, 2012, pp. 1–20.

[71] L. Wang, M. Izygon, S. Okon, L. Garner, and H. Wagner, "Effort to Accelerate MBSE Adoption and Usage at JSC," in *Proceedings of the AIAA Space 2016*, Long Beach, CA, USA, 2016, pp. 1–10.

[72] ECSS Secretariat, "ECSS-E-TM-10-25A Engineering design model data exchange CDF," Tech. Rep. October, 2010.

[73] W. J. Larson and J. R. Wertz, *Space Mission Analysis and Design*, 3rd ed., S. T. Library, Ed., 2005.

[74] S. J. Kapurch, "NASA Systems Engineering Handbook," *NASA Special Publication*, p. 360, 2007. [Online]. Available: https://www.nasa.gov/connect/ebooks/nasa-systems-engineering-handbook

[75] G. C. Birur, G. Siebes, and T. D. Swanson, "Spacecraft Thermal Control," in *Encyclopedia of Physical Science and Technology.* Elsevier, 2003, pp. 485–505. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/B0122274105009005

[76] F. Liu, S. Lu, and Y. Sun, *Guidance and Control Technology of Spacecraft on Elliptical Orbit*, Springer, Ed., 2019.

[77] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of NAACL-HLT 2019.* Minneapolis, Minnesota: Association for Computational Linguistics, 2019, pp. 4171–4186. [Online]. Available: http://arxiv.org/abs/1810.04805

[78] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., no. 33. Vancouver, Canada: Curran Associates, Inc., 2020, pp. 1877–1901.

[79] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "GLUE: a Multi-task Benchmark and Analysis Platform for Natural Language Understanding," in *Proceedings of the 7th International Conference on Learning Representation (ICLR 2019).* New Orleans, LA, US: OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=rJ4km2R5t7

[80] ESA, "SMOS Systems Requirements Document," Tech. Rep., 2005. [Online]. Available: https://earth.esa.int/documents/10174/1854456/SMOS-Mission-Requirements

[81] ——, "MarcoPolo-R Mission Requirements Document," Tech. Rep., 2012. [Online]. Available: https://sci.esa.int/web/marcopolo-r/-/51297-marcopolo-r-mission-requirements-document

[82] ECSS, "ECSS-P-00C standardization objectives, policies and organization," Tech. Rep. 2nd issue, 2013.

[83] D. Maynard, K. Bontcheva, and I. Augenstein, "Linguistic Processing," in *Natural Language Processing for the Semantic Web*, Y. Ding and P. Groth, Eds. Morgan & Claypool, 2017, ch. 2, pp. 9–24.

[84] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit," in *Proceedings of*

the 52nd Annual Meeting ofthe Association for Computational Linguistics: System Demonstrations. Baltimore, Maryland USA: Association for Computational Linguistics, 2014, pp. 55–60.

[85] Apache OpenNLP Development Community, "Apache OpenNLP Developer Documentation," 2021. [Online]. Available: https://opennlp.apache.org/docs/

[86] S. Bird, E. Klein, and E. Loper, *Natural Language Processin with Python.* O'Reilly Media Inc., 2009.

[87] I. Honnibal, Matthew Montani and A. Van Landeghem, Sofie Boyd, "spaCy: Industrial-strength Natural Language Processing in Python," 2020. [Online]. Available: https://doi.org/10.5281/zenodo.1212303

[88] F. N. A. Al Omran and C. Treude, "Choosing an NLP Library for Analyzing Software Documentation: A Systematic Literature Review and a Series of Experiments," in *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*, 2017, pp. 187–197.

[89] X. Schmitt, S. Kubler, J. Robert, M. Papadakis, and Y. LeTraon, "A Replicable Comparison Study of NER Software: StanfordNLP, NLTK, OpenNLP, SpaCy, Gate," in *2019 6th International Conference on Social Networks Analysis, Management and Security, (SNAMS)*, 2019, pp. 338–343.

[90] T. Kiss and J. Strunk, "Unsupervised multilingual sentence boundary detection," *Computational Linguistics*, vol. 32, no. 4, pp. 485–525, 2006.

[91] A. Taylor, M. Marcus, and B. Santorini, "The Penn Treebank: An Overview," in *Treebanks: Building and Using Parsed Corpora - Text, Speech and Language Technology*, A. Abeille, Ed. Dordrecht: Springer, 2003, no. 20, ch. 1, pp. 5–22. [Online]. Available: https://doi.org/10.1007/978-94-010-0201-1_1

[92] C. Fellbaum, "WordNet," in *Theory and Applications ofOntology: Computer Applications*, R. Poli, M. Healy, and A. Kameas, Eds. Springer Dordrecht, 2010, ch. 10, pp. 231–243.

[93] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval." *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[94] C. D. Manning, P. Raghavan, and S. Hinrich, *An Introduction to Information Retrieval.* Cambridge University Press, 2009.

[95] INCOSE, "SYSTEMS ENGINEERING VISION 2020 (INCOSE-TP-2004-004-02)," Tech. Rep., 2007.

[96] RHEA Group, "Digital Engineering Hub Pathfinder & New Tools," Tech. Rep. March, 2021.

[97] ESA, "Open Concurrent Design Tool Community Portal." [Online]. Available: https://ocdt.esa.int/

[98] RHEA Group, "CDP-4 User Manual." [Online]. Available: http://cdp4docs.rheagroup.com/

[99] Satsearch, "Streamlining space engineering design with the new satsearch-CDP4 integration tool," Delft.

