Technische Universität München

TUM School of Engineering and Design

Graphen-gestützte Systemanalyse

Florian Patzwahl, M.Sc.

Vollständiger Abdruck der von der TUM School of Engineering and Design der Technischen Universität München zur Erlangung eines

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

genehmigten Dissertation.

Vorsitz:		Prof. Dr. rer. nat. Martin Werner
Prüfer*innen der Dissertation:	1.	Prof. Dr. rer. nat. Ulrich Walter
	2.	Prof. Dr.-Ing. Kristin Paetzold-Byhain

Die Dissertation wurde am 13.03.2023 bei der Technischen Universität München eingereicht und durch die TUM School of Engineering und Design am 21.06.2023 angenommen.

Technical University of Munich

TUM School of Engineering and Design

Graph-Enhanced Systems Analysis

Florian Patzwahl, M.Sc.

One of the biggest challenges for a systems engineer of a large complex project is to "bring order from chaos."

Chris Hardcastle, Systems Engineering and Integration Manager, NASA's Constellation Program, Johnson Space Center

Für Markus, ohne dessen Unterstützung, Begeisterung, Mentoring und Glaube an die Wissenschaft ich diesen Weg nicht eingeschlagen hätte.

Und für meine Mutter, die mich schon als Kind am Esstisch jeden Tag für Wissenschaft und Forschung begeisterte.

Danksagungen

Die vergangenen Jahre waren eine Zeit mit vielen Höhen und einigen Tiefen und diese Seite ist all jenen gewidmet, die mich während beidem unterstützt und ertragen haben. Zunächst möchte ich mich bei Prof. Walter bedanken. Ihre unvergleichliche Art den Lehrstuhl für Raumfahrttechnik zu führen machte den LRT zu einem wahrhaft besonderen, kreativen, geschützten Ort, der hoffentlich im neuen ASG-Department vielen als Blaupause und Inspiration dient. Mittlerweile bin ich seit einem Jahr aus dem Lehrstuhlalltag heraus und erkenne mit diesem Abstand noch deutlicher, wie einzigartig die Umgebung ist, die Sie dort für Studierende, Forschende und Mitarbeitende geschaffen haben. Mein Dank geht auch an Dr. Martin Rott und Uta Fellermair, ohne deren Unterstützung und Voraussicht ich mir bei manchem bürokratischen Kunststück die Knie wund geschlagen hätte. An dieser Stelle sei auch Prof. Paetzold gedankt, für Ihre unkomplizierte und pragmatische Bereitschaft, eine Thesis zu begutachten von der Sie vorher noch nie gehört haben.

Ein warmherziger Dank geht auch an meine Kollegen, im speziellen David, Jonis, Martin D, Nico, Gerdi und Laura, die mir aus manchem Motivationstief wieder aufgeholfen haben und sich oft stundenlang mit mir über meine Forschung unterhalten haben. Danke dafür, dass es bei euch immer die metaphorische Schulter zum anlehnen gibt. Ein großer Dank geht auch an das MOVE-Team. Das Vertrauen, der unermüdlichen Einsatz, das Herzblut dass sie täglich in Ihre Arbeit stecken und die resultierende Enthropie sind die Inspiration für diese Thesis. Ein besonderer Dank geht an Max, der sich mit Leidenschaft besagter Enthropie widmete. Deine Implementierungen in Software und dein Wille auch den letzten Ungereimtheiten nach zu gehen haben dieser Forschung mit den Weg bereitet.

Bei einer Thesis wie dieser ist auch der Weg das Ziel. Umso wichtiger, sich Gedanken über die Richtung zu machen. Ein ganz besonderer Dank gebührt meinen beiden Mentoren Martin und Markus. Euer konstruktiver Rat war stets willkommen und aufschlussreich. Im Fall Martins um die Zeit nicht aus den Augen zu verlieren und dem Vorhaben als ganzes Struktur zu verleihen. Im Fall Markus um von pragmatischen Ideen zu tiefergehenden theoretischen Fragen zu gelangen. Danke, dass ihr an mich geglaubt habt und mir den Mut gemacht habt diese Forschungsarbeit umzusetzen. Danke auch dafür, dass ihr mich mit gezielten Nachfragen immer wieder motiviert habt weiter zu machen. Auch wenn Markus nicht mehr unter uns weilt, wird die Überzeugung mit der er Wissenschaft und Lehre betrieb mir immer eine Motivation und Inspiration bleiben.

Großer Dank gebührt auch meinen Eltern. Schon von klein auf beinahe täglich mit meiner Mutter über die Chemie des Alltags zu diskutieren hat mich wohl überhaupt erst auf den Weg in Richtung Wissenschaft und Forschung gebracht.

Zu aller letzt geht mein Dank an Lucie. Dafür dass du mich all die Zeit ausgehalten hast, mir immer beigestanden bist und ich die kleinen und großen Sorgen und Erfolge mit dir teilen darf.

Kurzfassung

Die fortschreitende Digitalisierung von Entwicklungsprojekten bringt sowohl Herausforderungen als auch Chancen mit sich. Eine Herausforderung der Digitalisierung ist die zunehmende Menge an Informationsquellen innerhalb eines Entwicklungsprojekts. Neben dem öffentlichen Teil des Internets wird eine Vielzahl von Informationsaustausch- und Speichersystemen von Entwicklungsteams eingesetzt, um Dokumente, Arbeitspakete und Design des zu entwickelnden Systems zu organisieren. Während Informationsaustausch- und -speichersysteme das Leben im Moment bequemer machen, wird das Auffinden von Informationen mit jeder zusätzlichen Quelle schwieriger. Da sich der Kontext einer bestimmten Information auf immer mehr Informationsaustausch- und -speichersysteme verteilt, wird das Auffinden des vollständigen Kontexts einer Information ineffizient. Somit steigt die Wahrscheinlichkeit, dass relevanter Kontext für eine korrekte Beurteilung fehlt, wie beispielsweise ob die Information noch aktuell ist. Untersuchungen des Autors und der Literatur zeigen, dass Ingenieurinnen und Ingenieure im Durchschnitt etwa 30% ihrer täglichen Arbeitszeit mit dem Abrufen von Informationen verbringen. Im Falle der Entwicklung von Raumfahrzeugen kann dies zu schwerwiegenden Fehlern führen. Daher wird der Stand der Technik bei Wissensmanagementsystemen diskutiert und analysiert, wobei der Schwerpunkt auf den Prozessen und Anforderungen bei der Entwicklung von Raumfahrzeugen liegt, mit dem Ergebnis, dass neue Wege zur Korrelation und zum Abrufen von Informationen notwendig sind, um schnellere und sicherere Entwicklungen zu gewährleisten.

Eine Technik, die sich auf eine bessere Korrelation von Informationen innerhalb eines Entwicklungsprojekts konzentriert, ist Model Based Systems Engineering. Studien zeigen, dass die Erwartungen an die Vorteile von Model Based Systems Engineering hoch sind, aber auch, dass zwischen den Erwartungen und dem Stand der Technik eine Lücke klafft, insbesondere in Bezug auf die Bereitschaft zu Veränderungen und die Fähigkeiten der Werkzeuge. Darüber hinaus werden fehlende Modellierungsziele und Modellierungsstrategien als Haupthemmnisse für eine breitere Anwendung des Model Based Systems Engineering genannt. Das völlige Fehlen von Veröffentlichungen in diesem Bereich zeigt, dass einer der Bereiche, in dem Model Based Systems Engineering derzeit noch keinen Nutzen bringt, die Montage-, Integrations- und Testphase bei der Entwicklung von Raumfahrzeugen ist. Der Stand der Technik zeigt jedoch vielversprechende Ergebnisse bei der Übertragung von SysML-Modellen in Graphdatenbanken für Analyseaufgaben.

Daher wird eine Reihe von Zielen in Form von typischen Analyseaufgaben skizziert, die sowohl für Reviews als auch während der Integration und des Testens von Raumfahrzeugen charakteristisch sind. Die Analyseaufgaben basieren auf dem typischen Informationsbedarf während der Entwicklung, Integration und Erprobung von Raumfahrzeugen, wobei der Schwerpunkt auf Aufgaben liegt, die typischerweise fehleranfällig und ineffizient sind und ein tiefgreifendes Wissen und Verständnis des Systems erfordern. Anschließend wird ein Graphschema entwickelt, mit dem SysML-Modelle in Graphdatenbanken übersetzt werden können, basierend auf den zuvor definierten Analyseaufgaben. Eine entsprechende Modellierungsstrategie wird entwickelt und an zwei realen Weltraummissionen, der MOVE-II Mission und dem Extendable Modular Power System, getestet. Anhand der Anwendungsfälle dieser zwei Systeme werden entsprechende Abfragen entwickelt und exemplarische Ergebnisse bereitgestellt. Es werden Änderungen des Graphschemas in Abhängigkeit von den Analysezielen vorgestellt. Es wird gezeigt, dass Analyseabfragen, die für ein System entwickelt wurden, so parametrisiert werden können, dass sie für ein anderes System wiederverwendet werden können. Die Anwendbarkeit des Schemas auf ein breiteres Spektrum von Systemen wird durch die Modellierung einer hypothetische Mondbasis getestet, mit viel versprechenden Ergebnissen.

Ausgehend von der Graphübersetzung des SysML-Modells wird ein kombinierter Ansatz des modellbasierten und dokumentenbasierten Systems Engineering diskutiert, der die Anpassungsfähigkeit von Graphdatenbanken und den hohen Digitalisierungsgrad bei der Entwicklung von Raumfahrzeugen nutzt. Anhand der Dokumentation und weiterer verfügbarer digitalisierter Informationen der MOVE-II-Mission wird untersucht, inwieweit Dokumente und andere Entwicklungsartefakte ihren Gegenstücken im SysML-Modell zugeordnet werden können.

Abschließend werden die Ergebnisse der Arbeit diskutiert und mit dem aktuellen Stand der Technik verglichen. Es wird eine Analyse der möglichen Auswirkungen dieser Arbeit erstellt, einschließlich einer kritischen Analyse der Chancen für die Einführung eines solchen Systems bei der Entwicklung eines Raumfahrzeugs. Die Analyse umfasst Aspekte wie geistiges Eigentum, den Schutz persönlicher Daten von Entwicklern und die Ergebnisse der Studien zur Anwendung von Model Based Systems Engineering in der Literatur. Es wird ein Ausblick auf mögliche Themen für zukünftige Forschung skizziert.

Abstract

The ongoing digitalization of development projects brings challenges as well as opportunities. A challenge of digitization is the increasing amount of informational sources within a development project. In addition to the public part of the internet, a multitude of information exchange and storage systems are deployed by development teams to organize documents, work packages, and the design of the system under development. While each of these information exchange and storage systems may make life more convenient at the moment, finding information becomes more challenging with every additional source. As the context to a certain piece of information is spreading over an increasing amount of information exchange and storage systems, retrieving the full context of a piece of information becomes inefficient, and the chance increases of missing relevant context that allows judgment such as whether the information is still up-to-date. Studies by the author as well as the literature show that the average amount of time engineers spend on retrieving information sums up to roughly 30% of their daily work time. In the case of spacecraft development, this may lead to severe failures. Therefore, the state of the art in knowledge management systems is discussed and analyzed with a focus on the processes and requirements of spacecraft development, concluding that new ways to correlate and retrieve information are necessary to ensure faster and safer developments.

One technique focused on better correlation of information within a development project is Model Based Systems Engineering. While studies show high expectations for the benefits of Model Based Systems Engineering, the same studies show a gap between expectations and the state of implementation, especially regarding reluctance to change and tool capabilities. They further identify lacking modelling goals and modelling strategies as key inhibitors to a wider spread application of Model Based Systems Engineering. A total lack of publications in the area shows that one of the fields Model Based Systems Engineering is currently still lacking to yield benefits is the Assembly Integration and Testing phase of spacecraft developments. However, the state-of-the-art shows promising results of transferring SysML Models into graph databases for analysis tasks. Therefore, a set of goals in the form of typical analysis tasks which are conducted for reviews as well as during spacecraft assembly integration and testing is outlined.

The analysis tasks are based on typical informational needs during spacecraft development, integration and testing with a focus on tasks that are typically failure prone, inefficient and require intricate knowledge and understanding of the system. Next, a graph schema by which SysML models can be translated into graph databases is developed, based on the previously defined analysis tasks. A corresponding modelling strategy is developed and tested on two real-life space missions, the MOVE-II Mission and the Extendable Modular Power System. Respective queries are developed, and exemplary results are provided, based on the use case of these two systems. Alterations of the graph schema depending on the analysis goals are presented. It is shown that analysis queries developed for one system can be parametrized to be reused on another system. The applicability of the schema on a wider range of systems is tested by applying the modeling schema to a hypothetical Lunar base, which works well.

Using the graph translation of the SysML Model as a baseline, a combined approach of Model-Based- and Document-Based Systems Engineering is discussed, which leverages the adaptability of graph databases and the high degree of digitization applicant in spacecraft developments. The degree by which documents and other development artifacts can be attributed to their counter-parts in the SysML Model is studied on the documentation and further available digitized information of the MOVE-II mission.

Finally, the results of the thesis are discussed and compared against the state of the art. An analysis of the possible impact of this thesis is provided, including a critical analysis of the chances of introduction of such a system in an actual spacecraft development. The analysis includes aspects such as intellectual property, personal data protection of developers and the findings of the studies on Model Based Systems Engineering application in the literature. Possible topics for future research are outlined in the end.

Table of Contents

List of Figures

List of Tables

List of Queries

List of Acronyms

BIO-ECLSS	Biological Environmental Control and Life Support System
BPA	Brine Processing Assembly
CCAA	Common Cabin Air Assembly
CDRA	Carbon Dioxide Removal Assembly
ECLSS	Environmental Control and Life Support System
ECSS	European Cooperation for Space Standardization
EMPS	Extendable Modular Power System
ESA	European Space Agency
ISRU	In-Sitru Resource Utilization
MBSE	Model Based Systems Engineering
MOVE-II	Munich Orbital Verification Experiment II
NASA	National Aeronautics and Space Administration
PBR	Photo Bio Reactor
PDF	Portable Document Format
RDF	Resource Description Framework
SRCA	Sabatier CO2 Reprocessing Assembly
stm	State Machine
SysML	Systems Modeling Language
UPA	Urine Processing Assembly
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WPA	Water Processing Assembly

1. Introduction

"Over the next 10 years we will reach a point where nearly everything will become digitized."[1]

Satya Nadella, 2013

When forecasting over a timeframe of ten years, the line between predictions and prophecies becomes blurred to the point of not being recognizable anymore. Without ever intending to discuss the accuracy of Mr. Nadella's statement, it cannot be argued against the significant advances digitization has been making over the past years and will continue to do so in the near future. Digitization has been transforming our work place over the past decades in ways that were unimaginable beforehand. However, it also comes with new challenges. With the rise of digitization, finding a specific piece of information and ensuring its correctness and integrity became enough of a challenge, that the term *infodemic* was coined as early as 2003 [1], which describes an epidemic of information.

Comprehensive research was conducted over the past five years to investigate the spread of false information in public webspaces such as Twitter or Facebook ([2]–[4]). While ample solutions exist for finding public information on the World Wide Web, the same cannot be said for information outside the public domain, for example the digital environment of a technical development project. On the other hand, information inside a technical development project is rarely generated with the intent to mislead others.

In 2004, the National Institute for Science and Technology of the United States of America published a report estimating the cost for dealing with the loss of data and information in development projects of the automotive supply chain at 1 billion dollars per year [5]. According to analyses of Levison et al. in 2004 and 2009 [6], [7], the problem of inefficient and ineffective information management is not limited to the automotive industry. They were able to trace back the loss or partial loss of five European Space Agency (ESA)- and or National Aeronautics and Space Administration (NASA) missions to insufficient management of critical information. Those five missions were in the timeframe of a mere 4 years, showing that such problems are not the exception. According to Levison, any of the five missions could have been saved by better information management, i.e., decisions were made by personnel that was not informed well enough and hence led to critical mistakes, although the information they lacked was available in the digital archive of the projects or known to other project members. Since the last of these missions' failure, over 20 years passed. It is therefore time to ask whether the problem of insufficient information management was solved since.

A very recent study by Berquand et al. on information management at ESA's concurrent design facility investigated the amount of time spent on information gathering in concurrent design studies [8] Interviewing 47 engineers at the concurrent design facility, the average time spent on gathering information amounts to over 30%. Asked where they find the most valuable information, also over 30% of the interviewees answered "in discussion with their colleagues". No statement is made on whether the time the interviewees spend on answering information requests of their colleagues is already included, which could lead to even higher shares of work time spent over the whole development team on gathering information.

The results of this study lead to multiple conclusions:

- A third of the working time is a sizable portion of every day's life for a development engineer. ESA's

[1]Blog Post by Satya Nadella, CEO of Microsoft Corp.

technology strategy published in November 2019 [9] formulates the goal to reduce the development time of spacecraft by 30% until 2023. A reduction of the time spent to find the correct information is hence mandatory to reach this goal.

- If the most valuable information is found when discussing with colleagues, there have to be reasons why written information is not considered as valuable a resource as questions answered in person.

Considering the time spent looking for information and the fact, that written information seems less trusted than questions answered in person, one may conclude, that a sufficient solution to the problem of information management has yet to be found. Adding the findings of Levison et al. (compare [6], [7]) about information miss-management leading to mission failures, one can see that the above stated problem imposes risk to a space mission's technical success as well as its schedule and funds. On the other hand, the interview conducted by Berquand was based on members of a single institution, working in the same facility, hence the scope of the available information is too narrow to draw general conclusions. Nevertheless, it motivates to further investigate how much time is spent at other aerospace institutions and in other mission phases with information storing and retrieval.

In 2017, Robillard et al. published an in-depth analysis into the current state of the art of tools that help to document and transfer knowledge for software developments [10]. They argue, that curated, formal documents are not seen as sufficient to develop and maintain systems by developers. According to them, perusing and interpreting information is further complicated by the heterogeneous nature and large quantity of online resources. They defined online resources as publicly available information on the internet as well as the collaborative development tools like project management systems, team communicator applications, email or centralized code repositories. The sum of such tools used in a specific development project makes up the digital environment of the development. Interdisciplinary projects such as a spacecraft development necessarily have a diverse digital environment. The environment provides the developers with a high number of possibilities on where to store information such as merge requests, email, chat-tools, comments in design files, readmes to the design files, review documentation, task descriptions, anomaly reports, etc.

Developers looking for specific information do not yet know where the information is stored, or if the specific information they are looking for is even recorded yet. The result is that developers often consult their colleagues. In 2007, Ko et al. [11] conducted an in-depth analysis of the behavior of developers at Microsoft. They looked into what information developers seek, where they seek this information and what prevents them from finding information. The result of the observation of 49 developers across the company was, that deferred searches most often concern the intent behind a specific design feature and that the main reason for developers having to defer tasks lies in the unavailability of coworkers who are the only source for certain information.

Ko et al. also found that more than 30% of the working time of developers is spent on gathering information. Their findings correlate well to those of presented by Berquand et al. in 2019. One can thereby draw the conclusion, that since Ko's publications in 2007, efficiency in information retrieval has not outpaced the growing complexity of the development environment [11], [12].

The publications by Ko as well as by Berquand strongly support, that retrieving information from coworkers is one of the central processes in a development. Ko's research however is strongly limited to software development, while Berquand shows that the same can be said about the early stages of spacecraft design.

How much the direct exchange between developers contributes in the later stages of a spacecraft's life cycle is yet to be assessed.

1.1. The Digitization of Systems Engineering

The central coordination of any spacecraft development employs the discipline of systems engineering. While a wide variety of methodologies and tool sets exist for systems engineering [13]–[17], one especially embraces the digitization of engineering environments; The Methodology of Model Based Systems Engineering. While studies reveal high expectations on the introduction of Model Based Systems Engineering by managers and developers, the current state of implementation shows that major inhibitors still dominate the field [18]–[21].

The following section is already handed in for publication by the author in [22].

In 2007, the International Council on Systems Engineering defined Model Based Systems Engineering as "the formalized 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." [23, p. 15]

In 2019, NASA published a survey with 50 participants from the industry, academia, US-government agencies and tool vendors for systems engineering tools on the topic of the current state of systems engineering at their own institution or as perceived at partnering entities (cf. [21]). The study identified an increased application of Model Based Systems Engineering (MBSE) as the number 3 factor on improving overall performance, after improving training of systems engineers and improving their domain specific knowledge. Asked about the expected benefits, 63% expect a reduction of 30% to 50% regarding the development cycle time. While encouraging the use of MBSE, the study also found the majority of participants reporting less than 25% adoption of MBSE in projects at their workplace. 33% of the participants across all disciplines see insufficiency of MBSE tools as a key weakness in their application of systems engineering. The only factor named more often are cultural issues with 46%, which the authors describe as people being reluctant to change from document based processes ([21]).

I chose this study for the introduction to this paper for the following reasons: According to the study, systems engineers across industry, public agencies and academia show a high level of expectations on MBSE. At the same time, they admit that the tools are inadequate. The acceptance of tools that are viewed as inferior to their current working environment is generally low. This is maybe one of the major factors for the by far smaller percentage of adaption of MBSE, than the expected improvement through its application would suggest.

The successful introduction of new products and processes always requires understanding the users' needs. Roughly around the same time as NASA's study was conducted, the ESA started its "Model Based for Systems Engineering" initiative, which aims at increasing the usability and interoperability of MBSE and MBSE tools within the European space sector. A report on perceived user needs, published by the initiative in 2020 sums up the current state of interoperability between tools as low, resulting in unnecessary duplicate work. The report identifies 15 key user needs that MBSE should deliver. One of the key user needs is to "ensure the consistency, completeness and feasibility of requirements and design", noting that "especially the functional complexity must be kept in [sic] control"[24, p. 11]. Another key user need is to "structure the knowledge about the system in such a way that the relations between the knowledge elements are established and traceable"[24, p. 11].

Looking at yet another survey on the adoption of MBSE, the report *Benchmarking the Benefits and Current Maturity of Model-Based Systems Engineering across the Enterprise* by the Systems Engineering Research Center asked 240 individuals across academia and industry what they see as the largest obstacles in MBSE adoption. "MBSE methods and processes" is cited as number one obstacle by the report [18, p.20]. [21] supports these findings; 88% of participants identified "Purpose and Scope Definition" as a major challenge in the adoption of MBSE.

Combining the results of the three reports gives insight into some obstacles to rolling out MBSE on a wider scale; MBSE tools shall enable users to ensure the completeness, consistency and feasibility of their development and a third of the participants in NASA's study identify insufficiency of MBSE tools as a key weakness to their application of MBSE. The conclusion lies near, that the tools are not yet fully up to the task of assessing completeness, consistency and feasibility. Combined with the findings of [18], the reports show tools and modelling strategies as major factor for a broader acceptance of MBSE. Part of the tools' inadequacy in my experience, which was gained over half a decade of working and teaching with products of NoMagic and Sparx Systems, is their inability to process complex queries on models. Starting at any given element of a model, one can usually only follow it for one hop, i.e., to the end of any of its direct relations, or along a single relation type. Although any consistent model contains complex modelling patterns, these can usually not be followed within standard commercial tools.

1.2. The Vicious Cycle of Project Delays

A survey from 2017 asked more than 100 managers and engineers what they see as the biggest challenges they face in managing projects. The top three challenges for managing projects, according to the report were managing costs, hitting deadlines and sharing information across teams, with over 40% consent each. "Visibility into what my team and others are working on" had a consent of over 38% as a top challenge and was on 6th place in the ranking. [25]

With hitting deadlines and managing costs as the biggest challenges one can draw the conclusion, that it is not uncommon for projects to be delayed and experience challenges in their overall costs. Also, information management is a key concern. Combining those three challenges with insights from other researchers (compare [10], [11]), a theory forms regarding the connection between delay, costs and information management. Project delay, costs and quality of information management directly affect each other. Once projects are delayed, tasks that are not vital are set to lower priority. Hence, documenting is often postponed in these situations. If projects overrun their budgets, cost-contributors that are not vital to the project are being reduced. Updating documentation requires the input of resources and hence becomes unattractive. [25]

Figure 1-1 shows the vicious cycle of project delay due to neglected documentation, which is explained in further detail in the following.

This theory gets some support by the research conducted by Robillard et al. [10]. According to their research, creating documentation is associated with high cost and a low immediate return on investment. They also state that documentation is not technically critical to the construction of software, which can be transferred to the construction of embedded systems in general. Being resource-intensive and not technically critical (i.e.,

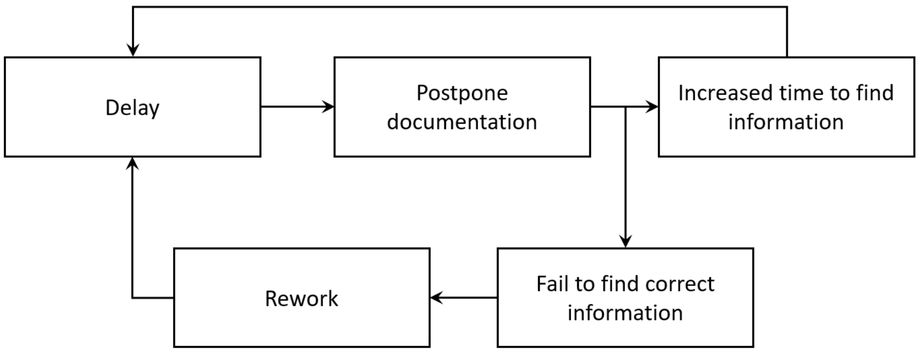

Delay
Postpone documentation
Increased time to find information
Rework
Fail to find correct information

Figure 1-1.: Vicious cycle of not documenting.

the software as well as any embedded system also works without being documented), the temptation to push documentation to a later phase in favor of delivering the product on time arises.

However, as presented before, developers spend about a third of their time acquiring information (assuming the findings of Berquand as well as Ko can be transferred to the general amount of time spent acquiring information). While this may of course differ from person to person and project to project, it is undeniably a considerable share of a developer's everyday work time. If documentation is not maintained, the only way to acquire accurate information is by direct correspondence with the respective developers.

Thereby, the share of information that is acquired by asking colleagues increases.

Information acquired by direct correspondence has overall higher costs. To acquire information, two participants are required: the person looking for information and the entity providing it. This entity can either be another person or a system containing the information. In case of project documentation, where the resources required to generate the documentation have to be considered, one may also conclude that two persons are required, the person who wrote the documentation and the person looking for information. This stays true independent of the way the information is shared. Retrieving information by direct correspondence is less efficient than retrieving information from written documentation, however.

- First, the person holding the information has to be present in order to retrieve it. Once information is available in writing, this is not required anymore.
- Also, looking for the person that might be able to help takes time. Often this takes even more time than asking them. After asking a first person, one is referred to a second person, which refers to a third person. If the person holding the information has left the project, the information may be non-recoverable.
- When someone else looks for the same information, the process repeats itself and does not get any more efficient. The information providing person again has to be available and once available again has to invest the same amount of time to explain it as the first time.

According to Ko, unavailability of coworkers with critical knowledge is the reason with the largest share for not finishing a task a developer set out to [11].

On the other hand, McChesney and Gallagher, who observed informational processes in two software development teams over several weeks come to the conclusion, that some degree of informal direct interaction to retrieve information is a relevant and valuable asset of any development team, a body of knowledge and customs derived from common lessons learned on the overall team that does not develop if everyone were to work on their own [26].

The opposite of retrieving any information by personal correspondence is documenting the system under development as well as possible. This approach is similarly inefficient, as the more you document, the higher the percentage of documentation that will never be read by anyone. During a development project, another problem arises with this approach. If the system under development changes, all the documentation needs to be updated.

The optimum regarding efficiency lies somewhere in the middle. If every possible aspect of a system is documented, maintaining the documentation becomes overly inefficient, including the risk of the documentation becoming untrustworthy due to a too high share of outdated documentation. If too little information is retrievable from the project's archive, too much time has to be spent on acquiring the information directly from the developers, including the risks of the knowledge-holder having left the project or simply not remembering anymore. How much should be formally documented depends on team size, how often changes occur that potentially lead to outdated documentation and formal documentation requirements by stakeholders.

The decreasing efficiency creates further delays and costs, causing a vicious cycle. Once maintaining documentation is pushed back to later project phases, another problem arises. Knowing, that if something changes in the design, the documentation is not updated accordingly, confidence is lost in the existing documentation. To assess the correctness of documentation, the responsible developer again has to be contacted. I.e., even where documentation exists, it is not trustworthy anymore, since it is not apparent whether the design was updated after the documentation was written.

Thus, a vicious cycle forms. Once the project has a certain delay, documentation requirements are relaxed to allow faster development. Thereby more documentation becomes outdated. The more outdated documentation exists, the more developers need to engage in inefficient direct information retrieval, causing further delays.

It is important to note, that this cycle can start independent of the previous documentation strategy. Projects with a high initial documentation load easily come to the point where maintaining the documentation is too costly. Projects with initially low documentation requirements may already run into delays due to the inefficiency caused by constantly needing to acquire information directly, thus never getting to the point where documentation can be written.

Looking at this situation the following questions arise:

- How can the abundant information available in a modern development project be organized to allow judging when a piece of documentation becomes outdated?
- How can information shared between developers upon direct requests be made retrievable?
- What kinds of information are developers looking for that require the knowledge of their peers even when accurate documentation is available?

1.3. Derived Fields of Work

Although the above presented hypothesis of the vicious cycle of project delays follows a valid argumentation, proving its soundness requires more data and deeper inspection, which shall be provided by the means of a survey.

Narrowing the field of work for gathering the data, the focus shall be set on small satellite projects. Although small satellites are quite complex, the projects are typically in the range of less than 5 years, with a nominal development time of two to three years, making for challenging schedules. Working in the field, the author has the necessary competence and access to information to properly assess related problems.

Therefore, data on project delays, the number of software tools in use in a project, the state of its documentation and the information retrieval behavior of developers shall be assessed with help of a survey. The assessment of information retrieval behavior shall also cover the employed means of communication.

Focusing on spacecraft systems, the survey shall focus on the development as well as the operations of small spacecraft. In comparison to other systems like cars, industrial plants etc. the operation of a spacecraft has special characteristics; once in orbit, a spacecraft cannot be repaired anymore and many aspects of a space mission cannot be tested properly before its deployment in space. Therefore, operators have special needs with regard to how deep their understanding of the spacecraft has to go. Any fault in the operation could cause the spacecraft to fail. Also, due to the intensive qualification phase, the regular occurrence of launch delays, and the distribution of development to subcontractors and sub-subcontractors, reaching the responsible developer to clear up questions about the documentation is often difficult if at all possible. However, these are just assumptions, founded on personal conversation with the operators of various spacecraft. To assess the scope and depth of the problem, a survey amongst operators of various institutions is necessary.

A thorough introduction to the terminology of information management and knowledge management is in order to establish a common ground for the following chapters. Furthermore, considering that several years of ongoing digitization passed since the research conducted by [6], [7], [11], [12], a discussion and comparison of current and past systems for information management and knowledge management is in order. While the focus of this should be on information management and knowledge management in technical development projects, a broader horizon is taken to see how other information intensive professions cope with the infodemic.

Potential areas of transfer for information management approaches from other industries shall be investigated and applied where appropriate. A new approach to information management for the spacecraft development cycle is synthesized and presented. This includes the information insertion and extraction processes, based on typical knowledge intensive tasks in a spacecraft development.

1.4. Overview

The thesis is structured in three parts. In the first part, which ranges from this introduction to Chapter 5, problems and challenges of information management are laid out (Chapter 2), a gap analysis is conducted (Chapter 3), a research layout formulated (Chapter 4) and the above-mentioned survey is conducted and interpreted (Chapter 5).

In the second part (Chapter 6 and 7) a tool and corresponding modeling guideline are developed that allows to automatize information gathering for knowledge intensive tasks with a focus on the needs of later mission lifecycle phases, based on SysML models. It builds on the needs, challenges and choke points identified in Chapter 2 and Chapter 5. The tool shall answer questions such as:

- Which parts of a system are affected by a requirement non-conformance? Either because they directly relate to the requirement, or to a requirement from which the not-fulfilled requirement was derived or because they interface a component which satisfies any of these requirements?
- Which path does any specifiable item take from its sensor down to the operator's screen?
- What happens in case of an electrical short of a specific spacecraft component? Which other components are offline? Which components do not receive their expected data inputs anymore? Which telemetry cannot be collected anymore due to the short?
- In case a certain system condition cannot be met, which alternative paths allow to reach a specified state? Which states cannot be reached anymore due to the anomaly?
- What information is available regarding any specifiable SysML model element or specifiable group of SysML model elements? Albeit as document, digitized communication or any other kind of digitized development artifact?

Figure 1-2 shows an overview of the workflow implemented by the tool.

The third part of the thesis shows how the tool and modeling guideline can be applied to different use cases. The primary use case is the MOVE-II mission, presented in Chapter 9. A model of the MOVE-II spacecraft, its ground station and operations environment is built for the purpose, validating the modeling guideline. The model is then transferred into a graph database via the graph schema for SysML developed in Chapter 6. Further, the mission's documentation with over 60000 informational artifacts is accessed by tool and connected to the graph database via the graph schema for informational artifacts developed in Chapter 7. The chapter concludes with a variety of queries to the database that allow the above-mentioned automatic information gathering on the spacecraft's model and informational environment. The impact of following or omitting the various modeling guidelines is discussed in detail. As secondary use cases, the Extendable Modular Power System (an experimental payload developed at the Chair of Astronautics) and the Environmental Control and Life Support System of a hypothetical Lunar Base are presented and discussed in Chapter 10.

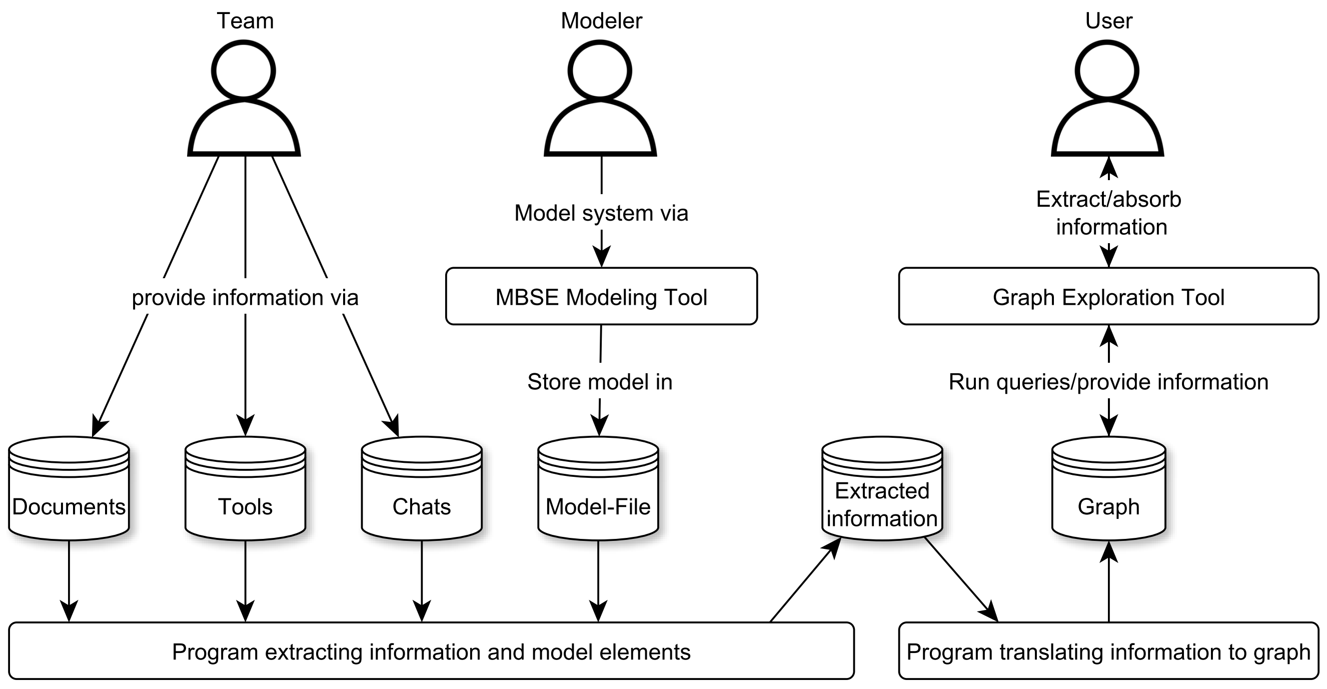


Figure 1-2.: Process overview of the Graph-Enhanced System Analysis

2. State of the Art

Capturing the state of the art in information management systems requires a broad approach. To keep an overview over the various scientific fields that overlap on this topic, the chapter is divided in a theoretical background on the concept of knowledge and information, theories on information and knowledge management, graph theories, classical information management technologies, model-based systems engineering and graph database systems.

2.1. Definition of Key Terms in Knowledge Research

The acquisition of knowledge is the foundation of every kind of scientific research. It is also the subject of research areas such as knowledge management, learning and communication theories. The key terms knowledge, information, data and learning shall be defined in the following, based on their public perception. In the book "Knowledge Management" North and Kumta give general definitions of the terms, that shall be taken as a basis and compared to other definitions in research in the following [27].

2.1.1. Definitions of Data

Various scientific disciplines provide and use different definitions of the term data. The following paragraph provides a comprehensive summary and explains the decision to stick with a certain definition based on the relevant context. The etymological origin of data lies in the Latin datum, which translates to "thing given"[28]. The Concise Encyclopedia of Statistics defines data as the "result of an observation made on a population or on a sample"[29]. They furthermore state that data necessarily contains information, and numbers and descriptions that do not contain any form of information may not be seen as data.

The Dictionary of Computing defines data as "a representation of facts, terms or instructions fit for storing, processing and transmitting" [30].

The definition used in knowledge management as provided by North and Kumta, "Symbols plus syntax become data"[27]stands in stark contrast to the Encyclopedia of Statistic's definition for the term. According to North and Kumta, "information is organized data adding meaning to a message. This information is interpreted differently depending on context, experience and the expectations of people" [27]. A second definition backing North and Kumta is provided in Witt's *Datenschutz kompakt und verständlich*: "Data are contextfree statements consisting of interpreted signs and signals"[31]. North and Kumta explicitly state that data may become information once combined with a certain meaning, while in statistics the definitions are exactly reversed.

Comparing all four definitions, they agree on the following: data is a tangible, recorded object. Furthermore, the definitions of North and Kumta as well as Witt are not challenged by the other two definitions, merely further constrained.

Since definitions should only be as constraining as necessary, the definition of North and Kumta is superior to Ferretti's, while the definition of Witt encloses data consisting of signals instead of mere signs which shows a better adaption to modern technology. Symbols that follow a certain syntax are by definition processable, storable and transferable. The definition provided in the Concise Encyclopedia of Statistics on the other hand is too narrow. Defining data as the result of an observation specifically on a population or a sample would not fit any kind of data generated by creative processes. Taking a technical example, software-code or designs made by an engineer are stored as data, but do not necessarily come from the observation of a sample or population.

Hence, for the rest of this thesis we will take Witt's definition of data as contextfree statements consisting of interpreted signs and signals.

2.1.2. Definitions of Information

Witt defines Information as "data which is interpreted (usually by humans) in relation to a context and which leads (especially via processes) to gaining new insights." [31].

North and Kumta define Information as "organized data adding meaning to a message [i.e., other data]. This information is interpreted differently depending on context, experience and the expectations of people" [27].

The Business Dictionary of Web Finance Inc. defines information as "Data that is (1) accurate and timely, (2) specific and organized for a purpose, (3) presented within a context that gives it meaning and relevance, and (4) can lead to an increase in understanding and decrease in uncertainty"[32].

While the details of the above definitions vary widely and are even disagreeable in parts, they all build on a common core: Data needs a context to become information. North and Kumta point out, that the same information may lead different readers to different conclusions. All three sources agree that appropriating information may lead to new insights. Hence, all three definitions implicitly agree, that information can be appropriated by a person and is expected to lead to further insights. The accuracy required by the Business Dictionary is obviously not a sensible requirement, since it would omit the possibility of false information.

The data that has to be added to a data-set to become information is also called metadata [33].

Summing up the above definitions and trying to provide a universal definition of the term, one may state that *information is data including a context that enables its interpretation*.