[100] ESA Education Office, "Fly Your Satellite! Design Specifications," Reference number ESA-DG-SET-2019-1636, Tech. Rep., 2019.

[101] C. Jenkins, R. Hope, A. R. Wilson, and A. Berquand, "STRATHcube : The design of a student CubeSat using concurrent engineering methods," in *Proceedings of the 9th International Conference on Systems & Concurrent Engineering for Space Applications (SECESA)*, Digital Event, 2020.

[102] J.-l. Terraillon, H.-p. de Koning, and S. Valera, "Overall System Modelling for System Engineering (OSMoSE) - Space System Ontology 1st Brainstorming Workshop Report," Tech. Rep. 0.2, 2019.

[103] P. Cimiano, "Ontology Learning from Text," in *Ontology Learning and Population from Text.* Springer US, 2006, pp. 19–34. [Online]. Available: https://doi.org/10.1007/978-0-387-39252-3_3

[104] P. Buitelaar, P. Cimiano, and B. Magnini, "Ontology Learning from Text : An Overview," *Learning*, pp. 1–10, 2004.

[105] G. Lakemeyer and B. Nebel, *Foundations of Knowledge Representation and Reasoning*, ser. Lecture Notes in Computer Science, G. Lakemeyer and B. Nebel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994. [Online]. Available: http://link.springer.com/10.1007/3-540-58107-3

[106] G. Jakus, V. Milutinovic, S. Omerovic, and S. Tomazic, "Knowledge representation," in *Concepts, Ontologies, and Knowledge Representation.* SpringerBriefs in Computer Science, 2013, pp. 47–62.

[107] B. Nebel, "Logics for Knowledge Representation," *International Encyclopedia of the Social & Behavioral Sciences: Second Edition*, no. June, pp. 319–321, 2015.

[108] C. Roussey, F. Pinet, M. A. Kang, and O. Corcho, "An Introduction to Ontologies and Ontology Engineering," in *Ontologies in Urban Development Projects.* Springer, London, 2011, vol. 1, pp. 9–38. [Online]. Available: https://link.springer.com/chapter/10.1007/978-0-85729-724-2_2

[109] H. Sack, "Semantic Web Technologies Lecture 4-Knowledge Representation 3. How to define a formal model of an ontology? Autumn 2020," Tech. Rep.

[110] A. Fokoue, A. Kershenbaum, L. Ma, E. Schonberg, and K. Srinivas, "The Summary Abox: Cutting Ontologies Down to Size," in *The Semantic Web - ISWC 2006.* Springer Berlin Heidelberg, 2006, pp. 343–356.

[111] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, no. 2, pp. 199–220, 6 1993.

[112] T. Gruber, "Ontology," in *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu, Eds. Springer, Boston, MA., 2008, pp. 1963–1965.

[113] M. Sabou, "An Introduction to Semantic Web Technologies," S. Biffl and M. Sabou, Eds., 2016, p. 413. [Online]. Available: http://link.springer.com/10.1007/978-3-319-41490-4

[114] A. Maedche and S. Staab, “Ontology Learning for the Semantic Web,” *IEEE Intelligent Systems*, vol. 16, no. 2, pp. 72–79, 2001.

[115] R. Lourdusamy and S. Abraham, “A Survey on Methods of Ontology Learning from Text,” in *Intelligent Computing Paradigm and Cutting-edge Technologies*, L. C. Jain, S.-L. Peng, B. Alhadidi, and S. Pal, Eds. Springer International Publishing, 2020, pp. 113–123.

[116] S. Staab and R. Studer, *Handbook on ontologies*, 2nd ed. Springer, 2009.

[117] W. Wong, W. Liu, and M. Bennamoun, “Ontology learning from text: A look back and into the future,” *ACM Computing Surveys*, vol. 44, no. 4, 2012.

[118] J. Lehmann and J. Volker, “An Introduction to Ontology Learning,” *Perspectives on Ontology Learning*, pp. ix–xvi, 2014. [Online]. Available: http://jens-lehmann.org/files/2014/pol_introduction.pdf

[119] M. Marciniak and A. Mykowiecka, “Terminology extraction from medical texts in Polish,” *Journal of Biomedical Semantics*, vol. 5, no. 1, pp. 1–14, 2014. [Online]. Available: https://doi.org/10.1186/2041-1480-5-24

[120] P. Martin-Chozas and P. Calleja, “Challenges of Terminology Extraction from Legal Spanish Corpora,” in *Proceedings of the 2nd Workshop on Technologies for Regulatory Compliance*, 2018, pp. 73–83. [Online]. Available: http://oa.upm.es/67249/

[121] K. Ahmad and L. Gillam, “Automatic ontology extraction from unstructured texts,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3761 LNCS, no. i, pp. 1330–1346, 2005.

[122] ECSS, “ECSS Terms and Definitions,” Tech. Rep., 2007. [Online]. Available: https://ecss.nl/home/ecss-glossary-terms/

[123] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” in *Proceedings of the International*

Conference on Learning Representations (ICLR 2013), 2013. [Online]. Available: http://arxiv.org/abs/1301.3781

[124] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A Neural Probabilistic Language Model," *Journal ofMachine Learning Research*, vol. 3, p. 1137–1155, 2003. [Online]. Available: https://dl.acm.org/doi/10.5555/944919.944966

[125] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *Advances in neural information processing systems*, vol. 1, pp. 3111–3119, 2013.

[126] X. Rong, "word2vec Parameter Learning Explained," 2014. [Online]. Available: http://arxiv.org/abs/1411.2738

[127] A. Smywiński-Pohl, K. Wróbel, K. Lasocki, and M. Strzała, "Automatic Construction of a Polish Legal Dictionary with Mappings to Extra-Legal Terms Established Via Word Embeddings," in *17th International Conference on Artificial Intelligence and Law*, no. 1, Montreal, Canada, 2019, p. 7.