2.1.3. Definitions of Knowledge and Learning

All three of the above definitions already show that information is ultimately gathered with a certain goal in mind; its interpretation by humans. According to North and Kumta, Knowledge can be gained, if you take in information, view it in a specific context and relate it with your expectations and own experience.

They define knowledge as "the tacit or explicit understanding of people about relationships among phenomena. It is embodied in routines for the performance of activities, in organizational structures and processes and in embedded beliefs and behavior. Knowledge implies an ability to relate inputs to outputs, to observe regularities in information, to codify, explain and ultimately to predict"[27, p. 36].

Bolisani and Bratianu dedicate a whole chapter of their book *Emergent Knowledge Strategies* to the definition of knowledge, going back to the schools of rationalism and empiricism as impersonated by Plato and Aristotle [34], which argue whether knowledge exists aside from sensory perception of the human or requires a physical form to be linked to [35], [36].

For the scope of this thesis, the specific differences of the various definitions for knowledge laid out by [34] do not play a major role, however, since this thesis is focused on the acquisition and application of knowledge in technical development projects. The underlying consensus, that knowledge in contrast to information exists only in the human mind is sufficient for this thesis.

Thereby knowledge per se cannot occur in written form, as it is always bound to the person knowing. If one were to transfer knowledge from one person to another, their experiences, expectations and relevant information would have to be transferred from one person to another, for example by teaching, practical training or by writing and reading information that builds up a new body of knowledge for the person learning.

Most recently, the field of artificial intelligence to support design and construction is arising, investigating how designs or decisions can be based on projecting information digested by an artificial intelligence [37], [38]. While theoretically possible, Mohammadpour et al. [38] also concede that practical applications are still a far way off.

The transfer of knowledge provides us with a proper definition of the term learning: Learning implies the personal appropriation of information albeit through practical training or recollection of recorded information resulting in greater knowledge.

2.1.4. Summary

The definitions provided above are but a small excerpt of the possible definitions for the terms data, information and knowledge and learning. Some of these definitions are in conflict with each other. As this thesis shall cover specifically the capture, structuring and analysis of information within development projects, the consistent set of terms above suffices for our purposes.

Regarding digitization, it is important to state that digitization may help to organize information and thereby make it easier to acquire knowledge. However, as by the definition of knowledge provided by North and Kumta, knowledge cannot be stored and hence also not digitized. Only information can be digitally recorded. Therefore, knowledge sharing requires knowledge to be transformed into information which can then be taken in by another individual who thereby gains further knowledge.

Taking this paradigm a step further, one may discern between information management and knowledge management systems. A knowledge management system entails the process of sharing personal knowledge and acquisition of new person-bound knowledge. An information management system does not incorporate the process of understanding and taking in new information by a human. In the literature, this distinction is not made as clear, which may be attributed to the different common definitions of knowledge and information (see Sections 2.1.2 and 2.1.3)

2.2. Information Management Technologies

To store and retrieve information, a variety of systems have been developed over the course of the years. According to Langenberg et al., knowledge and information management tools can be divided in four categories [39]:

- Wikis
- Collaboration management systems
- Structured knowledge databases
- Enterprise search engines
- Classical databases

Wikis and collaboration management systems like Confluence or Microsoft SharePoint provide an easy entrypoint, since they are simple to set up and use [40], [41]. However, data has to be entered manually and cross-connecting information mediums is not a focus of these tools. According to Langenberg, this is sometimes eased by structuring tags with the help of elaborate ontologies. Employing ontologies, which Gruber defines by

"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." [42, p. 1] in a running development project requires their constant maintenance as the development progresses. This is due to the fact, that new terms come up in the development which need to be put in perspective and therefore worked into the ontology. Creating and maintaining ontologies however is a resource-intensive process. Once the resources spent on maintaining the ontology overtake the savings created through better linkage of information, it becomes unattractive. Hence, according to Langenberg, it is not practiced in most cases, especially not in projects with challenging schedules [39]. Compared to wikis and collaboration management systems, Enterprise Search Engines try to make all data of a company available, in contrast to the typical approach of only being able to search within one information containing medium at a time. Enterprise search engines are especially attractive due to their low requirements on users. Not requiring additional steps by users lets Enterprise Search Engines mitigate the vicious cycle of project delays as described in Section 1.2.

However, setting up such systems is time-consuming and a cross-referencing between documents concerning parts of the system that belong together does not take place. While the above presented systems have their advantages, none of them focus on easily making information accessible from the direct exchange between developers. However, the direct exchange with each other is the most natural way of communication and contains a large share of the information within a development project as detailed by [11], [12], [26] Compared to written documents, the intent behind design decisions, as well as the weighing of arguments in the discussion leading to a decision are most often afterwards not put into curated documents [10].

2.2.1. Design Rationale Systems

In the 1980s and 1990s another tool for information management was in uprise; design rationale systems. In 1970, Kunz et al. proposed what they called "Issue-Based Information Systems". Yue et al. detail the journey from a tool to assist decision-making in politics, Issue-Based Information Systems found their way to technical projects [43], where in 1988 the term design rationale systems was introduced [44], [45]. In the structure above, design rationale systems fall in the category of structured knowledge databases. According to Lee and Lai, a design rationale is an explanation of why an artifact, or some part of an artifact, is designed the way it is [46].

The basic principle of design rationale systems is ordering the discourse that is an integral part of any collaborative effort into issues upon which the discourse unfolds and laying out arguments pro and contra for these issues as well as the decision that was made based on those arguments [45], [46]. Decisions and arguments are then saved and structured along the issues and thereby made retrievable. In case the decision has to be revised later on, the reasoning can again be understood and a proper assessment of the change to be conducted is possible. However, capturing and especially formalizing design rationales is very resource intensive and therefore suffers from an overall low efficiency [43], [47].

In the 1980s and 1990s Design Rationale Systems were the subject of in-depth research, with a variety of possible solutions to the task being proposed. Proposals went into diverse directions, from the most informal way of recording all meetings where design discussions were made on video [48], to systems where languages for formal logic should be learned and applied to their work by all participating developers [49]. Both design directions encountered the same basic problem: To retrieve the stored information either a lot of effort has to be put in when feeding the system, or when retrieving the information, limiting the applicability of design rationale systems [43], [47].

In 2000, Regli et al. published a critical survey of design rationale systems, discussing why no widespread use of those systems could be identified, although research on Design Rationale Systems had been conducted for two decades. They conclude that the systems were not user-friendly enough and that insufficient digitization of the work environment hinders the automatic collection of design rationales. An example provided in their research are appointment searches enhanced by digital technology. Back in 2000 it was not yet common practice to arrange appointments via digital calendars, whereas today making appointments via email and shared calendars is common practice. [47] In 2006, Tang et al. conducted an extensive survey on the recording and use of design rationales among software architects. The survey found, that most architects highly value design rationales for revisiting a decision and revising their design. 80% of the questioned software architects stated, that they usually do not understand a design without design rationales, if they are not the original designer. Asked about the reasons for not documenting design rationales, 60.5% of the software architects gave "No time/budget" as reason and another 30% no suitable tool [50].

In 2018, Yue et al. compared design rationale systems and knowledge management systems and found that while working on similar topics, design rationale systems are still not widespread [43]. They see the reasons for this in the neglect of human factors when working with such systems, for example the loss of personal leverage, if your knowledge is publicly accessible by your colleagues. They argue that social and psychological factors have to be considered for the methodology in order to get a broader acceptance. The second factor they see is the low benefit for the person entering data in design rationale systems. It takes additional time of the designer who is entering data but saves time for someone else, which makes it unattractive to use. Another factor coming into play are the expectations of workers regarding their workspace. A recent study by the Kelton Research LLC questioned nearly 1500 workers about what factors are important for their work life. Only 18% of workers view their company as transparent, while 87% of workers want their company to be transparent and 80% of workers are interested in learning more about how decisions are made in their company [51].

2.2.2. Knowledge Centered Services

While developed for the maintenance and support of technical systems rather than their development, the knowledge centered services methodology presents an interesting approach to knowledge management. The focus of the methodology lies on the same human factors that were responsible for the lacking distribution of Design Rationale Systems. Compared to Design Rationale Systems, there is a wide range of practitioners for Knowledge Centered Services, such as Hewlett Packard, Dell Technologies, Mindtouch, Autodesk or Zendesk [52]. The methodology emphasizes the need of personal incentives to share knowledge [53]. The Consortium for Service Innovation published a detailed guide on adopting Knowledge Centered Services (see [53]), which calls for a distribution of the transition to Knowledge Centered Services in multiple phases. Especially interesting are the benefits which a successful implementation of Knowledge Centered Services shall bring according to [53]. Amongst others, the time to find information shall be reduced, while the retrieved responses shall show an increase of consistency. For information providers, an elimination of redundant work, i.e., not having to provide the same information multiple times, as well as a faster development of resolutions and a visibility of the impact a contribution makes are listed. For the organization that employs Knowledge Centered Services, a cross-organizational communication and collaboration shall be enhanced, the time for on-boarding new team members shall be reduced, duplications reduced, amongst others [53].

The objectivity of the publication with regard to this list of benefits is questionable. It is published by the same consortium that developed Knowledge Centered Services and no indication as to how these presumable benefits were identified is provided. However – whether achieved by Knowledge Centered Services or not – the list of benefits is a good summary of what can be achieved by good knowledge management. According to [54], the methodology builds on the following principles:

1. "Integrate the reuse, improvement, and (if it doesn't [sic] exist) creation of [documentation] into the problem-solving process.

2. Evolve content based on demand and usage.
3. Develop a [documentation] base of collective experience to date.
4. Reward learning, collaboration, sharing, and improving"[54].

This goes to show that Knowledge Centered Services develops in a perpendicular dimension to the information management systems presented above. These systems concentrate on how information is stored, while the Knowledge Centered Services method focuses on how documentation should be generated and maintained during daily work processes. While Point 3 is the common base with any information management system, the other three points stand out. Point 1 applies to Design Rationale Systems as well, but Point 2 shows a deeper understanding of the challenges faced by any author of documentation: To know what to write down requires anticipating what information users need in the future. This leaves the author with the choice to either provide a great number of details of which a large portion never becomes relevant or to provide too little information, which increases the information retrieval effort of users. The solution suggested by Knowledge Centered Services is to keep the initial documentation simple. Once users have actual demands for further information, the content is evolved according to demand [53]. This shows a disadvantage of Knowledge Centered Services: It requires a willingness to change and a high level of discipline to achieve good results. As presented in Section 1.1, the literature suggests development teams to be rather reluctant to any change processes [21].

It has to be mentioned at this point, that the methodology of Knowledge Centered Services is targeted at technical support teams. In difference to development teams, a support team's work is not project-based. They therefore face no milestones or deadlines which results in a more even distribution of work load. Hence, it is easier to employ Knowledge Centered Services on a support team, as effort that leads to long-term benefits is usually not hampered by short-term deadlines, which does not apply to development projects.

On the other hand, as the literature presented in Chapter 1 ([5]–[8]) shows, technical development teams could still benefit from employing the Knowledge Centered Services methodology to reduce risks and development time, as it addresses key problems mentioned in these publications.

2.2.3. Team Communicators as Knowledge Resource

An atypical source of knowledge currently on the rise are Team Communicators such as Slack, Mattermost, or Microsoft Teams. These tools enable direct conversations between members of a team in the form of chats and store these conversations. The larger the team, the more convenient these tools become, since the problem of finding the right person to retrieve information from and the time required to get in touch with them decreases significantly when posting a question in a place where all concerned members of the team are able to read and respond to the question. Alkadi et al. propose a system to capture design rationale from Team Communicators [55], [56]. Communicators do not enhance linking of knowledge, however. A conversation is linked to a specific channel, although it could concern other topics as well, making the information retrieval from such a source chaotic. Even when the person entering information is aware of this, they still have to decide on a single channel to share the information. Although convenient, Team Communicators also increase the risk of blind duplicates of the same information. Thereby, the chance for the same piece of information being changed once outdated in one place but not the other rises.

Evaluating Team Communicators against the presumed benefits of Knowledge Centered Services, they excel at providing information on demand instead of requiring users to write documentation beforehand. Like in a wiki, information is created collaboratively, i.e., more than one team member read most information requests and any ambiguous or wrong information is usually pointed out quite fast, same as in a good wiki. While accurate in the moment of a request, the consistency of information is not maintained over longer periods. Information that is outdated never gets updated, as it is seen only as chat protocols, not a system holding information over time, such as a wiki entry. This also means that information providers often have a large share of redundant work. A positive aspect of Team Communicators however is their ability to enable cross-organizational communication and collaboration.

2.2.4. Wikis

The best known public wiki is probably Wikipedia (see [57]). Founded 2001 as byproduct of the encyclopedia project nupedia, the website is one of the largest resources of public knowledge online, with currently over 55.6 million articles in nearly 300 languages [58], [59]. The nupedia project relied on voluntary contributions that then should be vetted by experts in the relative field. The process from entering a contribution to seeing it published is described as lengthy and tedious [59], whereas anyone can publish an article on Wikipedia without a previous review. According to [60], only two articles were published in the first six months on nupedia, whereas 20 000 articles in 18 languages were published on Wikipedia within a year of its creation. This goes to show that processes for capturing knowledge need to yield results fast for people to willingly contribute. It may be argued that in a professional setting people are held to rules set out by their superiors and contributing to a wiki could be established as a rule. However, processes in which contribution is willingly provided will with no doubt generate better results.

Before going into the application of Wikis and similar Collaboration Management Systems in professional fields and discussing the handling of false information on wikis, what defines a wiki should be established first.

The Technology and Processes of Wikis

In 2004, Wagner published an elaborate work on what defines a wiki and how it serves the needs of users sharing knowledge and users looking for information. According to Wagner, wikis are a "technology that supports *conversational* knowledge creation and sharing" [61, p. 1].

Wagner further details a list of needs for users requiring information and users providing information [61]:

- Users requiring information:
  - Ad-hoc information: What information is required may be unclear beforehand. Hence, a system needs to be able to answer new questions fast.
  - Finding the information: Finding the relevant information in a larger system requires capable search or query engines
  - Filtering information from noise: complementary to finding information, filtering relevant information from irrelevant information is a basic need of information requiring users. Wagner states that especially in information sharing systems such as online-forums this is a prevalent problem.
  - Quality of the source: Users need to assess how reliable the presented information is. Unreliable information can either stem from false information being inserted in the system or the information being outdated in the meanwhile.
- Users providing information:
  - Dynamically changing information: As stated above, one of the qualities of information is its ability to become outdated. New information may become relevant or the subject of the information may have changed. Wagner further argues that maximizing the number of users involved in the process of updating information is a direct result of dynamically changing information.
  - Distributed knowledge: Wagner argues, that "in most cases, collective knowledge is superior to the knowledge of any individual" [61, p. 13]. Even for situations with a key expert, feedback by other

individuals on ambiguities and the like helps to increase the quality of the provided information.

- Quality assurance: The dynamically changing information inevitably results in information being outdated. Information providers hence need to maintain the information base and be able to find and identify outdated information.
- Publication overhead: Users with a vast knowledge typically incorporate key roles that let them gather this information, which often results in full schedules. To be effective, content creation hence benefits from a minimal overhead.

This list of needs can be taken to compare different knowledge and information management systems.

After defining the different user needs, Wagner also elaborates the characteristics of wikis. A wiki employs a software to enable collaborative authoring of information represented in web-pages. An important feature of wikis is their open editing and authoring policy, i.e., every user may add new articles and edit any article, independent of the original authorship. The goal pursued by employing a wiki is the fast gathering of a vast amount of information. This includes information on demand, which is helped by the theory of conversational knowledge creation. Wikis enable the creation of information on demand by implementing hyperlinks that cross-reference articles in the wiki. The feature of open hyperlinks (or open links), which lead to a page not yet created allows users to identify information required by others. Quality assurance is enabled in the open authoring and editing policy of wikis by employing a version control on all content. I.e., all edits are logged and any previous version of a page is retrievable for any user. Furthermore, the collaborative creation of knowledge is facilitated in wikis by comment features, which allow users to ask questions on information already provided.

Open hyperlinks and comments allow for an organic growth of the information provided in a wiki. Information providing users do not need to anticipate which information they should enter in the system, as comments and open hyperlinks provide them with feedback on additionally sought after information.

Wikis in Professional Research and Development Environments

In 2006, Majchrzak published results of a survey together with Wagner [62]. Goal of the survey was to answer the question whether wikis in corporate settings are sustainable, as well as what drives contributors of corporate wikis to share their knowledge amongst others.

Although the technology was still new and revolutionary at the time – the majority of work sites employing wikis represented in the study did so with success. They measured the sustainability of wikis by investigating the age, number of contributors, number of readers and frequency of page access. Their findings show that most wikis grow over time and the older a wiki is, the higher the activity and amount of readers and contributors [62]. The most frequent reasons reported by their interviewees for contributing to the wiki are an enhanced reputation, making work easier and helping their organization improve its processes. According to Majchrzak et al. this holds especially true when tasks require novel solutions and the articles are seen as credible.

One of the advantages of a professional environment compared to the voluntary environments of public wikis such as Wikipedia was identified by Standing in 2011 [63]: According to Standing, in difference to public wikis, wikis in corporate environments do not face the risk of vandalism, identified by Pfaff and Hasan in 2006 as knowingly falsifying content [64].

In 2010 Grudin and Poole published an extensive study titled "Wikis at work: success factors and challenges for sustainability of enterprise wikis" [65]. They investigated how wikis are used in professional environments on six different companies and tried to synthesize common success factors and challenges. The companies ranged in size from small startups to large pharma and software companies in which they were able to interview multiple teams. Some of their findings are exemplary for the introduction of any new information management system and are to be reconsidered in later sections of this thesis. Grudin and Poole categorize the challenges in expectation management between management, team leads and individual contributors, content organization and -flexibility and introduction challenges in an existing information ecology and corporate culture. Grudin and Poole identified a challenge for information management via wikis in the availability of additional communication paths outside the wiki. Compared to Wikipedia, where conflicts and discussions between contributors could only be resolved on the platform, as it typically presents the only available means of contacting another author, and thereby resulted in new contributions and frequent updates to the Wikipedia, developers in a company have a wide variety of possibilities to interact with each other. Emails, Instant messages, meetings and phone calls are just some possible communication paths available within a company to coordinate or resolve conflicts. This leads to a loss of information on the wiki; A developer might question whether information displayed is correct, but instead of commenting on the wiki page directly may resort to any other communication channel to get in touch with the respective author. This results in the information in the wiki never being updated, as the person looking for information has their need satisfied by the direct correspondence [65]. Arazy and Ofer observed, that the employment of wikis and their open readability by other project members can trigger risk avoidance motives [66]. Combining both observations one may conclude that users may not want to discuss the accuracy of content created by another team member in a format that the whole company can follow. This leads to avoiding the wiki's public features to discuss content accuracy and makes alternative ways such as direct and comparably private contact via email attractive.

While discussing whether content is accurate via alternative communication channels may be working in the short run, this results in another problem on the long run. Discerning outdated from accurate information in a wiki to which anyone can contribute what comes to their mind is difficult and time-consuming. Grudin and Poole found that the majority of successful[1] wiki deployments suffer from a large amount of outdated information, which increases with the age of the wiki. Participants in the study described maintaining the accuracy of the wiki as an unresolved problem, as no automatism can be established to discern accurate from outdated information [65].

Grudin's and Poole's findings support the theory of the vicious cycle of project delays (compare Section 1.2). Updating and creating content on the wiki was seen as a negligible task by managers which participated in the study, especially when deadlines of the project loomed, thus increasing the amount of outdated information on the wiki, the further a project is delayed. From the managers' perspective, reports on the project's progress had a higher priority [65].

This stands in direct contrast to the success factors of wiki usage identified by Grudin and Poole: Successful cases of wiki usage shared intense information sharing needs, no pre-existing work organization and tight deadlines. Therefore, if really embraced, a wiki can be a useful tool to meet a deadline, especially in situations of intense information sharing needs [65].

In summary, wikis are a seen as a useful and successful information management tool. While problems such as content being outdated are unresolved and the initial deployment of stand-alone wikis often lead to conflicts of interest as described by [65], the technology overall achieves what design rationale systems did not; being deployed on a massive scale and maintained as useful information management tools in corporate settings. This may be attributed to the different approach on recording knowledge: While a Design Rationale System asks to record any decision related information independent of whether someone requested it yet, wikis are based on users contributing information they consider useful or for which they see open links and hence know that a need for this information exists.