[128] M. Gao, F. Chen, and R. Wang, "Improving medical ontology based on word embedding," in *Proceedings of the 6th International Conference on Bioinformatics and Computational Biology*, 2018, pp. 121–127.

[129] G. Li, "Improving Biomedical Ontology Matching Using Domain-specific Word Embeddings," in *Proceedings of the 4th International Conference on Computer Science and Application Engineering*, 2020, pp. 1–5. [Online]. Available: https://doi.org/10.1145/3424978.3425102

[130] J. Youn, T. Naravane, and I. Tagkopoulos, "Using Word Embeddings to Learn a Better Food Ontology," *Frontiers in Artificial Intelligence*, vol. 3, no. November, pp. 1–8, 2020.

[131] L. Zhang, J. Li, and C. Wang, "Automatic synonym extraction using Word2Vec and spectral clustering," *Chinese Control Conference, CCC*, pp. 5629–5632, 2017.

[132] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks.* ELRA, 2010, pp. 45–50. [Online]. Available: http://is.muni.cz/publication/884893/en

[133] I. T. Jolliffe, *Principal Component Analysis*, ser. Springer Series in Statistics. New York: Springer-Verlag, 2002. [Online]. Available: http://link.springer.com/10.1007/b98835

[134] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, USA, 2017.

[135] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," no. 1, 2019. [Online]. Available: http://arxiv.org/abs/1907.11692

[136] I. Beltagy, K. Lo, and A. Cohan, "Scibert: A pretrained language model for scientific text," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).* Hong Kong, China: Association for Computational Linguistics, 2019, pp. 3615–3620. [Online]. Available: https://www.aclweb.org/anthology/D19-1371

[137] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine

Translation," *CoRR*, vol. abs/1609.0, pp. 1–23, 2016. [Online]. Available: http://arxiv.org/abs/1609.08144

[138] J. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers).* Melbourne, Australia: Association for Computational Linguistics, 2018, pp. 328–339.

[139] J. S. Lee and J. Hsiang, "Patent classification by fine-tuning BERT language model," *World Patent Information*, vol. 61, no. June 2020, p. Article 101965, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0172219019300742

[140] J. Krishnan, P. Coronado, H. Purohit, and H. Rangwala, "Common-Knowledge Concept Recognition for SEVA," *AAAI 2020 Spring Symposium on Com- bining Machine Learning and Knowledge Engineering in Practice (AAAI-MAKE 2020)*, 2020.

[141] S. R. Hirshorn, *NASA System Engineering Handbook*, 2nd ed., 2016. [Online]. Available: https://www.nasa.gov/sites/default/files/atoms/files/nasa_systems_engineering_handbook_0.pdf

[142] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "BioBERT: A pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020.

[143] C. N. Chau, T. S. Nguyen, and L. M. Nguyen, "VNLawBERT: A Vietnamese Legal Answer Selection Approach Using BERT Language Model," in *Proceedings of the 7th NAFOSTED Conference on Information and Computer Science, NICS 2020.* IEEE, 2020, pp. 298–301.

[144] E. Alsentzer, J. R. Murphy, W. Boag, W.-H. Weng, D. Jin, T. Naumann, and M. B. A. McDermott, "Publicly Available Clinical BERT Embeddings," 2019. [Online]. Available: http://arxiv.org/abs/1904.03323

[145] K. Huang, J. Altosaar, and R. Ranganath, "ClinicalBert: Modeling clinical notes and predicting hospital readmission," in *CHIL '20: ACM Conference on Health, Inference, and Learning; Workshop Track.* Toronto, Ontario, Canada: Association for Computing Machinery, 2020, p. 9 pages.

[146] Z. Liu, D. Huang, K. Huang, Z. Li, and J. Zhao, "FinBERT: A pre-trained financial language representation model for financial text mining," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20) Special Track on AI in FinTech*, Online, 2020, pp. 4513–4519.

[147] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "LEGAL-BERT: The Muppets straight out of Law School," in *Findings of ACL: Empirical Methods in Natural Language Processing (EMNLP 2020) (Short Papers).* Online: Association for Computational Linguistics, 2020, p. 2898–2904. [Online]. Available: https://www.aclweb.org/anthology/2020.findings-emnlp.261

[148] E. Tseytlin, K. Mitchell, E. Legowski, J. Corrigan, G. Chavan, and R. S. Jacobson, "NOBLE - Flexible concept recognition for large-scale biomedical natural language processing," *BMC Bioinformatics*, vol. 17, no. 1, pp. 1–15, 2016. [Online]. Available: http://dx.doi.org/10.1186/s12859-015-0871-y

[149] A. Grivas, B. Alex, C. Grover, R. Tobin, and W. Whiteley, "Not a cute stroke: Analysis of Rule- and Neural Network-based Information Extraction Systems for Brain Radiology Reports," in *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis.* Stroudsburg, PA, USA: Association for Computational Linguistics, 2020, pp. 24–37. [Online]. Available: https://www.aclweb.org/anthology/2020.louhi-1.4

[150] O. Uzuner, B. R. South, S. Shen, and S. L. DuVall, "2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text," *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 552–556, 2011.

[151] Y. Si, J. Wang, H. Xu, and K. Roberts, "Enhancing clinical concept extraction with contextual embeddings," *Journal of the American Medical Informatics Association*, vol. 26, no. 11, pp. 1297–1304, 2019.