Comparing wikis to the presumed benefits of Knowledge Centered Services, they help to reduce the time to find information and by employing hyperlinks between the pages of a wiki also reduce the chances of recording information twice and thereby increase the consistency compared to storing the same information in static

[1]Successful being defined by the number of contributors and editors and the amount of content available in the wiki

documents. Information providers benefit greatly from the concept of open links which helps them identify informational needs that are still open. As the wiki can be seen by all people who are considered to need access to the information therein, a cross-organizational communication and collaboration is facilitated, however not on the same level as with Team Communicators, which allow for faster communication.

Modern collaboration management solutions such as Redmine, Atlassian's Jira or GitLab, often incorporate a wiki and natively support linking information in the wiki with other parts of the collaboration management system.

2.2.5. Collaboration Management Systems

A variety of collaboration management systems exists, covering similar features. Whether Redmine [67], GitLab [68] or Atlassian's Jira [69], they all feature git-based revision control for code and text files, a ticketing system to distribute work, features for project planning and a wiki [70]–[72].

Used with appropriate processes, for example state-of-the-art software engineering methods such as Scrum or Kanban (compare [73], [74]), collaboration management systems allow recording not only arbitrary added information in their respective wikis, but additionally capture the complete workflow of a project. The status of work packages is tracked in issues, while any new features to the code-base of a project are reviewed via so-called Pull or Merge Requests, by which developers review each other's code style and thereby insights into the design rationales are recorded. Most of these systems allow directly linking any artifacts within the collaboration management system, including the creation of backward-traceable links[2]. A new commit to the software can directly be attributed to an issue or a specific snippet of code can be referenced in a wiki entry. Allowing for the creation of these links was taken over from the mechanics of wikis, which also allow for backward-traceable links. The shortcomings of wikis from a management perspective as described in [65] are thereby overcome. As the whole work process can be organized in a collaboration management system, they come close to fulfilling the original goals of design rational systems. Although information is not recorded for the pure sake of being able to retrace decisions but rather to organize the current work load of the development team, the recording of work processes and the regular reviews enable the reconstruction of past decision processes.

While Collaboration Management Systems solve some challenges their predecessors in information management left open, others still prevail:

- Collaboration management systems do not resolve where to record specific information, a problem already described by Grudin [65]. The large amount of features available in a collaboration management system rather aggravates the problem.
- No new solution on identifying outdated information is presented by collaboration management systems. Therefore, collaboration management systems aggravates the problem of finding out whether information is still accurate, as more information is recorded that has to be sighted.
- Collaboration management systems do not close alternative communication paths such as email, meetings and in-person communication or even other data storage systems such as network drives. While a large amount of information may now be found in the collaboration management system and is even searchable by the system's integrated search engine, it is impossible to tell whether further information to a topic was also shared along another communication path or lies in a network drive that is not part of the collaboration management system.

In reference to the presumed benefits of Knowledge Centered Services, Collaboration Management Systems also reduce the time to find information compared to a purely document based approach. The collaborative

[2]I.e., If page A references page B, page B automatically also references page A, allowing for traceability in both directions.

approach also acts as a guard for the consistency of information.

For information providers, redundant work is reduced by being able to link information across the whole Collaboration Management System. However, since – similar to Team Communicators – Collaboration Management Systems are mainly there to structure the current work process, identifying and dealing with outdated information is not a focus point of Collaboration Management Systems. Therefore, no mechanism is in place to deal with outdated information and queries to the system may return inconsistent answers. The chance of inconsistency increases the amount of duplicate work for information providing users again, as confirming information is still up-to-date becomes a regular necessity.

The last point in the above listing, concerning alternative communication paths is addressed by Enterprise Search Engines.

2.2.6. Enterprise Search Engines

Together with the rise of the internet, intranets in companies gained importance and track in the early 2000s. One of the areas of research in setting up a working intranet was enterprise-wide search. An intranet is defined as "a network operating like the World Wide Web but having access restricted to a limited group of authorized users (such as employees of a company)"[75]. Therefore, intranets are not part of the public internet, but the network infrastructure only available to a group of users working together. Fagin et al. investigate the differences in structure between the public internet and company-specific intranets [76]. They found the structures and mechanisms that build content on the internet to be vastly different from an intranet-landscape: While the internet often contains multiple pages holding the same or similar information, created by multiple people, in an intranet specific information is ideally represented exactly once.

Thereby there is a single source of truth for this information, allowing for proper maintenance of the information. For a search engine the prerequisites of the available information are therefore vastly different, as it has to find the one right page instead of a wide variety of pages containing similar information. Also, algorithms such as page-rank on which Google's search builds, which rely on how many pages reference a page to judge its popularity and thereby importance, do not show satisfying results [77] in intranets. Another problem for Enterprise Search Engines is the heterogeneity of data sources in an intranet. To be effective, the Enterprise Search Engine has to query all data sources in which relevant information could be stored, including any file format, database format, server application programming interface etc. Therefore, an Enterprise Search Engine requires sizeable efforts to set up.

Another question is which representation of the search results is the easiest to navigate. Jones et al. propose to extend the typical table-based search result representation as known from any web-search tool by a domain specific view. The domain specific view may entail a CAD-Model of the system under development, a world map showing the different sites of the company or any other visualization the users are familiar with [78], [79]. This however requires a model of the system under development or the object of interest for a group that can be used to underlie the search.

Compared to Design Rationale Systems, Wikis, Team Communicators and Collaboration Management Systems, i.e., the before presented information management systems, Enterprise Search Engines resolve multiple problems:

- Where information shall be recorded becomes less relevant, as the engine is able to find the information again, independent of where it is stored.
- The problem of alternative communication paths gets mitigated to some degree. If information is only shared in private direct communication, it is not retrievable by the Enterprise Search Engine, as basic privacy protection has to be observed. In any other case, an Enterprise Search Engine is able to retrieve

information independent of the communication path employed.

- It becomes easier to decide whether information is still accurate. By comparing the date last updated between the retrieved artifacts the latest revision of a certain piece of information can be identified, no matter whether it was sent via email, stored on a wiki page, or is part of a document.
- The process of introducing an Enterprise Search Engine does not require any compliance by the staff of the enterprise and hence is robust against reluctance to change and lack of discipline. Reluctance to change and to comply with new processes is a common pitfall for the introduction of an information management system [21], [47], [53], [62], [65].

While it requires sizeable effort to set up an Enterprise Search Engine, it may well be worth it, considering the amount of time spent by engineers on retrieving information (see Chapter 1). Various Enterprise Search Engines are available on the market, e.g., Apache Solr or searchIT [80], [81].

Enterprise Search Engines may sound like an ideal solution, but there are still hurdles to overcome as described above. While the proposal of Jones to link information to the taxonomy of the system under development is promising, such a taxonomy has to exist in the first place to realize the proposal. Also, an outdated taxonomy is of little use, hence it has to grow during the development, keeping pace with new terms arising. Jones solves this by employing the Bill of Materials from a Computer Aided Design Model of the system under development. A Computer Aided Design model may be a good start for the mechanical engineering taxonomy of a system, but fails to cover electronics, software, requirements, sales, testing etc. on an appropriate level [78], [79].

2.2.7. Model Based Systems Engineering

As elaborated in Section 1.1, MBSE faces its own challenges. Compared to other information management technologies, such as collaboration management systems or enterprise search engines, Model Based Systems Engineering is focussed on a small portion of the overall amount of information available in a development project, i.e., usually only the technical system under development. Modern MBSE Solutions such as Cameo and Cameo collaborator allow engineers to design the system from a holistic point of view, incorporating stakeholders, mission objectives, requirements, the structure of software, mechanics and electronics and tests. However, while in theory MBSE should be applied across the life cycle, such an application is still rare. According to recent studies, the majority of MBSE practicing organizations only apply MBSE in pilot projects and usually only for requirements management and early drafts of the system architecture [21], [82].

A recent project that highlights the current abilities and challenges of MBSE is the National Aeronautics and Space Administration's Europa Clipper mission [83]. The deep space mission targets the Jupiter moon Europa [84]. From 2011 on, MBSE was applied to the mission. According to Bayer et al. the mission's mass budget was maintained in a MBSE environment since 2011, joined by the power equipment list in 2012, a power and energy simulation in 2013 and a framework for tracing specific requirements regarding science measurements [85]. While tracking the system's budgets and a specific subset of requirements under a model based approach works well according to Bayer et al., modelling the architecture and all other requirements was discontinued in 2019. After the preliminary design review, the model encompassing these elements was retired, as after five years the model was still not complete. Multiple reasons are provided by Bayer et al., the most prominent being the mismatch between human resources and scope of the task, the inefficiency of state-of-the-art modelling tools such as MagicDraw/Cameo and the problematic decision to work with partially incomplete in-house designed modelling software (CAESAR and openMBEE), which had to be developed at the same time the mission should be modelled with it [85]. The modelling approach employed by the mission is detailed by Dubos et al. [83]. They describe how the model should be built on four columns, called Concerns, Parent Concepts, Concepts and "Child" Concepts as illustrated in Figure 2-1. The modeling approach detailed in Figure 2-1

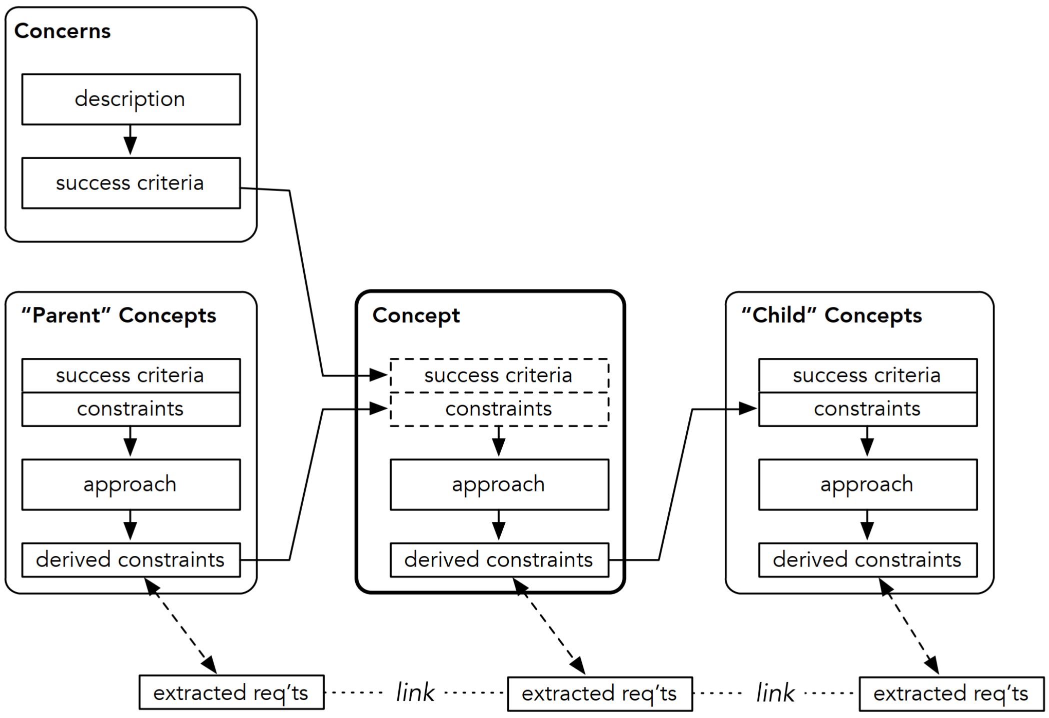


Figure 2-1.: Modelling strategy of the Europa Clipper project[83]

was cited as one of the reasons the scope of the MBSE activities in the mission was drastically reduced: The required effort for the approach having been inconsistent with the available human resources.

One of the few public success stories of MBSE on a large and complex project is the application in the design of large astronomic telescopes. Combining the tight requirements imposed by a telescope's optics with environmental conditions, mechanical assemblies electronics and software is very demanding and according to Karban et al. works very well employing MBSE. Going back to 2008, Karban et al. published a series of articles on how MBSE is applied to verify the design of specific subassemblies of first the Very Large Telescope in Chile and later the European Extremely Large Telescope (now the Extremely Large Telescope) [86]–[90].

The models they build focus on requirements verification, built on Systems Modeling Language (SysML) Models designed in MagicDraw and its successor Cameo. From an early stage on, the group focused on state analysis as a verification technique and automatically generating code for the telescopes' control units [87], [90]. Later publications focus on model reuse, fault management modelling and executable models [88], [91], [92]. The procedure for fault management modelling as presented by Gibson et al. in [92] is to model every expected fault and simulate the system's response to it within the model. They apply the presented procedure to the fault processing unit of the Soil Moisture Active Passive (SMAP) spacecraft, which is in operation since 2015.

Apart from spacecraft and telescope modelling, another example of a successful application of MBSE are airplanes of the aerospace company Boeing. According to Malone et al., Boeing is successfully applying MBSE with thousands of contributors over their complete airplane designs, successfully decreasing necessary resources for rework in the integration phase of the airplane while distributing the development to over 30 sites internationally. However, they see major challenges regarding the tool landscape and academic curriculae: They elaborate how missing standards for data exchange between modelling tools hinder their work as well as missing education in adapting modelling tools to the need at hand in engineering courses. They further stress the importance of integrating all information artifacts in the engineering environment, such as spreadsheets, documents and databases rather than relying on a single graphical modelling language. [93]

Further, works by Malone et al. cover the engineering of Boeing's own MBSE environment. Similar to the works of Brandstätter et al. [94]–[96], they view the complete engineering environment, including the various models and other informational artifacts as an environment that can be modelled by systems engineering methods. Malone et al. use this model to define requirements for the tools used in their MBSE approach.

Thereby instead of abolishing document based resources, their goal is to understand the processes leading to these resources and to create a specific solution for each engineering team or subcontractor that integrates the resources in use rather than replacing them, thereby being the first Model-Based Systems Engineering approach that comes close to being a holistic information management system.

2.2.8. Information Management with Knowledge Graphs and Semantic Web Technology

Around the time of the publication on knowledge management systems by Langenberg et al. in 2013, a new type of database management systems, so-called graph databases had its first releases. According to Miller, applications range from social networks over semantic web to chemistry, biology and recommendation engines[97]. More recent publications show examples of graph databases being used to connect scientific papers as well as technical reports and trace requirements [98]–[100].

According to [101], the term knowledge graph was minted in the 1980s by C. Hoede and F.N. Stokman. The fundamental idea behind knowledge graphs is to link semantic information between various artifacts, such as texts or websites. [102] compares a variety of definitions of knowledge graphs and comes to the conclusion that "A knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new [insights]"[3] [102, p.3].

[103] demonstrate how a domain-specific knowledge graph can be generated via natural language processing and stored and queried in a graph database. Knowledge graphs gained momentum as Google introduced them as basis for their search engine in 2012. [104]. Then Senior Vice President for Search at Google Inc., Singhal points out new search features of the Google web search engine in [104], that build on connecting information and understanding informational entities instead of treating the search terms only as strings of characters. This requires the storage of such linked information about as many entities as possible.

In 2016, Paulheim published a comparison between different approaches to refining knowledge graphs. He defines the term knowledge graph by four requirements that have to be met for a set of structured data to be called a knowledge graph [105]. "A knowledge graph

1. mainly describes real world entities and their interrelations, organized in a graph.
2. Defines possible classes and relations of entities in a schema.
3. Allows for potentially interrelating arbitrary entities with each other.
4. Covers various topical domains."[105, p. 2f].

While the definition provided by Paulheim partially contradicts that of Ehrlinger (see above), one has to take the different viewpoints into account. Paulheim was mainly focused on defining knowledge graphs to identify datasets on the web that can be called a knowledge graph, while Ehrlinger takes in the application of reasoning to derive new information.

General purpose knowledge graphs, as investigated by Paulheim suffer from trying to cover a body of information with no boundaries. The application of such general purpose knowledge graphs is hence not necessary

[3]The original term "knowledge" at the end of this sentence was replaced by insights as it is in contrast to the definition of knowledge used in this thesis.

to structure information on a technical development project. A smaller and much more focused body of information is sufficient to describe a technical development project. Therefore, the ontologies defined in these knowledge graphs can also be considered too vast for the purposes and scope of this thesis. However, the technology behind these graphs is to be considered and thoroughly investigated. Compared to Enterprise Search Engines, which yield information from the semi-structured and diverse landscape of a company's infrastructure as simple hits on pages or documents, knowledge graphs allow calling for specific information. They retrieve exactly the information required, as they allow the user to post specific queries regarding an exactly defined informational entity. This fits well with one of the observations of Fagin et al. that search in an intranet should often only lead to one correct answer instead of a variety of possible results [76].

The technology behind knowledge graphs is well established. Tim Berners-Lee proposed the semantic web in 1998 (see [106], [107]) as part of the internet, to contain machine-understandable information, linked to rules for the reasoning of this information. The concept of the semantic web is thereby very close to the knowledge graph concept proposed by Singhal. The backbone of the semantic web as proposed by Berners-Lee builds on the Resource Description Framework (RDF), a standard of the World Wide Web Foundation [108]. The World Wide Web Foundation provides the following definition for the Resource Description Framework [109]:

"RDF is a standard model for data interchange on the Web. RDF has features that facilitate data merging even if the underlying schemas differ, and it specifically supports the evolution of schemas over time without requiring all the data consumers to be changed."

The RDF is outstanding for its universal ability to link information. Any website can host information in RDF and anyone with access to the website can reference this information with their own [110]. This reference to any information is made directly to the source, no copies of the resource are created. These references are created as so-called triples. Triples make up the basic layer of RDF. They consist of three Uniform Resource Identifier (URI), which work as subject, predicate and object[111]. The URI allows for exact identification of an entity on the world-wide-web. While the URIs of subject and object are arbitrary and no special taxonomy has to be followed to define them, the predicate part of an RDF triple is usually taken from a well-defined vocabulary that specifies how different entities can be linked. Triples can also be used to characterize an URI, for example to define that an URI represents a restaurant or public site. In these cases, the objects are also taken from a vocabulary. One of the most basic vocabularies is the RDF Schema published by the World Wide Web Foundation [109]. According to Chah and Paulheim, RDF is also used for Google's Knowledge Graph as well as most other sizable publicly accessible knowledge graph [105], [112].

Knowledge Graphs in Technical Development Projects

Elaborate vocabularies for Knowledge Graphs, which cover a specific technical area of interest are called ontologies [113]. The European Space Agency currently tries to establish an ontology for space systems. The project with the acronym OsMoSE was kicked off and is still ongoing. While it shall span the whole life cycle of a space mission, the project's scope is limited to the interaction between stakeholders, i.e., it only covers the spacecraft's interfaces to a level of detail necessary to model stakeholder interactions. Long term, the ontology shall cover any discipline represented in the European Cooperation of Space Standardization. However, these goals are far off. After over two years, not even an initial draft of the ontology is mentioned, which may be attributed to how challenging a project with such a wide scope is [114]–[116].

A similar effort was started by the NASA in the form of the Caesar software [117], in which MBSE and ontological reasoning are combined to a new tool-set, which shall enable a better exchange of information across tools and disciplines. Caesar was employed on the Europa Clipper mission, which shall conduct measurements in the vicinity of the Jupiter moon Europa[84]. Initially, the scope of MBSE encompassed modelling the architecture and requirements [83]. But after the Preliminary Design Review, it was decided to shift from

MBSE back to a traditional documents based approach[85]. While Power-, Thermal-, and Mass Budget were kept in openCaesar, architecture and requirements are since being designed in the traditional (i.e., not MBSE) way again. The reason fell as five years into the mission's development still only a partial implementation of modeling tool and model existed which would not do. In a dedicated lessons-learned paper, Bayer et al. reflect on the reasons for this failed application of MBSE [85]. They come to the conclusion, that workforce training, creation of tutorials, writing the software framework and applying these measures on a running project was simply too much to maintain a working and consistent system model. Another problem identified by [85] is the inadequacy of existing tools for qualitative model analysis. The examples they provide for this statement are rather shallow, though. Checking whether any component is linked to a function that requires it, or whether a requirement also refers to a component contained by a subsystem that the requirement's higher level requirement refers to.

Knowledge Graphs in Journalism

One of the reasons attributed to the failure of applying MBSE on Europa Clipper is the human reluctance to change to new methods and processes [85]. While a certain amount of cooperation can be expected from members of a development team, a tool that does not require any cooperation is naturally robust against non-compliance. The problem of managing information and drawing conclusions out of large quantities of complex and diverse data is not exclusive to the engineering domain. A field of profession that cannot expect any compliance from the creators of the data under analysis is investigative journalism.

As systems engineers are tasked with "bringing order from chaos" [118, p.4], entities seeking international tax optimization and evasion pursue the exact opposite goal. The more chaotic a tax-evasion system is set up, the harder it is to pervade its true intents and perceive the beneficiaries behind the scheme. The publication of the so-called Panama Papers in 2016, which contained leaked documents on tax evasion of a panama-based law firm lead to investigations in over 80 countries, multiple arrests and the recovery of over $1.36b in taxes [119]. The discovery is based on a leakage of 2.6 TB of unstructured data, comprised of 11.5 million files. The data consisted of emails, database dumps, Portable Document Format (PDF) files and scanned texts and was generally unstructured [120].

The central tool used to link all of this information and make it accessible to journalists and the public is a graph database [120]. The documents are used to create a network of clients and companies, based on names, addresses and correspondence and documents which link those entities. To retrieve such a network from a stack of partially structured data, it is necessary to extract names of clients and companies and their respective addresses from the data. Hunger and Lyon lay out the approach used to identify named entities and deduct relations between entities in [121]. They also review the model by which information is included in the graph, with the majority of their suggestions for improvement being centered on the extraction of named entities. Being able to query the network allowed and allows journalists from around the globe as well as the public to understand and reveal the schemes of obscuring ownership and business relations, as connections via intermediaries, or businesses that share the same address or owner can be traced through the network.

The example shows how complex and only partially structured information can be made accessible to people with a very varying level of technological understanding [120] and without any cooperation by the creators of the data.

Similarities between Investigative Journalism and Information Management in Development Projects

While it is usually not the intention of any spacecraft designer to artificially obscure information, we find some parallels between journalists trying to review the information of data leaks such as the Panama Papers and spacecraft engineers:

- Both are looking for information in vast repositories, without knowing exactly how much information they will find for the respective topic.
- The data sources come in a variety of formats with varying ease of access and information on the same topic can often be found distributed over various sources.
- The system under scrutiny is complex with a high amount of dependencies and interrelations.
- The users have a varying degree of affinity to database-based information systems.

Another similarity comes up between journalists and spacecraft operators; there is a high likelihood that neither can ask the original source/ author of a document for further information. For the journalists, this is based on the unwillingness to cooperate from entities under scrutiny, while for spacecraft operators the once responsible developer may simply be out of reach due to the project cycles being up to a decade for development and up to one and a half decades of operations, or the information being restricted due to intellectual property considerations.

On the other hand, there are some rather obvious differences; For one, spacecraft engineers tend to know exactly what they are looking for and talking about, i.e., employ a search in their informational system with a very specific goal in mind. Journalists on the other hand cannot know beforehand whose information they may find and what information to look for while they are sifting through a data leak such as the Panama Papers. Also, the scope of what one is looking for is limited in a spacecraft development's documentation, which cannot be said for journalists. Journalists may be satisfied with finding a certain amount of information in their system, and while it is certainly on their agenda to provide a complete picture, lacking pieces of information do not reduce the value of the information already gathered. For engineers, a missing memo which updates the expected ranges of a certain piece of telemetry may be a major problem on the other hand. I.e., while sorting out false positives on a search is considered cumbersome but acceptable, false negatives impose an actual problem. While the datasets may show some similarities, they each also have individual features not represented in the other one; For example a spacecraft design usually goes through several revisions until it is being built, follows an overall project schedule and already during testing includes telemetry databases which play a large role. None of this may be said for a leak of tax evasion documents.

Another topic is data protection and data governance; While journalists have a large amount of freedom once they obtained a source, the same cannot be said of the engineering world, which is governed by international standards such as the General Data Protection Regulation of the European Union and is usually closely monitored by employee representative bodies and unions, which may curb what information is to be processed in which way even further. Still both journalists and engineers are very interested in protecting their datasets from unauthorized access.

Open Services for Lifecycle Collaboration

An initiative to address the topics of outdated information, data governance and data protection is the Open Services for Lifecycle Collaboration. Founded in 2008, the idea behind the collaboration is to increase the interaction capabilities of information management systems [122]. Open Services for Lifecycle Collaboration works on the principle sharing access to information instead of sharing the information itself. Thereby no copies are made of the information itself and hence no copies of the information need to be updated, when the original information is updated [123].

2.2.9. Graph Databases as Central Information Repositories

In comparison to relational databases, graph databases are designed to enable efficient creation and querying of relations between data points [124]. Hence, they are specifically useful if it is of interest how entities or information is related. Applications range from shopping recommendation systems [125] over lessons learned graphs and correlation of Earth Observation Data at NASA [99], [126], as well as social networks such as Twitter and Facebook, to the search algorithm behind Google's search engine [112]. Understanding how a graph database leverages the relations between data points requires basic understanding of the concept of graphs. According to West's *Introduction to Graph Theory* [127, p. 2],

A graph G is a triplet consisting of a vertex set V(g) an edge set E(G) and a relation that associates with each edge two vertices (not necessarily distinct) called its endpoints..

In their book *Graph Databases* Robinson et al. introduce the modern synonyms for vertices and edges, naming them nodes and relations respectively[128]. Graph databases employ the concept of a graph to store relations and nodes separately from each other, increasing the efficiency of querying relations compared to relational databases [128].

The Labelled Property Graph Model

The following subsection builds on work published by the author in [22].

Like any other graph database model, the main components of the Labelled Property Graph Model are nodes and edges. Compared to other graph data models, the Labelled Property Graph Model allows to store key-value pairs as properties directly on nodes and edges, whereas other graph data models such as the Resource description Framework require a separate node for every property that shall be stored and do not allow for properties of edges ([129], [130]). Edges in the Labelled Property Graph Model carry a single type, describing the relation formed between the nodes. If multiple types shall be assigned to a connection between two nodes, the nodes are connected with multiple edges. The edges are directed, i.e., every edge has a dedicated source node and a dedicated target node. An edge also connects exactly two nodes. If more than two nodes shall be connected, a so-called hypernode can be employed, to which all nodes are connected that share the relation ([131]). Apart from key-value based properties, nodes can carry multiple labels, which define the type of node. To transfer information from another data source into the graph database, a graph schema has to be defined, which specifies which information is to be stored as nodes and defines the labels and properties for each type of information as well as how relations are to be drawn from the information and what types and properties are to be assigned to the relations.

Application of Graph Databases as Central Information Repositories in Technical Development Projects

Similar to the concept of enterprise search engines, graph databases can be used to access the informational environment of a technical development project [132]. In a series of publications, Bajaj et al. [132]–[134] propose to link the various information management systems used in a project, such as collaboration management systems, requirement management systems and Network Access Storage systems via a graph database.

They also propose the transfer of SysML Models to the graph database and show how elements of the SysML Model can be linked to elements of other information management systems, for example requirements in SysML to their equivalent in IBM Doors [134]. However, while showing how a wide variety of information can be inserted into the graph and how links can manually be created within the information from one data source to another, the queries presented by Bajaj et al. do not make use of a graph database's ability to efficiently query the relations between data points. All except one query presented in [134] could be resolved just as well with a conventional relational database.

One other example of transferring a SysML Model into a graph database is published so far. Petnga [135] proposes the transfer of a SysML Model from modelling tools to Graph Databases in order to analyze the completeness, consistency and correctness of requirements. While – in contrast to Bajaj et al. – publishing the graph schema employed for the analysis in [135], the analyses conducted with the graph remain on the same basic level as those presented by Bajaj et al. The only point by which Petnga makes use of the abilities of a graph database is to employ so-called graph algorithms, by which the nodes' position in the database's network of nodes can be characterized. However, apart from showing that these algorithms can be applied to a network of nodes and edges extracted from a SysML Model, no valuable information is provided on the topic. While Petnga presents results of the Betweenness Centrality algorithm, which is a measure on how important a node is for a network (cf. [130]), no discussion is provided on how the results can be interpreted, or which relations and parameters were used to run the algorithm.

2.2.10. Summary

Gathering information is a critical and time-consuming part of the daily work in an engineering project and a lack of information management can have dire consequences for any development project [5]–[8], [10], [11]. The interconnected nature of the information in a development project along with the multitude of possibilities of how to share and where to store information deny hierarchic storage systems. Information management tools and strategies such as design rationale systems, wikis, collaboration management systems, knowledge databases, Knowledge Centered Services, team communicators, Enterprise Search Engines and MBSE are valuable assets for managing the information of complex projects.

However – except for Enterprise Search Engines, team communicators and certain types of knowledge graphs – all of these solutions require a high level of commitment of the development team to enter and maintain the information in order to be effective [10], [62], [66], [136].

The reasons for this are that on the one hand, adding information to these systems manually and maintaining it requires the investment of additional time by the users. On the other hand, connecting information in these systems is done mostly manually, since ontologies, that would help to connect information, are not in widespread use.

Currently, Enterprise Search Engines pose the only approach which provides a holistic insight into any information available on a development project. With any information available on the whole project, delivering the right information becomes a challenge that greatly benefits from a well maintained vocabulary on the system [77]–[79].

Successful applications of MBSE as an information management strategy usually require an extra engineering team to adapt existing solutions to the specific needs of the development team. This is confirmed by the successes of such applications as well as the failures attributed to insufficient resources for a successful adaption [85], [89], [92], [93], [137], [138]. This can be attributed to the lack of interoperability of model editors as well as their lacking ability to handle complex queries on the model.

An approach outside the engineering field to information management was presented in Section 2.2.8. Without any compliance of the entities who generated the data in the first place, a multitude of interesting insights was produced by connecting information across various sources. Also – apart from Enterprise Search Engines – all systems described here lack a methodical approach on how to identify outdated information.

2.3. Knowledge Intensive Tasks in a Spacecraft Mission's Lifecycle

Like any modern development project, spacecraft missions employ some of the above described information management systems. Unmanned spacecraft cannot be accessed physically anymore, once they are in operation. This makes proper documentation of the system vital. The organization of spacecraft development in so-called phases and the long development timelines underscore this need, as parts of the personnel change from phase to phase, and it cannot be expected for engineering personnel to still be available to answer questions once the spacecraft is in orbit [137].

2.3.1. A Spacecraft Mission's Lifecycle

The following descriptions are based on the classical spacecraft mission lifecycle as defined by ESA and NASA [139], [140]. While not every space mission may follow this approach, the basic tasks necessary to bring such a system to success stay the same and the phases and review-based process employed by the two agencies is well-tried and documented. Note that this section shall not serve as a general or by any means exhaustive description of activities to be performed in a space mission's lifecycle. It concentrates on interdisciplinary knowledge intensive tasks in each phase of the development to enable a comparison between technical solutions for information management and the respective user needs in a spacecraft's lifecycle. Space missions are organized in so-called phases. A phase is a span of time during the development which is characterized by specific tasks and a specific portion of progress in the project and marked by its preceding and concluding reviews. The phases are described in detail in ECSS-E-ST-10C and the NASA Systems Engineering Handbook [139], [140].

At the end of each phase and at strategic points within, reviews are conducted. A set of documents specific for each review defines which information has to be provided. The documents are then provided to experts otherwise not involved in the mission. Their task is to examine the provided information and investigate whether the provided solution is adequate for the defined mission goals. Depending on the review, the information to be examined includes the identification of stakeholders and definition of the mission concept, definition of measures of effectiveness, cost and schedule plans, an architecture definition, various documents capturing the state of requirements to detailed definitions of design solutions and interfaces, plans and results for verification and validation or plans and procedures for the operations of the spacecraft [141].

Table 2-1 provides an overview of the mission phases and respective reviews as defined in [140].

Table 2-1.: Overview of the Phases in the NASA mission life cycle as defined in [140]

		Formulation		Implementation		Operations	
Phase	Pre–Phase A	Phase A	Phase B	Phase C	Phase D	Phase E	Phase F
	Concept Studies	Concept & Technology Development	Preliminary Design & Technology Completion	Final Design & Fabrication	System Assembly Integration & Test, Launch & Checkout	Operations & Sustainment	Closeout
Phase characterization	Small team of scientists and engineers conducting feasibility studies on multitude of mission scenarios, purely theoretical work, no experiments	Develop final mission concept and systeml–level requirements based on most promising results of Pre–Phase A	Detailed definition of interfaces, subsystem requirements and overall architecture of all mission segments (spacecraft and ground segment)	Final designs and fabrication of all hard– and software	Assembly and integration of all hard– and software, Verification and Validation of Requirements and Mission Scenarios, Launch and preparation for operations	Operating the spacecraft according to the mission's concept and goals, implementation of the mission operations plan	Safe disposal of the spacecraft, final analysis of returned data and samples

2.3.2. Pre-Phase A, Concept Studies

During Pre-Phase A, a multitude of ideas for potential missions is evaluated. This usually includes work in Concurrent Design Sessions, in which specialists for all parts of the mission and the spacecraft are represented and jointly work out a concept for a potential mission. Due to the interdisciplinary character of the work to be conducted in this phase and the small team size, interdisciplinary knowledge intensive tasks are performed by nearly every member of the team. Berquand et al. published the results of interviews conducted with 47 members of the European Space Agency's Concurrent Design Facility [8]. The majority of interviewees quantified the share of their work time spent looking for information with 25% to 50%, and nearly a third of the interviewees identified discussions with their colleagues as the most useful source of information. As a Pre-Phase A study targets to identify the overall feasibility of a space mission, neither of these findings is surprising. Every member of the team has either a budget such as the thermal, volume, mass or power budget to account for or the design of a specific piece of equipment. Hence, the information exchange between the participants during the session is a crucial element, as parameters for the space mission need to be defined which are highly dependent with each other. Therefore, while the amount of time allocated to finding information is astounding, the findings cannot be extrapolated to other mission phases.

Tasks standing out as interdisciplinary and knowledge intensive are the definition of mission timelines and judging the feasibility of instruments and performance requirements. The small team size and the overall low amount of information to be maintained are ideal for the implementation of Model-Based Systems Engineering in Pre-Phase A, while the model is then often not maintained in later phases [115], [116]. Compared to later phases, the Mission Concept Review has its focus set on the alignment of the mission concept with strategic goals of the respective entity as well as risks, necessary resources and the technology readiness level of the involved technology. As the technical details are still few and likely to change in Phase A and B, only key mission needs are reviewed in the Mission Concept Review in the form of a preliminary technical requirements specification [139].

2.3.3. Phase A, B and C, from Concept and Technology Development over Preliminary Design and Technology Completion to Final Design and Fabrication

During these phases, a mission concept from a Pre-Phase A study is developed on to a level of detail which allows to define all involved systems down to their respective architecture and interface definitions. Therefore, all requirements have to be defined and allocated to respective parts of the system, as well as verification methods defined by which the implementation of the requirement can be verified later on. The overall system architecture is developed and design and fabrication work is distributed to internal sources as well as contractors. Over these three phases, the personnel involved in the mission increases significantly. While the share of interdisciplinary work is smaller than during Pre-Phase A, the amount of information to be managed increases by orders of magnitude. [139], [140]

As every subsystem is designed by respective specialists, interdisciplinary knowledge intensive tasks concentrate on complex functions that require the collaboration of multiple subsystems, interfaces, budgets and the overall architecture [139]. Especially analyses on the architecture for redundancy, risk, complexity and failure detection isolation and recovery as well as on how complex requirements are implemented by the system can be characterized as knowledge intensive interdisciplinary tasks. These tasks also characterize the respective reviews along with planning for the next phases such as plans for verification and assembly integration and testing, documentation of all systems and first drafts of the spacecraft's user manual. One of the required deliverables to progress from Phase C to Phase D is the documentation of conducted verifications, which span all requirements that can be verified without integrating the subsystems or instruments [139]. These tests are mostly conducted by the same subteam or entity that built the respective subsystem or instrument. As it is difficult to draw lines between development and testing in this phase, the requirements for documenting anomalies vary.

2.3.4. Phase D – System Assembly Integration and Test, Launch and Checkout

Phase D starts by assembling and integrating the hard- and software fabricated in Phase C in incremental steps, after each of which verification tests are conducted on the spacecraft. With an increasing integration of the spacecraft, system-level tests of increasing complexity can be performed. The integration is only finished and the spacecraft ready to progress to launch, once all requirements are verified and validation tests for the overall functionality of the system are conducted.

Compared to the interdisciplinary, knowledge intensive tasks in the former phases, the integration of the spacecraft and resolution of anomalies present new challenges with regard to information management systems. During the phases Pre-Phase A, A, B and C the amount of personnel increased from phase to phase and information generated in previous work was to be distributed and further specified. As in Phase D all parts of the system that were distributed to multiple subteams and contractors convene to form the spacecraft, the information generated during the previous four phases has to be convened as well. At the same time, the amount of personnel actively involved in the mission decreases [142]. Therefore, the amount of information available and relevant to each person involved is by far higher than in the previous phases, while information that is known to the staff of former phases but not explicitly recorded is not available anymore. This leads to the transfer of knowledge becoming a crucial task in Phase D and aggravates the complexity of handling anomalies during the integration.

Any anomalies detected during the integration of the spacecraft are recorded and so-called Anomaly Resolution Boards formed. These boards are tasked with identifying the cause of the anomaly and making respective decisions as to how they shall be resolved[143]. According to [144], the process of identifying and resolving an anomaly can be abstracted into the following steps:

- Problem Identification
  - Recording a problem statement
  - Recording the problem's context
- Problem Assessment/ Investigation Analysis
- Root Cause Determination
- Corrective Action/ Problem Resolution
- Problem Recurrence Control

Handling anomalies often requires not only to ingest information provided by others but to spot the inconsistencies occurring between a variety of sources. Furthermore, the required information to resolve an anomaly often can not be anticipated and prepared beforehand as this would imply anticipating the anomaly by which it could have been prevented before occurring.

The step of root cause determination is the most crucial and also most diverse step in the process of anomaly resolution. Unfortunately, literature on the topic is rather sparse:

- The NASA *Fault Management Handbook* [145], was published as draft in 2012 and never came to an official release. Its chapters on Assessment and Analysis as well as Operations and Maintenance which would describe root cause determination processes were never included in the document.
- The *Procedural Handbook for NASA Program and Project Management of Problems Nonconformances and Anomalies* [146] only states that root causes of problems need to be defined, but gives no further

reference on how such an analysis is to be conducted.

- The European Space Agency's *Savoir FDIR Handbook*[4] from 2019 references [145] for root cause determination, acknowledging that the source is only published as draft and assessing that no further documentation on the topic seems available.

This assessment was wrong however, as the *Root Cause Investigation Best Practices Guide* by Duphily et al. [147], prepared for the US-American National Reconnaissance Office published in 2014 provides detailed steps for root cause determination. A variety of methods for root cause determination are presented in [147]. Among these are various cause and effect analyses, such as

1. The Apollo Root Cause Analysis Methodology,
2. The Advanced Cause and Effect Analysis,
3. The Fault Tree Analysis
4. Process Classification Cause and Effect Diagram
5. Process Analysis Method
6. Root Cause Analyses Stacking.

The methods 1, 2 and 3 are centered around the physical composition of the system and surrounding systems showing the fault, while methods 4 and 5 are centered on the implementation processes and workmanship. Method 6, Root Cause Analysis Stacking, is the selection and combination of other methods to generate a holistic view of the problem. All of these methods require detailed knowledge of various aspects of the system, the processes of its implementation and integration as well as a thorough understanding of the physics and engineering aspects that lead to the anomaly under investigation.

As the teams contributing subsystems and instruments are not necessarily part of the same organization which integrates the system, the implementation of documentation requirements and usage of information management systems varies, leading to the challenge of finding information in a heterogeneous informational environment. Adding to the challenge are considerations for intellectual property. Not all information that may be useful for the integration of the spacecraft is necessarily accessible and may be made accessible only upon specific request. Especially the steps of Problem Assessment/ Investigation Analysis and Root Cause Determination require detailed understanding of what could contribute to an anomaly and what could have caused it.

During Phase D another knowledge transfer process is started: The operators of the spacecraft are trained. Apart from nominal operations this also covers contingency operations in case of anomalies on the spacecraft. Procedures are developed for nominal maneuvers and anticipated contingencies which contain every command to be sent and the expected response by the spacecraft.