[152] A. Brack, J. D'Souza, A. Hoppe, S. Auer, and R. Ewerth, "Domain-Independent Extraction of Scientific Concepts from Research Articles," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12035 LNCS, pp. 251–266, 2020.

[153] M. Hofer, A. Kormilitzin, P. Goldberg, and A. Nevado-Holgado, "Few-shot learning for named entity recognition in medical text," *arXiv*, pp. 1–10, 2018.

[154] M. Al-Smadi, S. Al-Zboon, Y. Jararweh, and P. Juola, "Transfer Learning for Arabic Named Entity Recognition With Deep Neural Networks," *IEEE Access*, vol. 8, pp. 37 736–37 745, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8993806/

[155] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. V. Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers : State-of-the-Art Natural Language Processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations.* Online: Association for Computational Linguistics, 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6

[156] ESA Technology coordination & planning office, "ESA Technology Tree version 4.0 (ESA STM-277)," Tech. Rep., 2020. [Online]. Available: https://esamultimedia.esa.int/multimedia/publications/STM-277/STM-277.pdf

[157] I. Alonso Gómez, "ESA Generic Product Tree (TEC-TP/0045)," Tech. Rep., 2011. [Online]. Available: http://emits.sso.esa.int/emits-doc/e_support/ESA_Generic_Product_Tree_June_2011.pdf

[158] G. Vrbančič and V. Podgorelec, “Transfer learning with adaptive fine-tuning,” *IEEE Access*, vol. 8, pp. 196 197–196 211, 2020.

[159] J. Dodge, G. Ilharco, R. Schwartz, A. Farhadi, H. Hajishirzi, and N. Smith, “Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping,” *arXiv*, 2020.

[160] M. Mosbach, M. Andriushchenko, and D. Klakow, “On the Stability of Fine-tuning BERT: Misconceptions, Explanations, and Strong Baselines,” 6 2020. [Online]. Available: http://arxiv.org/abs/2006.04884

[161] W. A. D. Santos, J. R. Bezerra, L. F. Wanderley Goes, and F. M. F. Ferreira, “Creative Culinary Recipe Generation Based on Statistical Language Models,” *IEEE Access*, vol. 8, pp. 146 263–146 283, 2020.

[162] Z. Liu, D. Huang, and K. Huang, “Pretraining Financial Text Encoder Enhanced by Lifelong Learning,” *IEEE Access*, vol. 8, pp. 184 036–184 044, 2020.

[163] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter,” in *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing*, Vancouver, BC, Canada, 2019. [Online]. Available: http://arxiv.org/abs/1910.01108

[164] E. Strubell and A. Mccallum, “Energy and Policy Considerations for Deep Learning in NLP,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.* Florence, Italy: Association for Computational Linguistics, 2019, p. 3645–3650. [Online]. Available: https://aclanthology.org/P19-1355

[165] P. Darm, “Automatic concept recognition in unstructured data for the design of a space system knowledge graph,” University of Strathclyde, Tech. Rep., 2021.

[166] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent Dirichlet Allocation,” *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003. [Online]. Available: http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

[167] T. Hofmann, "Probabilistic latent semantic indexing," *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1999*, pp. 50–57, 1999. [Online]. Available: https://doi.org/10.1145/312624.312649

[168] Y. Moshfeghi, B. Piwowarski, and J. M. Jose, "Handling Data Sparsity in Collaborative Filtering using Emotion and Semantic Based Features," in *34th International ACM SIGIR Conference on Research and Development in Information*, Beijing, China, 2011, pp. 625–634.

[169] J. Jagarlamudi, H. Daumé III, and R. Udupa, "Incorporating Lexical Priors into Topic Models," in *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics.* Avignon, France: Association for Computational Linguistics, 2012, pp. 204–213.

[170] A. P. Shiryaev, A. V. Dorofeev, A. R. Fedorov, L. G. Gagarina, and V. V. Zaycev, "LDA Models for Finding Trends in Technical Knowledge Domain," in *2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus).* IEEE, 2017, pp. 551–554.

[171] J. S. Park, N. R. Kim, H. R. Choi, and E. Han, "A new forecasting system using the latent dirichlet allocation (LDA) topic modeling technique," *WSEAS Transactions on Environment and Development*, vol. 14, pp. 363–373, 2018.

[172] L. Layman, A. P. Nikora, J. Meek, and T. Menzies, "Topic modeling of NASA space system problem reports research in practice," *Proceedings of the 13th Working Conference on Mining Software Repositories, MSR 2016*, pp. 303–314, 2016.

[173] T. Iqbal, P. Elahidoost, and L. Lúcio, "A Bird's Eye View on Requirements Engineering and Machine Learning," in *2018 25th Asia-Pacific Software Engineering Conference (APSEC)*, Nara, Japan, 2018, pp. 11–20.

[174] L. Chen, H. Zhang, J. M. Jose, H. Yu, Y. Moshfeghi, and P. Triantafillou, "Topic detection and tracking on heterogeneous information," *J Intell Inf Syst*, vol. 51, pp. 115–137, 2018.

[175] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth, "The Author-Topic Model for Authors and Documents," in *UAI '04: Proceedings of the 20th conference on Uncertainty in artificial intelligence*, 2004, pp. 487–494.

[176] D. M. Blei and J. D. Lafferty, "A Correlated Topic Model of Science," *The Annals of Applied Statistics*, vol. 1, no. 1, pp. 17–35, 2007.

[177] S. J. Blair, Y. Bi, and M. D. Mulvenna, "Aggregated topic models for increasing social media topic coherence," *Journal of Applied Intelligence*, vol. 50, pp. 138–156, 2020.

[178] C. Schnober and I. Gurevych, "Combining Topic Models for Corpus Exploration Applying LDA for Complex Corpus Research Tasks in a Digital Humanities Project," in *TM '15: Proceedings of the 2015 Workshop on Topic Models: Post-Processing and Applications*, Melbourne, Australia, 2015.

[179] L. Alsumait, D. Barbara, and C. Domeniconi, "On-Line LDA : Adaptive Topic Models for Mining Text Streams with Applications to Topic Detection and Tracking," in *8th IEEE International Conference on Data Mining.* IEEE, 2008.

[180] R. Wilcox, "Chapter 6 - Some Multivariate Methods," in *Introduction to Robust Estimation and Hypothesis Testing*, 2013, no. 3rd, pp. 215–289.

[181] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for Hyper-Parameter Optimization," in *NIPS'11: Proceedings of the 24th International Conference on Neural Information Processing Systems*, 2011.

[182] J. Bergstra, D. Yamins, and D. D. Cox, "Making a Science of Model Search : Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures," in *30th International Conference on Machine Learning*, Atlanta, Georgia, USA, 2013.

[183] N. Craswell, "Mean Reciprocal Rank," in *LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems.* Springer, Boston, MA, 2009.

[184] Q. V. Le and T. Mikolov, “Distributed Representations of Sentences and Documents,” vol. 32, 2014. [Online]. Available: http://arxiv.org/abs/1405.4053

[185] T. Renter, A. Borisov, and M. De Rijke, “Siamese CBOW: Optimizing word embeddings for sentence representations,” in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Berlin, Germany: Association for Computational Linguistics, 2016, pp. 941–951. [Online]. Available: https://www.aclweb.org/anthology/P16-1089

[186] M. Pagliardini, P. Gupta, and M. Jaggi, “Unsupervised learning of sentence embeddings using compositional n-gram features,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, vol. 1. New Orleans, Louisiana: Association for Computational Linguistics, 2018, p. 528–540. [Online]. Available: https://www.aclweb.org/anthology/N18-1049

[187] R. Kiros, Y. Zhu, R. Salakhutdinov, R. S. Zemel, A. Torralba, R. Urtasun, and S. Fidler, “Skip-Thought Vectors,” in *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS’15).* Montreal, Canada: MIT Press, 2015, p. 3294–3302. [Online]. Available: https://dl.acm.org/doi/10.5555/2969442.2969607

[188] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence embeddings using siamese BERT-networks,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).* Hong Kong: Association for Computational Linguistics, 2019, pp. 3982–3992. [Online]. Available: https://www.aclweb.org/anthology/D19-1410

[189] J. H. Lau and T. Baldwin, “An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation,” in *Proceedings ofthe 1st Workshop on Representation Learning for NLP.* Berlin, Germany:

Association for Computational Linguistics, 2016, pp. 78–86. [Online]. Available: https://www.aclweb.org/anthology/W16-1609

[190] L. T. B. Ranera, G. A. Solano, and N. Oco, "Retrieval of Semantically Similar Philippine Supreme Court Case Decisions using Doc2Vec," in *International Symposium on Multimedia and Communication Technology, ISMAC 2019*, 2019, pp. 1–6.

[191] H. Zhang and L. Zhou, "Similarity Judgment of Civil Aviation Regulations Based on Doc2Vec Deep Learning Algorithm," in *Proceedings of the 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, CISP-BMEI 2019*, 2019, pp. 1–8.

[192] S. Feng, "The proximity of ideas: An analysis of patent text using machine learning," *PLoS ONE*, vol. 15, no. 7 July, pp. 1–19, 2020. [Online]. Available: http://dx.doi.org/10.1371/journal.pone.0234880

[193] O. Mendsaikhan, H. Hasegawa, Y. Yamaguchi, and H. Shimada, "Identification of cybersecurity specific content using the Doc2Vec language model," in *Proceedings of the IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 1. IEEE, 2019, pp. 396–401.

[194] S. Akef, M. H. Bokaei, and H. Sameti, "Training Doc2Vec on a Corpus of Persian Poems to Answer Thematic Similarity Multiple-Choice Questions," in *Proceedings of the 10th International Symposium on Telecommunications: Smart Communications for a Better Life (IST)*, 2020, pp. 146–149.

[195] CDF ESA, "CDF Study Report - ATHENA: Assessment of an X-Ray Telescope for the ESA Cosmic Vision Program," Tech. Rep., 2014. [Online]. Available: http://sci.esa.int/cosmic-vision/54013-athena-the-advanced-telescope-for-high-energy-astrophysics/

[196] N. Rando, A. Lyngvi, P. Gondoin, D. Lumb, M. Bavdaz, P. Verhoeve, D. de Wilde, A. Parmar, and A. Peacock, "ESA Study of Xeus, a Potential

Follow-on to XMM-NEWTON," in *Proceedings of the International Conference on Space Optics—ICSO 2006*, Noordwijk, Netherlands, 2006, p. 6.