The operations of a spacecraft are either performed by the same entity that built the spacecraft or outsourced to dedicated space operations centers or service providers. Especially in the second case, where operations are outsourced, the knowledge transfer becomes crucial, as access to internal databases of the spacecraft's testing and documents specifying the design may not be available anymore. Phase D ends when the spacecraft is launched and checked out.

2.3.5. Phase E – Operations and Sustainment

The operations of a spacecraft can take from less than a day (for example launch vehicles) to over a decade. During operations, maneuvers are to be planned and executed which require detailed procedures. Anomalies

[4]FDIR = Fault Detection Isolation Recovery

are to be dealt with, which together with planning new maneuvers form the most knowledge intensive interdisciplinary tasks during operations. Anomalies often can not be anticipated, requiring even more skill in determining the cause of the anomaly than during the integration of the spacecraft, as physical access to the spacecraft is not available anymore, and a wrong decision can mean the end of the mission [6], [7], [148].

Additionally, tacit knowledge of the developers and integrators of the spacecraft may not be available anymore, as it cannot be expected to be able to reach specific personnel again years after their work on a mission is completed. Therefore, recorded information is even more crucial than in previous phases.

2.3.6. Characteristics of a Small Spacecraft's Life Cycle

In difference to traditional, larger spacecraft for which the above described processes were developed, small spacecraft are often built on a tighter budget, with fewer personnel and more flexible mission goals but higher risk. Clements et al. [149] characterize the resulting situation as one where traditional processes can often not be implemented due to financial restrictions of the mission while missions also often suffer from failures that could have been prevented by following these processes. The tight limitations of the spacecraft's envelope and available financial budget cause missions of small spacecraft to be constraint driven rather than requirements driven. Additionally, small spacecraft are often built as educational projects or science vessels by universities or newly founded companies [150], [151]. Compared to the decades of experience in traditional aerospace companies, these entities typically have little prior experience to build on to form processes such as quality assurance, risk management and proper documentation.

It may therefore be concluded that while a small spacecraft may show less complexity due to the more limited mission goals compared to traditional spacecraft, the advantage gained thereby for managing the information in such a project is countered by the tight constraints regarding time, budget and experience of the development team, which results in a more heterogeneous informational environment than necessary.

2.3.7. Summary

In summary, knowledge intensive, interdisciplinary tasks can be found in all phases of a spacecraft's mission lifecycle. Table 2-2 provides a summary of these tasks. As can be seen, knowledge intensive, interdisciplinary tasks occur in every phase of a spacecraft mission. However, with increasing progress in the lifecycle, mistakes due to erroneous information gain in significance. Reviews are in place to catch design flaws early in the mission and the implementation of the design is thoroughly verified and validated in Phase D. Therefore, significant errors made in the design phases (up to Phase C) are usually found during the verification and validation in Phase D. While fixing such errors can be costly as they may require the redesign of parts of the spacecraft, this does not lead to an inevitable termination of the mission.

However, flaws not identified during Phase D can lead to the mission's immediate termination once the flaw occurs in orbit. The same can be said for errors made in the procedure design or anomaly resolution during operations.

Since Phase D and E are usually accompanied by a reduction in personnel and involved entities [142], the information source regarded as most valuable by [8], the direct correspondence with colleagues is often not possible anymore, as these people may not be involved in the project any longer. This means that information made available in the mission's information management systems as basis of gaining insights and knowledge even increases in importance during Phase D and E.

The overall process of informational transfer from mission phase to mission phase is summarized in Figure 2-2. The amount of applicable documents can be taken as an index to the overall amount of information relevant and necessary to complete the next phase. As can be seen in Figure 2-2, the amount of applicable documents

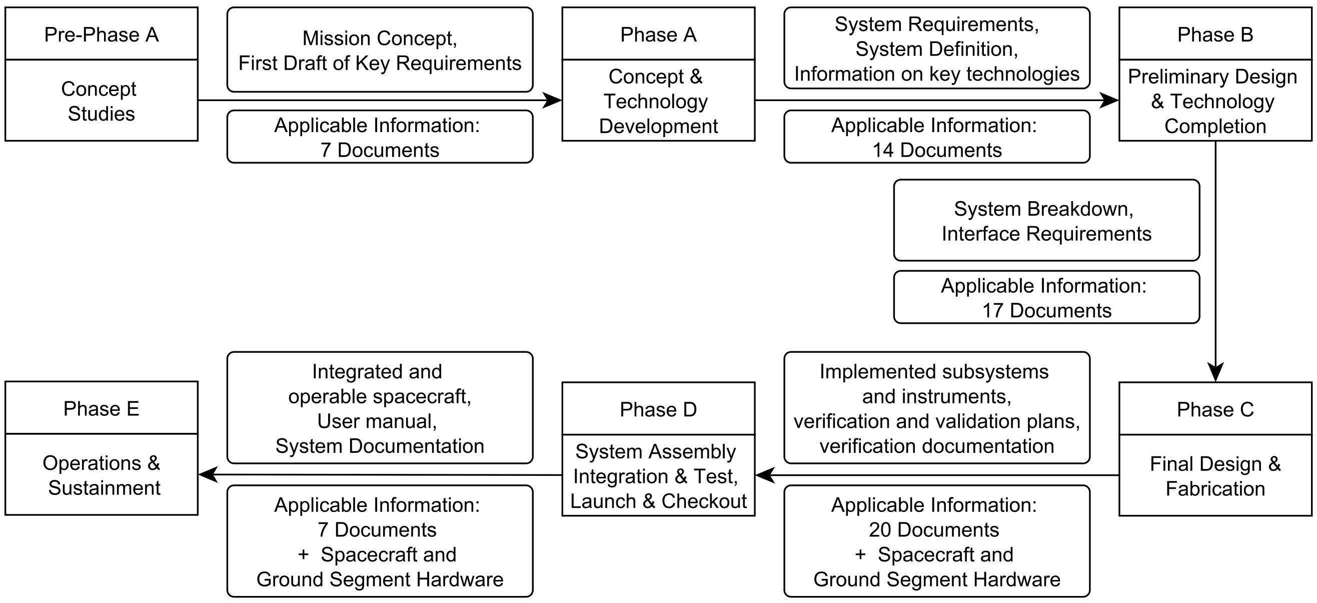

Pre-Phase A
Concept Studies
Mission Concept, First Draft of Key Requirements
Applicable Information: 7 Documents
Phase A
Concept & Technology Development
System Requirements, System Definition, Information on key technologies
Applicable Information: 14 Documents
Phase B
Preliminary Design & Technology Completion
System Breakdown, Interface Requirements
Applicable Information: 17 Documents
Phase C
Final Design & Fabrication
Implemented subsystems and instruments, verification and validation plans, verification documentation
Applicable Information: 20 Documents + Spacecraft and Ground Segment Hardware
Phase D
System Assembly Integration & Test, Launch & Checkout
Integrated and operable spacecraft, User manual, System Documentation
Applicable Information: 7 Documents + Spacecraft and Ground Segment Hardware
Phase E
Operations & Sustainment

Figure 2-2.: Information transfer process between the phases of a space mission. Applicable documents according to [140]

increases up to Phase D and shows a steep decrease between Phase D and E. Partially this is due to the fact that the purpose of the information is fulfilled, and a document may therefore not be applicable anymore. For example documentation required by the launch provider to safely deliver the spacecraft to orbit is not necessary anymore once the spacecraft is launched. The rest of the information used as input for Phase D has to be condensed from a wide variety of sources to a few key documents, such as the user manual of the spacecraft or the contingency procedures. One more challenge for information management arising with Phase D is the competition between documents and implemented spacecraft for the source of truth. Any non-conformance between the spacecraft and its documentation that is found during Phase D needs to be documented and assessed, whether the non-conformance has to be changed in the spacecraft's implementation or whether it suffices to update the documentation accordingly [143], [144].

Table 2-2.: Overview of knowledge intensive, interdisciplinary tasks in the mission lifecycle of a spacecraft

Mission phase	**Knowledge intensive, interdisciplinary tasks**
Pre-Phase A, Concept Studies	▪ Overall mission design ▪ Timelines ▪ Feasibility analyses ▪ Definition of key performance requirements
Phase A, B, C Concept and Technology Development, Preliminary Design and Technology Completion, Final Design and Fabrication	▪ Complex functions requiring the collaboration of multiple subsystems and instruments ▪ Interfaces ▪ Budgets ▪ Architecture and requirements management
Phase D System Assembly Integration and Test Launch and Checkout	▪ Anomaly Detection and Resolution ▪ Verification and Validation ▪ Generation of User Manual and Procedures ▪ Generation of Contingency Plans ▪ Spacecraft Checkout
Phase E Operations and Sustainment	▪ On-Orbit Maintenance ▪ Generating new procedures ▪ Anomaly Resolution

3. Gap Analysis and Derivation of Hypothesis

This chapter defines shortcomings of existing information and knowledge management systems based on the necessities of a small spacecraft's lifecycle. Focus is laid on information-intensive tasks that require system-level knowledge.

3.1. Gap in the Capabilities of Information Management Systems with Regard to Space Missions

In Section 2.2, various information management systems are described, including studies on strengths and weaknesses of the respective systems.

Design Rationale Systems, Knowledge Centered Services, Wikis, collaboration management systems and classic approaches to Model Based Systems Engineering all require discipline and commitment from all involved users, for example on when information is to be updated, or where and how a certain type of information is to be stored/communicated. The vulnerability to users that do not comply to these requirements makes these systems brittle and prone to becoming ineffective. Once it cannot be trusted anymore that the information retrieved from such a system is accurate, the chance that users rely on direct communication with their peers again to retrieve information increases. This again exacerbates the problem of curated documentation becoming untrustworthy, resulting in a vicious cycle.

Team communicators – compared to the systems mentioned above – are built around the direct communication between users. By applying a construct of channels for technical topics, to which users involved with the topic can subscribe, information is easily shared across the team and the chance of the same question being asked twice reduced, as any subscriber to a channel reads all communication shared within. Similar to the content review processes in wikis, the chances of getting reliable information via a team communicator are good, as the information distributed by one person is directly reviewed by their peers upon reading. However, a team communicator can only be a part of a larger solution. As external partners usually do not have access, emails remain a competing method of communication and the information stored in a communicator lacks in formality.

Most information is shared in a question/answer style. This means the information can be found in bits and pieces and the density of information is rather slim, which reduces the efficiency. A disadvantage of team communicators is that an increasing use leads to a decreasing efficiency. As it is impossible to judge the relevance of new information in a communicator before reading it, every member of a channel reads every new message, even when they are not relevant to them. The alternative is creating channels for every specific topic and only inviting those people who are interested, which can result in a loss of overview, having to search through a multitude of channels one was not part in and the same or similar topics being discussed in parallel in multiple channels. Therefore, maintaining the communication happening via a team communicator requires time from every team member that can be spent else wise. Looking at the phase-wise character of a spacecraft development with tasks that may be distributed across multiple partners, the application of communicators comes with an additional hurdle: To be effective across phases, all team members are required to have access to the communicator. This applies especially when responsibility for systems is redistributed, i.e.

at the commissioning of a new partner for the implementation of a part of the spacecraft or the integration of a commissioned part with the rest of the spacecraft. As most entities have their own entity-specific communication management, a new system to be specifically used for a certain project again encounters the problem of competing channels of communication, which are not even accessible to all members of the project.

Enterprise search engines and graph database applications such as described in Section 2.2.8 applied as central information repositories are the only information management systems described in Section 2.2 able to resolve the problem of alternate communication paths and information storage possibilities, as they are the only systems connected to all information repositories used in the development, which may well include a combination of other information management systems. They are thereby also the only approaches that allow for the identification of outdated information, as – independent of the communication path – new information on a topic could be retrieved via these systems. Solutions based on the Open Services for Lifecycle Collaboration standard may be used to omit the occurrence of the same information being stored twice if applied consistently throughout the project. This requires all partners in a project of the size of a spacecraft development to be convinced to work with Open Services for Lifecycle Collaboration and to ensure all team members at every involved entity apply Open Services for Lifecycle Collaboration correctly to the information they share, which is unlikely to succeed. However, as pointed out in Section 2.2.6, the search in an enterprise search engine requires a fine-tuning of algorithms to find the right information which may change depending on what you are looking for and which relies on respective ontologies. As proposing systems that require more time and resources than they save lacks economical appeal, developing an ontology only used for the search engine is not attractive.

While introducing the thought of applying a system-level model as basis for the search, the proposal by Jones et al. [79] to employ a CAD-Model leaves out any aspects such as requirements, electronics or software, which are not part of a CAD-Model's part tree but highly relevant. Furthermore, enterprise search engines lack in tunability of the queries. A text field search requires too much interpretation of information that can be well expressed in a formalized query (such as stating that the information could be attributed to a certain component or any of its subcomponents). This requires some understanding of what queries the users may pose. An enterprise search engine would thereby have to be adapted in multiple ways to allow to properly query information with the already existing knowledge of the users.

Graph databases applied as central information repositories in theory allow for more elaborate queries on the stored information. However, the examples presented by Petnga [135] and Bajaj et al. [134] only show basic queries on the graphs, most of which could also be answered by any relational database storing the same information. Thereby they are far from leveraging the capabilities and possibilities granted by a graph database. The reason for this may be attributed to their respective use cases: The goal of Petnga's research was to find a possibility to judge the completeness, consistency and correctness of requirements specified in a pre-existing SysML model, not to gain insights on complex networks of relations. Bajaj et al. aim their research at providing a general purpose database system that connects the various development artifacts and a SysML model of the system and see the problem solely from the perspective of a tool provider. Both use pre-existing SysML models in their research and hence the quality of information they are able to retrieve has a natural upper limit beyond which it would be necessary to make assumptions on the way the SysML Model is built up and thereby lose general applicability.

Comparing graph database based information management systems with other information management strategies it becomes apparent however, that graph databases yield a key advantage: as seen in the excursion in Section 2.2.8, it is possible to organize information via a graph database without any compliance by those who create the information. Therefore, such a system does not have the same vulnerability to a lack of user compliance as Wikis, collaboration management systems and others.

3.2. Capabilities of Information Management Systems vs. Needs of Information Management during System Assembly Integration and Test, Launch and Checkout of a Spacecraft

Comparing the state of the art of information management systems and applied methods with the life cycle of a spacecraft, it becomes apparent that Phase D of the development cycle forms a pinnacle for information management in a spacecraft project:

- The tasks distributed to multiple partners over the previous phases of the project coalesce in the form of hard- and software to be integrated in Phase D.
- The information flow of the previous phases is reversed. Information is provided by the various partners about the to-be-integrated hard- and software to the team conducting the integration, whereas in previous phases information was distributed further to contractors and internal teams.
- Information that is actually lacking may become apparent for the first time, as it is the first time for most systems to be tested by someone who was not involved in their development and also the first time different subsystems are to be integrated with each other. I.e. interfaces that only fit in theory beforehand now have to be tested against each other.
- Even a component implemented in exact keeping with its specification may be flawed, if the specification of the component was flawed.

Additionally, new information about the spacecraft's structure and behavior is gathered through the tests. This information requires careful consideration and comparison with the provided documentation of the spacecraft components and their specification. All of these aspects require a rigorous information management. The fact, that the project has progressed quite far at this point enhances problems experienced with information management solutions implemented in previous phases. The team for the integration cannot start on a clean slate in terms of information management, but needs the information that should have been properly managed in the previous phases. As elaborated in Section 2.2.4, with decreasing enthusiasm and increasing amount of work and delay in a project, information management strategies often fail in the course of a project due to a lack of commitment and discipline.

During Phase D, processing anomalies is the most knowledge intensive, interdisciplinary task. Understanding all components potentially involved in the anomaly and how to validate whether they are operating nominally or are part of the anomaly requires a deep understanding of the spacecraft itself, which has to be gained via the documentation provided with the components, previous experience with other anomalies and further investigative steps which are to be defined based on the already existing knowledge to gain new insights.

Testing a spacecraft can take years [142] and the mission afterwards may span more than a decade, both of which make good information management essential. For small spacecraft where schedules are significantly shorter, the testing phase is no less significant. The failure rate in small spacecraft is higher than for regular spacecraft by an order of magnitude as described by [152].

This observation aligns well with the findings of Section 2.2.7, showing that the majority of applications of MBSE are in the early phases of a project, where the complexity is still low as well as the amount of involved personnel and only few reports of successful application of MBSE in later phases exist.

Combining the findings of above; that information management systems with a few exceptions are brittle against non-compliance with the respective processes to maintain them and the needs of information management during Phase D of a spacecraft project, it becomes apparent that the needs of information management in Phase D are not covered by state of the art information management systems.

Considering SysML Models translated into graph databases, a lot of information that requires engineers to deeply understand the system they analyze could be gathered via a graph-transformed SysML model. The current state of the literature shows little to no regard to such topics however and only provides the most basic query capabilities. This may be due to the lack of a corresponding set of modelling rules and a graph schema specifically developed to allow complex queries on the interrelations of an embedded system such as a spacecraft. Root cause analyses as described in Section 2.3.4 however require capabilities such as:

- Querying all components (of hardware and software) transmitting a certain (faulty) telemetry and returning the documentation of these components, as well as any other input or output related to each component
- Understanding all conditions that can lead to an observed state of the system,
- Conducting fall-out analyses which show which components would be affected by a short-circuit or blackout at a certain position in the system.

Defining queries for a graph database that allow answering these questions requires investing time and personnel. However, compared to the traditional approach of gathering the information necessary for root cause analyses in spacecraft missions by hand, the queries can be parametrized and thereby reused on any following anomaly resolution. Apart from detailed state machine analyses, no automatized or queryable systems exist which can be leveraged to conduct fault analyses in spacecraft projects. While the methods described in [147] are valuable tools to conduct root cause analyses, all methods presented still require that the intricate connections and influences within the spacecraft are known to the engineers conducting the analysis or have to be retrieved from documentation by hand, which due to the amount of available information can be a tedious and error-prone task.

3.3. Hypothesis

The gaps described above, linked with the foray on applying graph databases to retrieve, organize and relate information in the profession of investigative journalism leads to the following hypothesis:

The integration of graph databases with MBSE allows tasks requiring interdisciplinary knowledge and system understanding to be performed efficiently and effectively during the assembly integration and testing as well as operations of a small spacecraft and further allows reusing acquired capabilities in the analysis of such systems effectively.

In a graph database, the SysML model of the spacecraft's implementation can be combined with all digital informational artifacts available on the spacecraft, independent of their source or format. Thereby independent of the communication channel used or position the information is filed, it can be retrieved for the spacecraft's integration.

4. Research Layout

To prove the hypothesis presented in Section 3.3, an information management system based on a graph database and the SysML Model of a spacecraft's implementation has to be developed and tested. This requires several steps.

1. To allow for the integration of the information management systems already used in the previous phases of a spacecraft mission, more information on which systems are in use in space missions is necessary. This information shall be acquired via a survey. Furthermore, the processes of informational exchange within a spacecraft development need to be better understood. As the field of possible agents is large, the survey shall be reduced to the developers of small spacecraft.
2. In order to integrate a SysML Model in a graph database, a so-called graph schema, which defines how the elements of the SysML language are to be translated into the graph is necessary. Defining such a schema requires to define the questions it shall be able to answer beforehand, however.
3. Rules for creating the SysML Model need to be defined, by which a model can be created in consistency with the graph schema. The rules have to enable answering the questions defined beforehand.
4. Respective software needs to be developed which can extract the SysML model from its files and loads it into the database according to the before designed graph schema.
5. An extension of the graph schema for the informational artifacts of the development has to be designed and validated against the informational environment of an actual development.
6. As creating software to extract and translate the information from all identified information management systems into the graph is too large a task, exemplary software to translate the artifacts of some information management systems into the graph shall be written.
7. Analyses as to how the information from the various information management systems can be connected in the graph database with the SysML Model shall be conducted.
8. The analysis system shall be validated on different datasets. These shall include actual space missions to show the benefits and challenges created by the application of the previously defined theory.
9. Finally, an analysis shall be conducted to compare the original goals and the results which could be validated and verified.

The following chapters are structured according to the research layout presented above. Hence, the next steps are to analyze the results of the survey conducted for Step 1 and afterwards derive a graph schema allowing to manage complex and knowledge intensive tasks within the later phases of a spacecraft's lifecycle.

5. Survey on Information Management in Small Satellite Projects

As discussed in Chapter 3 and Chapter 4, more information on how information is managed in small satellite projects is necessary. A survey gathering this information was designed and conducted over 2019 and 2020.