[197] ESA, "IXO: Revealing the physics of the hot Universe," Tech. Rep., 2011. [Online]. Available: https://sci.esa.int/web/ixo/-/48362-ixo-assessment-study-report-yellow-book

[198] ——, "Euclid: Mapping the geometry of the dark Universe," Tech. Rep., 2011. [Online]. Available: https://sci.esa.int/web/euclid/-/48983-euclid-definition-study-report-esa-sre-2011-12

[199] ——, "PLATO - Revealing habitable worlds around solar-like stars," Tech. Rep., 2017. [Online]. Available: https://sci.esa.int/web/plato/-/59252-plato-definition-study-report-red-book

[200] ——, "SMILE Solar wind Magnetosphere Ionosphere Link Explorer," Tech. Rep., 2018. [Online]. Available: https://sci.esa.int/documents/35028/36141/1567260374869-SMILE_RedBook_ESA_SCI_2018_1.pdf

[201] ——, "XIPE: X-Ray Imaging Polarimetry Explorer," Tech. Rep., 2017. [Online]. Available: https://sci.esa.int/web/cosmic-vision/-/59113-xipe-assessment-study-report-yellow-book

[202] ——, "ARIEL: Atmospheric Remote-sensing Infrared Exoplanet Large-survey," Tech. Rep., 2020. [Online]. Available: https://sci.esa.int/web/ariel/-/ariel-definition-study-report-red-book

[203] ——, "M5 SPICA - CDF Study Final Presentation," p. 191, 2018. [Online]. Available: https://sci.esa.int/web/future-missions-department/-/61029-spica-phase-0-cdf-study-internal-final-presentation

[204] ——, "Phobos Sample Return Phobos Moon of Mars Sample Return Mission," Tech. Rep., 2014. [Online]. Available: https://sci.esa.int/web/future-missions-department/-/55323-cdf-study-report-phobos-sample-return

[205] F. Topputo, Y. Wang, C. Giordano, V. Franzese, H. Goldberg, F. Perez-Lissi, and R. Walker, “Envelop of reachable asteroids by M-ARGO CubeSat,” *Advances in Space Research*, vol. 67, no. 2021, pp. 4193–4221, 2021. [Online]. Available: https://doi.org/10.1016/j.asr.2021.02.031

[206] ESA, “EJSM-Laplace: Exploring the emergence of habitable worlds around gas giants,” Tech. Rep., 2011. [Online]. Available: https://sci.esa.int/web/ejsm-laplace/-/48360-ejsm-laplace-assessment-study-report-yellow-book

[207] ——, “LISA: Unveiling a Hidden Universe,” Tech. Rep., 2011. [Online]. Available: https://sci.esa.int/web/lisa/-/48364-lisa-assessment-study-report-yellow-book

[208] ——, “CHEOPS: CHaracterising ExOPlanet Satellite,” Tech. Rep., 2013. [Online]. Available: https://sci.esa.int/documents/34375/36249/1567259940843-CHEOPS_EST_SCI_RP_001_RedBook_i1.0.pdf

[209] ——, “THOR Exploring Plasma Energization in Space Turbulence,” Tech. Rep., 2017. [Online]. Available: https://sci.esa.int/documents/34375/36249/1567260313300-ESA_SCI-2017-3_THOR.pdf

[210] ——, “LOFT: The large observatory for X-ray timing,” Tech. Rep., 2013. [Online]. Available: https://sci.esa.int/web/loft/-/53447-loft-yellow-book

[211] S. Valera, “Semantic modelling and Semantic Interoperability Information Modelling Glossary (Draft),” Tech. Rep., 2018.

[212] M. Seidl, M. Scholz, C. Huemer, and G. Kappel, *UML @ Classroom: An Introduction to Object-Oriented Modeling.* Springer International Publishing Switzerland, 2015.

[213] M. M. Mkhinini, O. Labbani-Narsis, and C. Nicolle, “Combining UML and ontology: An exploratory survey,” *Computer Science Review*, vol. 35, p. 100223, 2020. [Online]. Available: https://doi.org/10.1016/j.cosrev.2019.100223

[214] A. G. Romero, K. Schneider, and M. G. V. Ferreira, “Semantics in space systems architectures,” *Innovations in Systems and Software Engineering*, vol. 12, no. 1, pp. 27–40, 2016.

[215] E. F. Codd, “A Relational Model of Data for Large Shared Data Banks,” *Communications of the ACM*, vol. 13, no. 6, pp. 377–387, 1970.

[216] Y. Li and S. Manoharan, “A performance comparison of SQL and NoSQL databases,” in *IEEE Pacific RIM Conference on Communications, Computers, and Signal Processing (PACRIM).* IEEE, 2013, pp. 15–19.

[217] P. P. S. Chen, “The Entity-Relationship Model—toward a Unified View of Data,” *ACM Transactions on Database Systems (TODS)*, vol. 1, no. 1, pp. 9–36, 1976.

[218] G. Harrison, *Next Generation Databases: NoSQL, NewSQL, and Big Data.* Apress, 2015.

[219] I. Robinson, J. Webber, and E. Eifrem, *Graph Databases*, 2nd ed., O’Reilly, Ed., 2014. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/B9780124071926000030

[220] T. Halpin, “Ontological Modeling : Part 1,” Tech. Rep., 2009. [Online]. Available: http://www.orm.net/

[221] A. Singhal, “Google Blog - Introducing the Knowledge Graph: things, not strings,” 2012. [Online]. Available: https://www.blog.google/products/search/introducing-knowledge-graph-things-not/

[222] D. Fensel, U. Simsek, K. Angele, E. Huaman, E. Karle, and O. Panasiuk, *Knowledge Graphs.* Springer Nature Switzerland AG, 2020. [Online]. Available: https://link.springer.com/book/10.1007%2F978-3-030-37439-6

[223] L. Ehrlinger and W. Wöss, “Towards a definition of knowledge graphs,” in *Proceedings of the 12th International Conference on Semantic Systems (SEMANTICS2016): Posters and Demos Track, CEUR Workshop Proceedings*, Leipzig, Germany, 2016. [Online]. Available: http://ceur-ws.org/Vol-1695/

[224] J. M. Gomez-Perez, J. Z. Pan, G. Vetere, and H. Wu, “Enterprise Knowledge Graph: An Introduction,” in *Exploiting Linked Data and Knowledge Graphs in Large Organisations*, J. Z. Pan, G. Vetere, J. M. Gomez-Perez, and H. Wu, Eds. Springer International Publishing Switzerland, 2017, pp. 1–14.