5.1. Survey Objectives

The survey published by Berquand et al. in 2019 [8] brought various insights on how developers work with information in the early phases of a space mission, especially during Concurrent Design Sessions:

- The majority of CDF engineers use between 25% and 50% of their working time to acquire information.
- The information source identified as most reliable is the direct correspondence with colleagues.

As the participants of the survey stem from a small group (all work for the European Space Agency in Concurrent Design Studies), the results cannot be generalized to spacecraft developers or even small spacecraft developers. Hence, the first objective of the survey is to validate or disprove the findings of the study conducted by Berquand et al. in a broader field of spacecraft developers. Therefore, the same questions asked by Berquand are asked again in this survey but to a broader audience.

The second objective stems from the gap in information management systems to identify outdated information as described in Chapter 3. The state of the documentation in their projects shall be assessed by the developers to identify possible practical deficits.

A third objective of the survey is to characterize the informational environment and processes in small spacecraft developments. Therefore, questions are asked regarding which information management systems are used by the developers' teams and how information is shared within the team.

The fourth objective of the survey is to either validate or disprove the hypothesis of the vicious cycle of project delays. Therefore, the current state of delay in the project is assessed and shall be compared with the state of the documentation. In case such a vicious cycle can be proven, a correlation between project delay and amount of missing or outdated documentation, or dependency on coworkers for information or how well the documentation is trusted should be visible.

5.2. Survey Method

The survey was conducted during the Small Satellite Conference 2019 in the United States of America, where participants who actively build or operate a small spacecraft were asked to fill out the survey. The survey was provided as online form and the participants were supported by staff who answered any open questions on how to understand certain aspects of the survey. Operators were included in the survey as well as developers as they are a crucial part of any spacecraft mission and knowing a spacecraft (i.e. good information management) is a critical requirement for successful operations.

29 responses were gathered over the conference. A large population size of small spacecraft developers[1] is assumed, which is justified according to Kulu [153]. This results in an error span of 18% at a confidence level of 95% [154]. The results of the survey can hence only be interpreted on a qualitative level.

5.3. Survey Results

The results presented in the following are always to be read with an error span of 18% at a confidence level of 95%.

5.3.1. Characterization of Participants

The participants were asked a series of questions to categorize their project. This allows to realize trends within the data, for example small teams could have less information management systems in use, or governmental missions could take the longest. The questions asked in this part of the survey are the following:

- How many people work on your project?
- Do you work in operations or development?
- Which level of experience describes your team best?
- What is the average retention time of your team members?
- How long does your project run already?
- What delay do you estimate your project to still accumulate relative to the timespan until the spacecraft has to be handed in?
- What share of your system's design is done by external contractors?

Of the 29 participants, 3 work in operations, 25 in the development of the spacecraft and one in both. Figure 5-1 depicts the distribution over various mission types[2]. While the amount of participants is rather diminutive, the field of participants is quite diverse. No more than two participants came from the same entity, with 27 spacecraft building or operating entities represented in the survey.

The mission types are characterized as follows:

- Constellation: multiple spacecraft of the same type are developed and launched over a range of time which may span several years. Typically, multiple spacecraft are in development at the same time with only minor changes from one revision to the next [155].
- Startup: Startups are characterized by a fast-growing workforce, a low amount of established internal processes and a high pressure to create revenue.
- Governmental institutions: These institutions form the opposite to startups, as they can be characterized by well established processes, a steady workforce and lower pressure compared to startups to create revenue.
- Satellite Bus providers and integrators: Entities which provide the integration qualification, launch and operation of spacecraft as a service to scientists and companies that are focused on instrument design and do not have the capacity to build their spacecraft.

[1]With a population size of 1000 or larger, error span and confidence interval stay the same
[2]Ticking multiple mission types was allowed

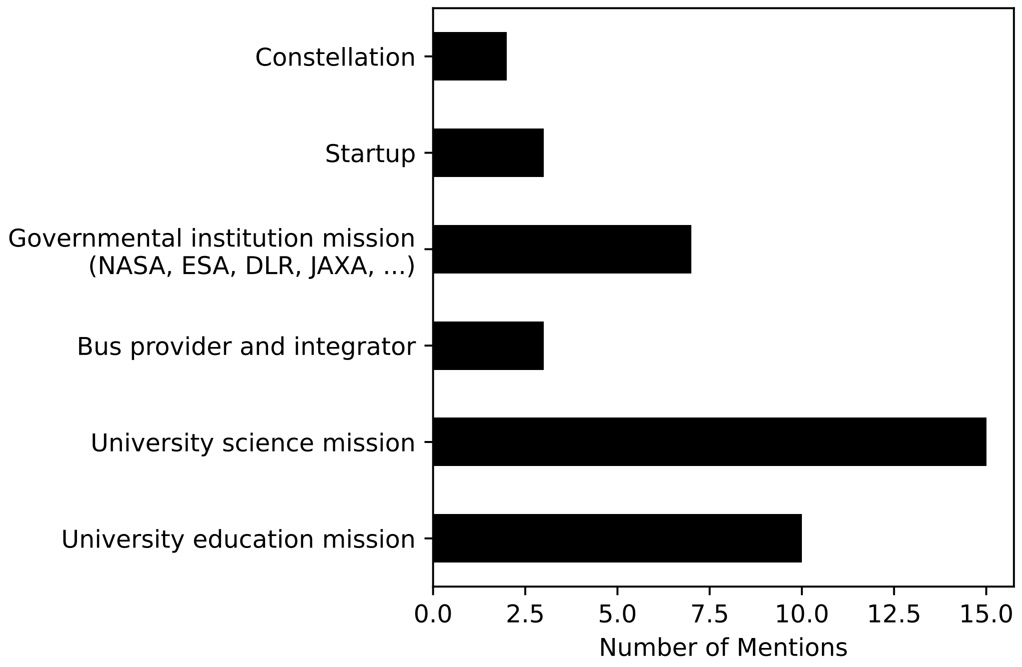


Figure 5-1.: Background of survey participants

- University science mission: The spacecraft is built by university staff. The missions are characterized by a paid and stable workforce (compared to University education missions).
- University education mission: The spacecraft is built by students either in the form of scientific theses and seminars or as an extracurricular volunteer project, typically under the guidance of few scientific staff. These missions are characterized by little to no prior experience by the workforce and a short average retention time of the participating developers.

Out of the 29 developers surveyed, 17 work in a university setting[3], 7 work for governmental agencies, while 5 participants have a commercial space background. This implies an underrepresentation of commercial spacecraft developers, compared to their relative share in launched spacecraft (commercial: 981, University: 486, Governmental Institutions: 160) [150][4]. Compared to the share in entities conducting small satellite missions, the underrepresentation can be confirmed: According to [150], 419 companies were active in building and operating small spacecraft until 2019. Extrapolating data from Berthoud et al. [156], 192 academic programs had launched their own spacecraft until 2019. The reason for this imbalance is the availability of participants: Students and university staff were more willing to share insights into their work than industry representatives. It therefore has to be concluded, that the dataset favors university-built spacecraft while underrepresenting commercial spacecraft with a factor of 5.

The average retention time of team members and their previous experience are important boundary conditions for knowledge management within a team. Figure 5-3 displays the average retention time of team members for each project, while Figure 5-2 provides insights into the previous experience in spacecraft design.

The majority of projects can rely on previous experience, either by themselves or at least by other members of their institution. All governmental and industrial participants logged previous experience of a part of the team or the whole team.

[3]Seven participants characterized their missions as education as well as science missions
[4]Data from 2021-11-17

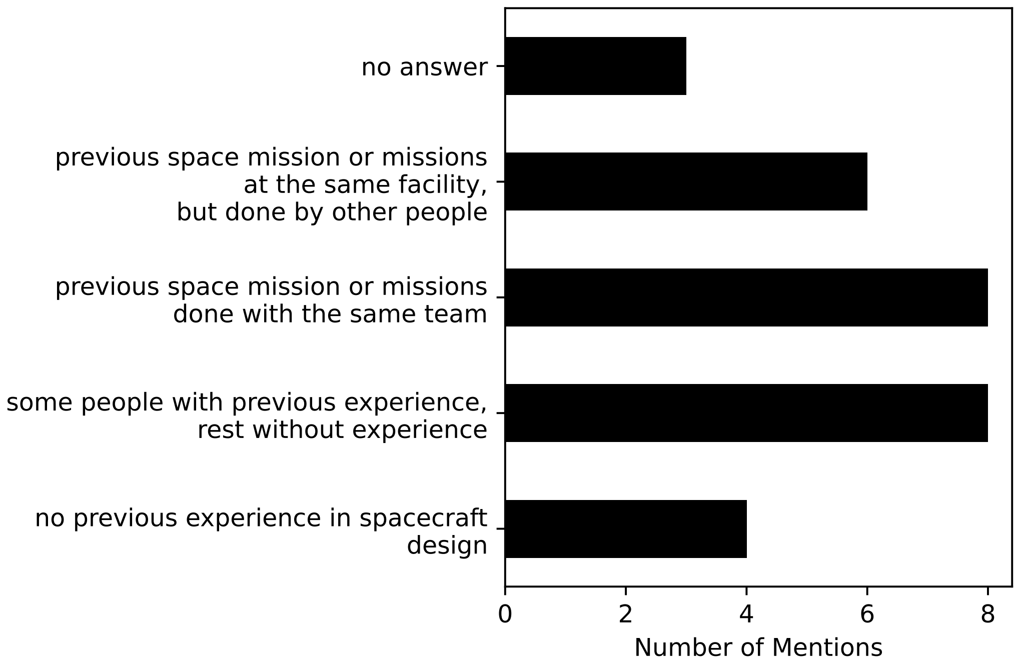


Figure 5-2.: Team Characterization of survey participants

Apart from the average retention time of team members, the time the project runs already was taken as another indicator. As presented in Section 2.2, most information management systems have a good start, but become less effective as the amount of outdated information grows. Also, the longer a project runs, the more information has to be managed. Figure 5-4 shows the project runtime at the time of the interview, with an even distribution. The average retention time of project members displayed in Figure 5-3 on the other hand shows that team members in most projects stay more than a year on the team. No correlation could be found between short retention time and short project runtime.

Another topic relevant to the use of information management systems is the amount of people working on a project. The more people work on a project, the more difficult it becomes to establish new rules or policies and also the more important good information management becomes. Figure 5-5 shows the distribution of participants over team sizes. While the overwhelming majority of teams have less than 40 members, outliers with larger teams exist and are to be taken into account. The outliers in the survey all stem from university education and science missions. However, due to the small sample size this insight cannot be generalized.

Another question relevant to the use of information management systems is the share of a system's design done by external contractors. External vendors are often not granted access to internal information management systems due to intellectual property considerations. Therefore, projects with a higher share of external contractors could yield a different landscape of information management systems in use. Figure 5-6 provides the respective data.

The last question to characterize the participants asked was their estimated project delay regarding their current schedule. Figure 5-7 shows that a clear minority of projects are on schedule (5 out of 29), while a third of the projects report major delays (estimating over 25% more time than still available until the end of the project).

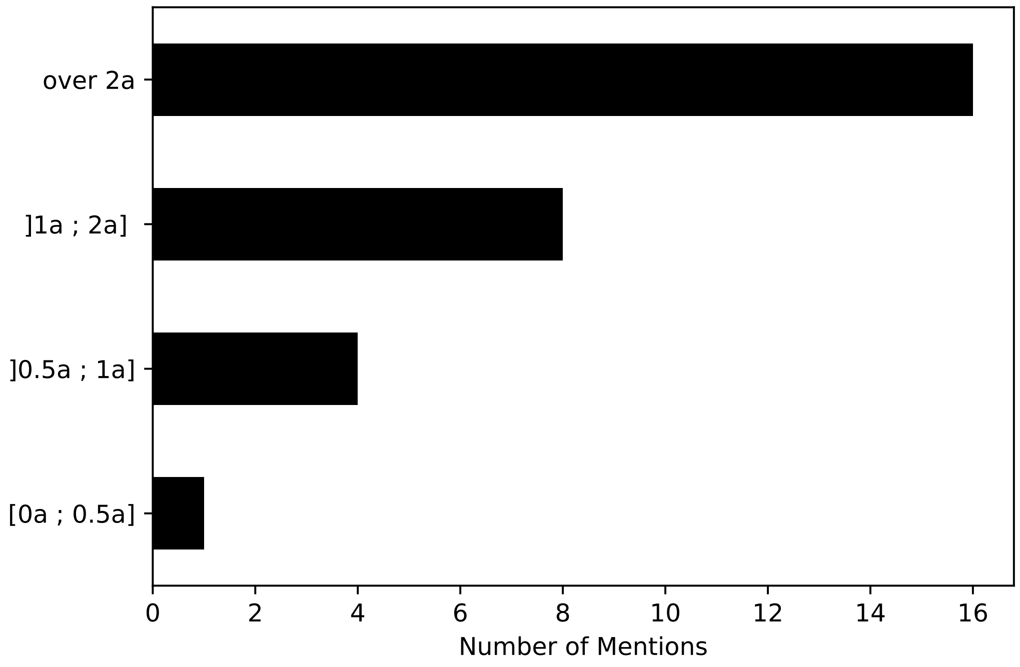


Figure 5-3.: Average retention time of team members

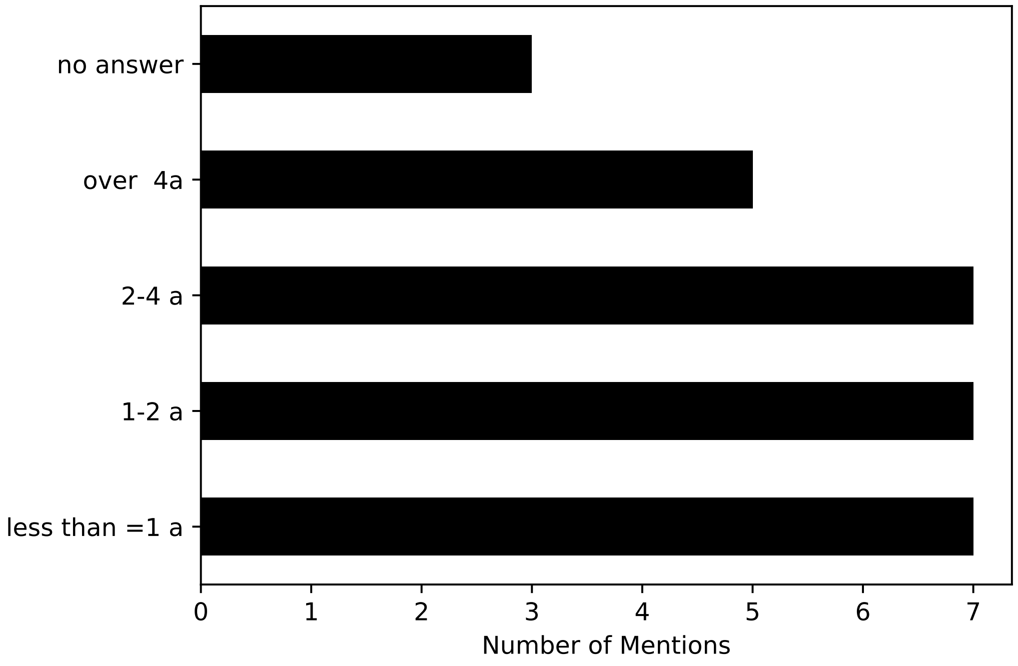


Figure 5-4.: Current project runtime

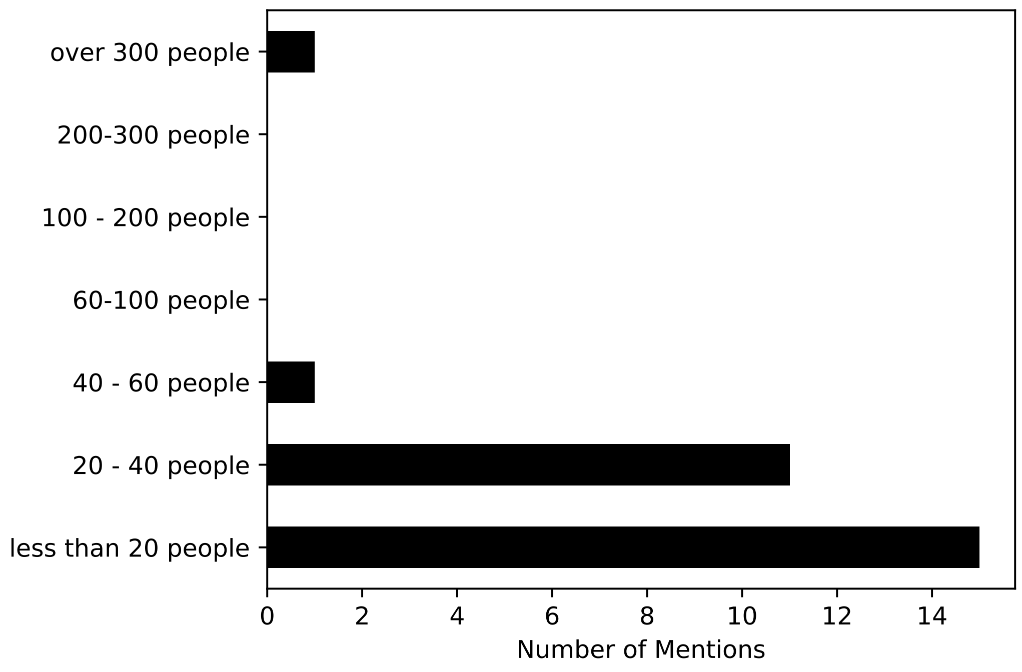


Figure 5-5.: Team size of survey participants

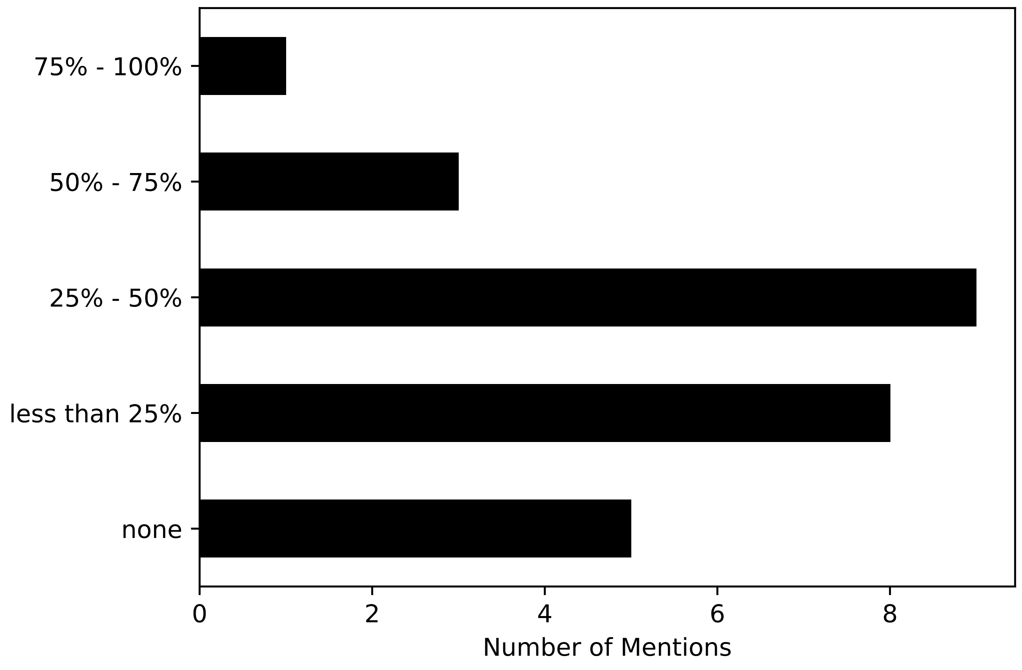


Figure 5-6.: Share of the design done by contractors

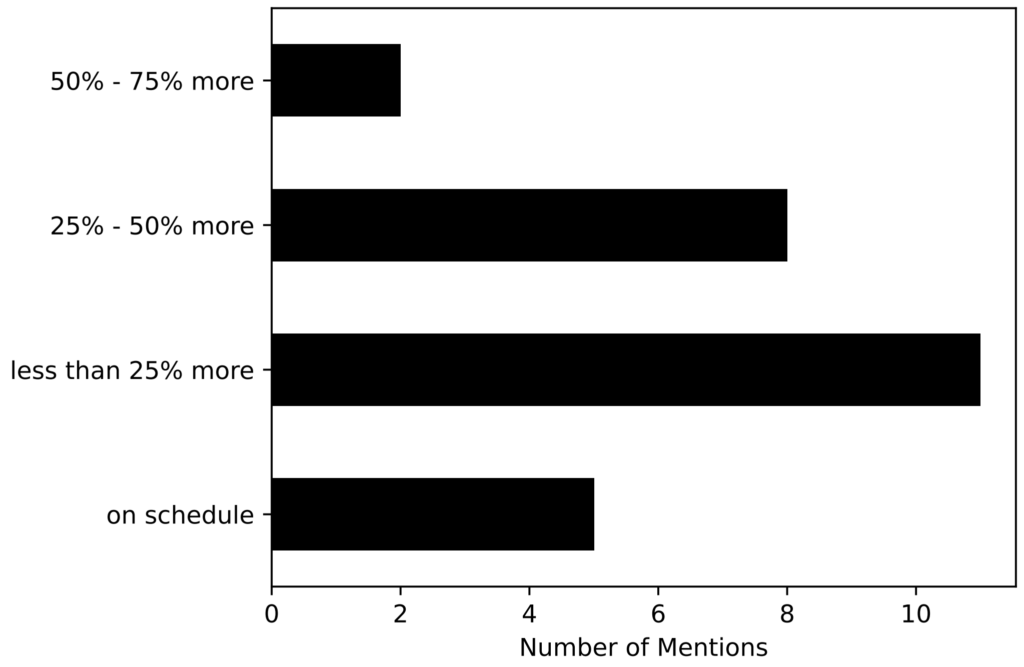


Figure 5-7.: Estimated current project delay relative to the remaining development time

5.3.2. Objective 1: Comparison with the Results of Berquand et al.

The study published by Berquand et al. in 2019 [8] gave insight into the part of the work time spent researching through available information and which sources were seen as most useful. Similar questions were posed to the developers of small spacecraft. The results can be seen in Figure 5-8 and Figure 5-10. As Figure 5-8 shows, small spacecraft developers tend to spend less time looking for information than their colleagues at the European Space Agency's Concurrent Design Facility. On the other hand, as Figure 5-9 shows, small spacecraft developers spend considerable time on creating documentation, on average spending roughly one day per week with creating documentation.