[225] J. Z. Pan, G. Vetere, J. Manuel, and G.-p. H. Wu, *Exploiting Linked Data and Knowledge Graphs in Large Organizations.* Springer, 2017.

[226] N. Noy, Y. Gao, A. Jain, A. Patterson, A. Narayanan, and J. Taylor, “Industry-scale knowledge graphs lessons and challenges,” *Commun. ACM*, vol. 62, no. 8, pp. 36–43, 2019.

[227] Q. He, B.-C. Chen, and D. Agarwal, “Building The LinkedIn Knowledge Graph,” pp. 1–7, 2016. [Online]. Available: https://engineering.linkedin.com/blog/2016/10/building-the-linkedin-knowledge-graph

[228] X. Wei and Y. Liao, “Contextualizing Airbnb by Building Knowledge Graph,” pp. 1–12, 2019. [Online]. Available: https://medium.com/airbnb-engineering/contextualizing-airbnb-by-building-knowledge-graph-b7077e268d5a

[229] D. Mccreary, “eBay’s Beam Knowledge Graph,” 2019. [Online]. Available: https://dmccreary.medium.com/ebays-beam-knowledge-graph-905cc42fdcfe

[230] A. Baker, “NASA’s Knowledge Graph,” pp. 1–9, 2017. [Online]. Available: https://www.stardog.com/blog/nasas-knowledge-graph/

[231] ——, “NASA: Stardog’s Going to Mars!” pp. 2–5, 2017. [Online]. Available: https://www.stardog.com/blog/nasa-stardogs-going-to-mars/

[232] D. Meza, “How NASA Finds Critical Data Through a Knowledge Graph?” pp. 3–11, 2017. [Online]. Available: https://neo4j.com/blog/nasa-critical-data-knowledge-graph/

[233] GraphAware, “GraphAware : ESA Case Study,” Tech. Rep., 2019.

[234] V. Kůs, "Hume in Space : Monitoring Satellite Technology Markets with a ML-powered Knowledge Graph," pp. 1–8, 2020. [Online]. Available: https://graphaware.com/products/hume/

[235] T. Sabat, "Comparing Grakn to Semantic Web," pp. 1–14, 2020. [Online]. Available: https://towardsdatascience.com/comparing-grakn-to-semantic-web-technologies-part-1-3-3558c447214a

[236] ——, "SQL vs . Graql : Modelling and Querying of Biomedical Data," pp. 1–11, 2018. [Online]. Available: https://blog.grakn.ai/sql-vs-graql-modelling-and-querying-of-biomedical-data-13a0d20ed0c1

[237] EoPortal Directory, "QARMAN." [Online]. Available: https://directory.eoportal.org/web/eoportal/satellite-missions/q/qarman

[238] G. Bailet, I. Sakraker, T. Scholz, and J. Muylaert, "Qubesat for Aerothermodynamic Research and Measurement on AblatioN." in *Proceedings of the 9th International Planetary Probe WorkShop*, Toulouse, France, 2012.

[239] L. Martin, B. Muller, P. J. Ortiz Suárez, Y. Dupont, L. Romary, E. de la Clergerie, D. Seddah, and B. Sagot, "CamemBERT: a Tasty French Language Model," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 2020, pp. 7203–7219. [Online]. Available: https://aclanthology.org/2020.acl-main.645

[240] M. Polignano, P. Basile, M. de Gemmis, G. Semeraro, and V. Basile, "AlBERTo: Italian BERT language understanding model for NLP challenging tasks based on tweets," in *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*, vol. 2481, Bari, Italy, 2019.

[241] P. Nayak, "MUM: A new AI milestone for understanding information," pp. 1–6, 2021. [Online]. Available: https://blog.google/products/search/introducing-mum/

[242] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Information Fusion*, vol. 58, pp. 82–115, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253519308103

[243] R. Kurzweil, *The Age of Intelligent Machines.* MIT Press, 1990.

[244] A. Turing, "Computing Machinery and Intelligence," *Mind*, vol. 49, no. 236, pp. 433–460, 1950.

[245] J. Weizenbaum, "ELIZA—A Computer Program For the Study of Natural Language Communication Between Man And Machine," *Communications of the ACM*, vol. 9, no. 1, pp. 36–45, 1966.

[246] N. Bostrom, *Superintelligence: Paths, Dangers, Strategies.* Oxford University Press, 2014.

[247] F.-H. Hsu, "IBM's Deep Blue Chess grandmaster chips," *IEEE Micro*, vol. 19, no. 2, pp. 70–81, 1999.

[248] D. A. Ferrucci, "Introduction to "This is Watson"," *IBM Journal of Research and Development*, vol. 56, no. 3, p. 235–249, 2012.

[249] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, 2016. [Online]. Available: http://dx.doi.org/10.1038/nature16961

[250] A. Burkov, *The Hundred-Page Machine Learning Book*, 2019.

[251] M. B. Hoy, "Alexa, Siri, Cortana, and More: An Introduction to Voice Assistants," *Medical Reference Services Quarterly*, vol. 37, no. 1, pp. 81–88, 2018. [Online]. Available: https://doi.org/10.1080/02763869.2018.1404391

[252] L. Matthies, M. Maimone, A. Johnson, Y. Cheng, R. Willson, C. Villalpando, S. Goldberg, A. Huertas, A. Stein, and A. Angelova, "Computer vision on Mars," *International Journal of Computer Vision*, vol. 75, no. 1, pp. 67–92, 2007.

[253] J. R. Firth, "A synopsis of linguistic theory, 1930-1955," pp. 1–32, 1957.

[254] K. Lund and C. Burgess, "Producing high-dimensional semantic spaces from lexical co-occurrence," *Behavior Research Methods, Instruments, and Computers*, vol. 28, no. 2, pp. 203–208, 1996.

[255] M. Baroni, G. Dinu, and G. Kruszewski, "Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014*, vol. 1. Association for Computational Linguistics, 2014, pp. 238–247. [Online]. Available: https://www.aclweb.org/anthology/P14-1023

[256] P. Goyal, S. Pandey, and K. Jain, "Word Vector Representations," in *Deep Learning for Natural Language Processing: Creating Neural Networks with Python*. Berkeley, CA: Apress, 2018, pp. 75–118. [Online]. Available: https://doi.org/10.1007/978-1-4842-3685-7_2

[257] R. Collobert and J. Weston, "A unified architecture for natural language processing," in *Proceedings of the 25th International Confer- ence on Machine Learning*. Helsinki, Finland: Association for Computing Machinery, 2008, pp. 160–167. [Online]. Available: https://dl.acm.org/doi/10.1145/1390156.1390177

[258] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, pp. 1532–1543.

[259] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching Word Vectors with Subword Information,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017. [Online]. Available: https://doi.org/10.1162/tacl_a_00051

[260] G. Berardi, A. Esuli, and D. Marcheggiani, “Word embeddings go to Italy: A comparison of models and training datasets,” in *Proceedings of the 6th Italian Information Retrieval Workshop*, Cagliari, Italy, 2015, p. 8.

[261] A. Sutton and N. Cristianini, “On the Learnability of Concepts,” in *Proceedings of the Artificial Intelligence Applications and Innovations (AIAI)*, I. Maglogiannis, L. Iliadis, and E. Pimenidis, Eds. Springer International Publishing, 2020, pp. 420–432.

[262] L. Yeganova, S. Kim, Q. Chen, G. Balasanov, W. John Wilbur, and Z. Lu, “Better synonyms for enriching biomedical search,” *Journal of the American Medical Informatics Association*, vol. 27, no. 12, pp. 1894–1902, 2020.

[263] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 2227–2237. [Online]. Available: https://aclanthology.org/N18-1202

[264] Q. Liu, M. J. Kusner, and P. Blunsom, “A Survey on Contextual Embeddings,” pp. 1–14, 2020. [Online]. Available: http://arxiv.org/abs/2003.07278

[265] D. Bahdanau, K. H. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track*, San Diego, CA, USA, 2015, pp. 1–15.

[266] M. T. Luong, H. Pham, and C. D. Manning, “Effective approaches to attention-based neural machine translation,” in *Proceedings of the 2015 Conference*

on Empirical Methods in Natural Language Processing (EMNLP). Lisbon, Portugal: Association for Computational Linguistics, 2015, pp. 1412–1421. [Online]. Available: https://aclanthology.org/D15-1166

[267] J. Vig, "A multiscale visualization of attention in the transformer model," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations.* Florence, Italy: Association for Computational Linguistics, 2019, pp. 37–42. [Online]. Available: https://www.aclweb.org/anthology/P19-3007

[268] J. Alammar, "The Illustrated Transformer," 2018. [Online]. Available: https://jalammar.github.io/illustrated-transformer/

[269] Y. Le Cun and I. Misra, "Self-supervised learning: The dark matter of intelligence," pp. 1–16, 2021. [Online]. Available: https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/

[270] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving Language Understanding by Generative Pre-Training," Tech. Rep., 2018.

[271] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," OpenAI, Tech. Rep., 2019.

[272] R. Sennrich, B. Haddow, and A. Birch, "Neural Machine Translation of Rare Words with Subword Units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Berlin, Germany: Association for Computational Linguistics, 2016, pp. 1715–1725. [Online]. Available: https://aclanthology.org/P16-1162

[273] B. Al-Salemi, M. Ayob, S. A. M. Noah, and M. J. A. Aziz, "Feature selection based on supervised topic modeling for boosting-based multi-label text categorization," *Proceedings of the 2017 6th International Conference on Electrical Engineering and Informatics (ICEEI)*, pp. 1–6, 2017.

[274] J. Yun and Y. Geum, “Automated classification of patents: A topic modeling approach,” *Computers and Industrial Engineering*, vol. 147, no. July, p. 106636, 2020. [Online]. Available: https://doi.org/10.1016/j.cie.2020.106636

[275] W. Sriurai, “Improving Text Categorization By Using A Topic Model,” *Advanced Computing: An International Journal (ACIJ*, vol. 2, no. 6, pp. 21–27, 2011.

[276] Z. S. Harris, “Distributional Structure,” *WORD*, vol. 10, no. 2-3, pp. 146–162, 1954. [Online]. Available: https://doi.org/10.1080/00437956.1954.11659520

[277] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, “Indexing by latent semantic analysis,” *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990. [Online]. Available: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-4571%28199009%2941%3A6%3C391%3A%3AAID-ASI1%3E3.0.CO%3B2-9