Figure 5-10 compares the sources of useful information between the data from Berquand et al. to the data gathered in the small spacecraft developers survey. While the options for answering were kept equal for comparison, the data of Berquand et al. seems to be gathered with the limitation of choosing only one option, whereas the small spacecraft developers survey allowed multiple mentions. As explained above, the data only allows for qualitative analyses in any case. Both studies name the direct discussion with colleagues as the most valuable resource. Both studies agree that no important source of information was left out, as the option *other* has the least popularity. But while Berquand's study names previous missions reports as the second most useful information source, the small spacecraft developers survey identifies public resources in the second place and puts the current project's documentation above the fourth place it holds in the study by Berquand et al.

A possible explanation for the differences lies in the project phase. The data of Berquand et al. is focused on the early mission phases where concurrent design sessions are typically employed as a method of development. The participants of the small spacecraft developers survey over project runtimes is quite even in comparison, see Figure 5-4. The goal of concurrent design sessions is to explore a variety of design options for a space mission. Therefore, the focus on previous mission reports over the current project's documentation makes sense. In later project phases more restrictions apply to the design work, which are recorded in the project

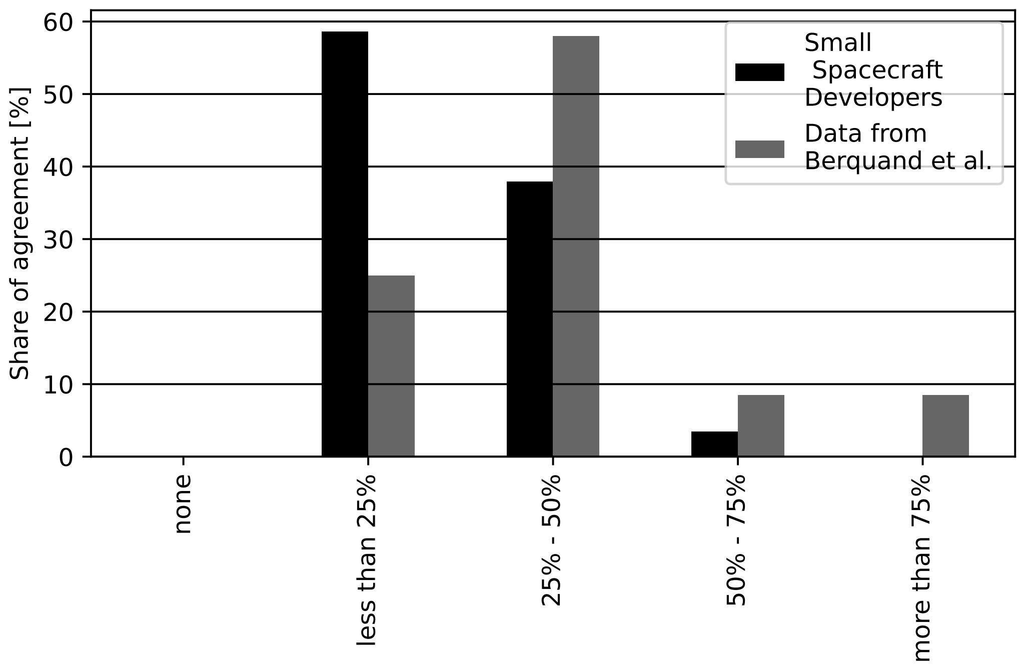


Figure 5-8.: Share of the daily work time spent looking for information compared to Berquand et al. [8]

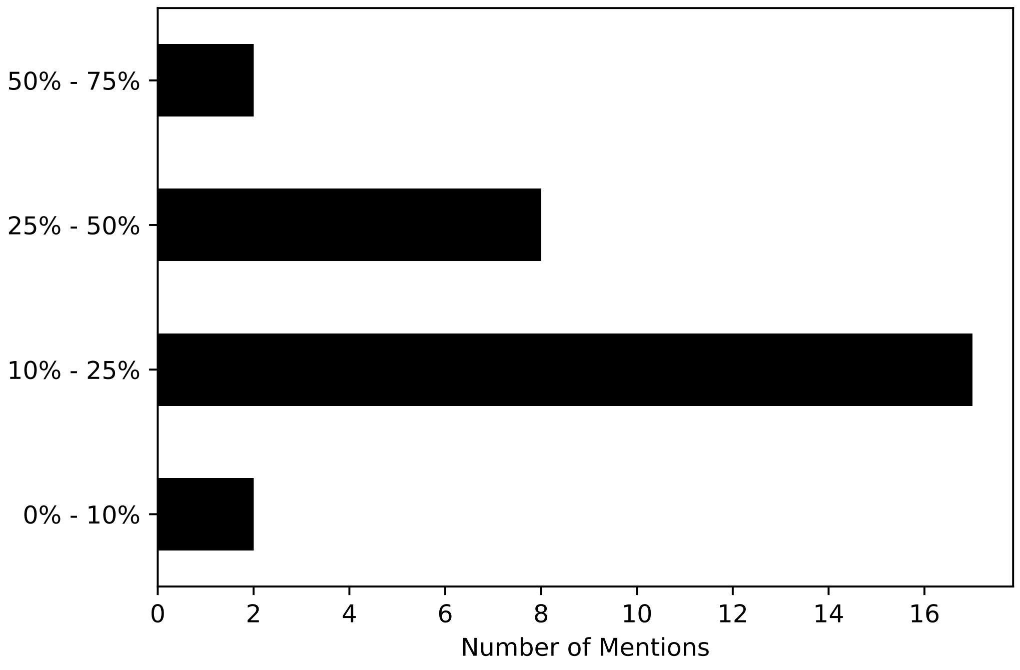


Figure 5-9.: Share of the daily work time spent writing documentation

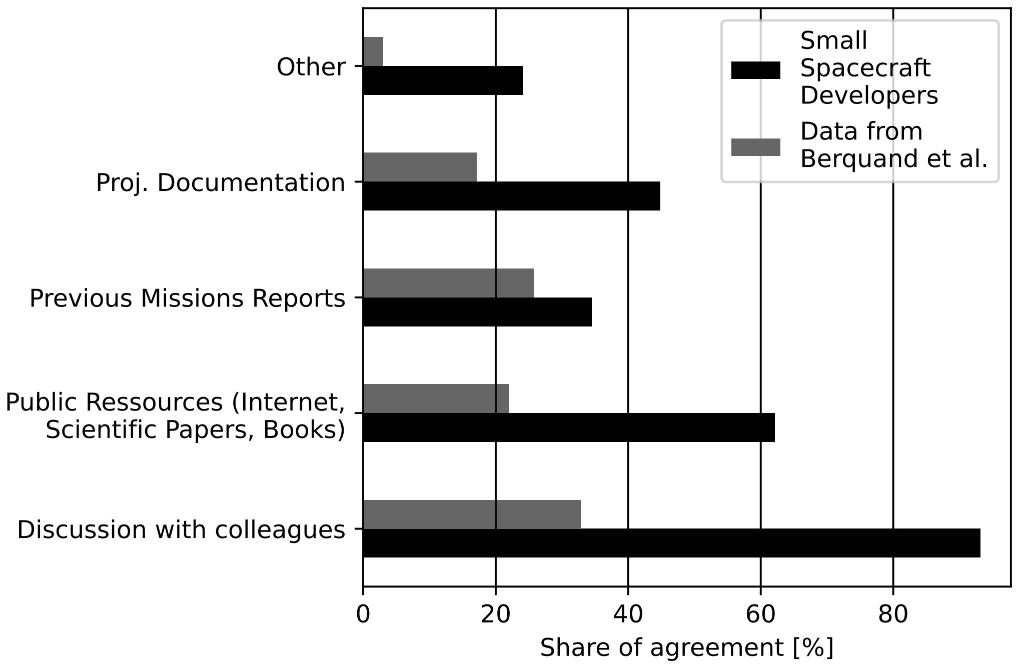


Figure 5-10.: Most useful sources for information compared to Berquand et al.

documentation.

In summary, the results compare well. Both name the discussion with colleagues the most valuable source of information and show that a significant part of the daily work time of developers is spent acquiring new knowledge. Acquiring information via discussions however requires even more time spent on informational exchange, as someone need to share the information in the first place. It goes to show that even minor efficiency increases in information retrieval are beneficial for the overall project runtime and by extrapolation to the iron triangle of funds, time and quality might yield further benefits, for example in the quality of the developed system. Both surveys further show an open potential for available project documentation. Only 13 out of 29 interviewees see their project's documentation as one of the most useful resources, which means that 16 interviewees do not see their project's documentation as a valuable resource.

5.3.3. Objective 2: Assessing the State of Documentation

As elaborated in Chapter 1 and 2.2, even partially incomplete and outdated information make a corpus of information unreliable if it cannot be determined which part of the information is outdated or missing.

Asked for the share of documentation that is currently outdated, 12 interviewees responded with 25% to 50%, while 17 responded with less than 25%. No participant judged their documentation to not be outdated.

To further judge the situation of currently available documentation, the survey participants were asked for their agreement to a series of statements, see Figure 5-11. The options range from a strong disagreement (1) to strong agreement (5). A clear majority of participants do not consider their documentation complete, but show a relatively high amount of trust in the retrievable documentation. Statement three however shows that the majority of participants do not consider themselves to know where the latest information regarding a topic is stored. This shows that the variety of possibilities how to communicate and store information is problematic.

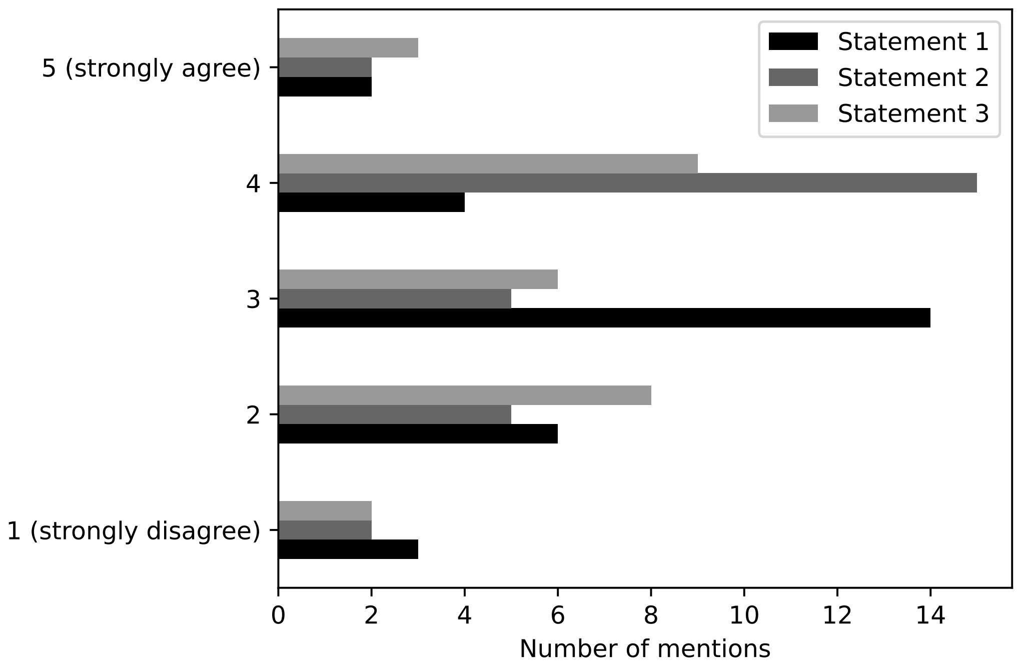


Figure 5-11.: Agreement to statements regarding the current state of documentation. Statement 1:"The documentation of our project is complete with regard to the state of development.", Statement 2: "I trust the currently retrievable documentation of our project to be correct." Statement 3: "Looking for information, I always know where the latest information regarding the topic is stored."

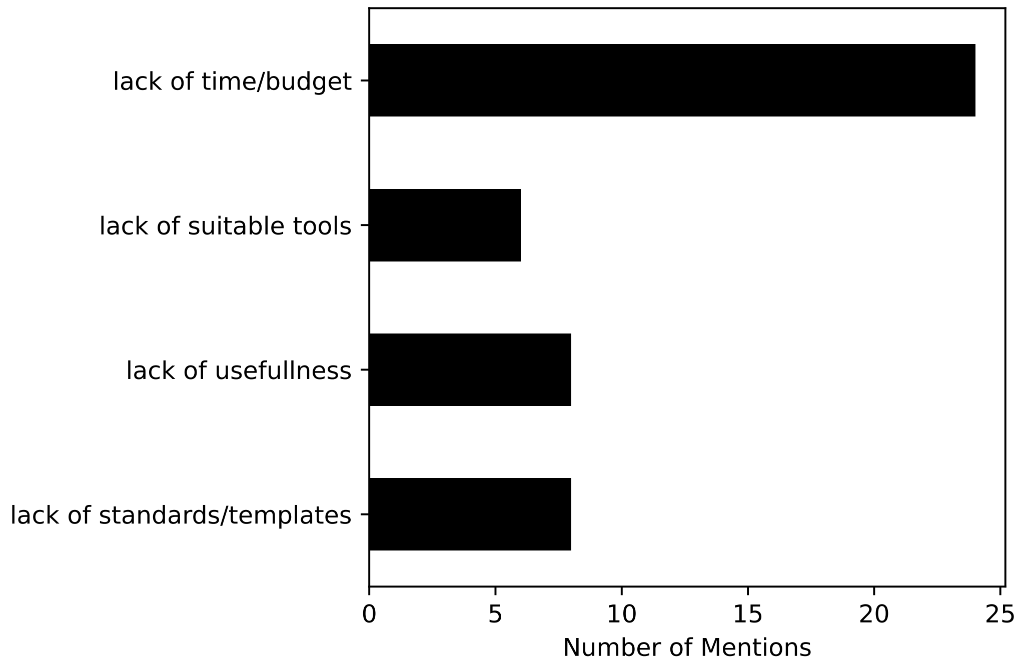


Figure 5-12.: Reasons to not update outdated documentation

Combining the findings of questions regarding what share of the project's documentation is outdated and what keeps developers from updating the documentation, insights in the reasons for outdated documentation can be gathered: Seventeen answered with "between 1% and 25%", while twelve answered with "between 25% and 50%" regarding the share of outdated documentation. Figure 5-12 summarizes the provided reasons for not updating the documentation. More participants see a lack of time or budget as the reason for not updating their documentation than for all other three provided reasons together.

Figure 5-13 shows that developers also spend a considerable share of their work time writing documentation. Combining the three findings, that a sizeable portion of the documentation is considered to be outdated, that the majority of developers see a lack of time/budget as the reason why their documentation is outdated and that they still spend a large portion of their work time writing documentation leads to multiple conclusions:

1. Since the documentation is neither correct nor complete, personal inquiries are mandatory to assess the worth of written information.
2. The efficiency of working with documentation and information has a high impact on the overall work efficiency, as developers spend a considerable share of their work time writing and retrieving information.
3. Since lack of time/budget is the main reason for documentation to be outdated, tools that reduce the time to identify outdated information or to find the most up-to-date information on a topic will increase the efficiency of working with documentation/information.

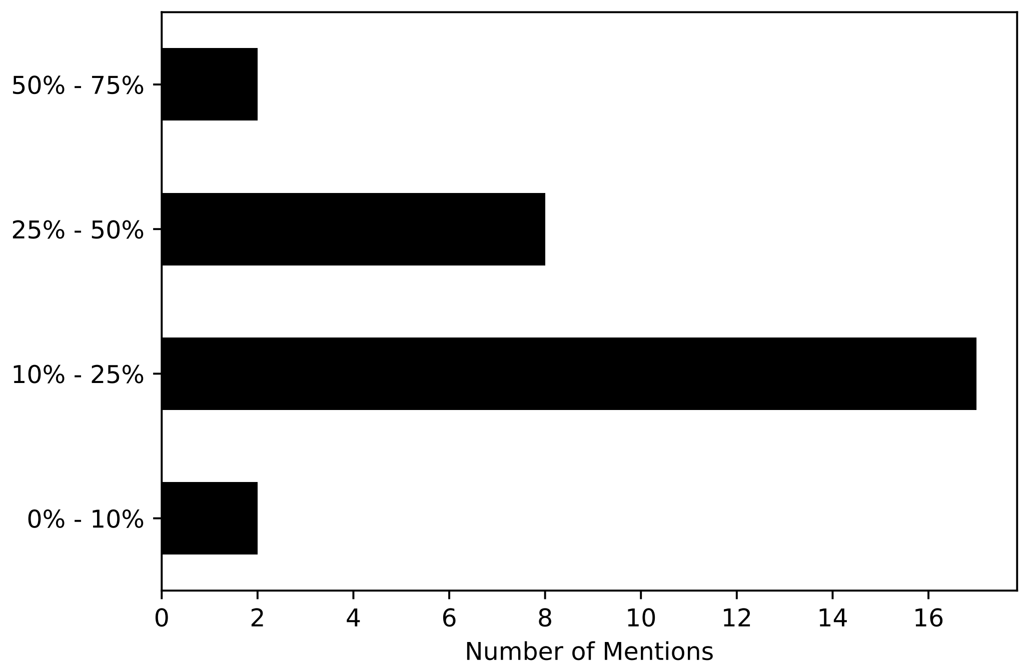


Figure 5-13.: Share of work time spent creating documentation

5.3.4. Objective 3: Assessment of Tools and Processes

In a further question, the participants were asked to tick off information management systems in use in their projects and name any additional information management systems not specified in the survey that are in use. Figure 5-14 shows that the majority of teams employ four to six information management systems.

Another information that can be taken from this question is the popularity of various information management systems. Figure 5-15 shows the popularity for the most mentioned software tools. Collaboration Management Systems are the most mentioned information management systems, together with central file storage systems. Just over half of all participants employ team communicators. Wikis and personal communicators make up the rear, with nine and six mentions respectively. Not one team employs an enterprise search engine or a design rationale system. The low popularity of wikis can be attributed to the high popularity of collaboration management systems, which typically include wiki-like features and therefore integrate their functionality in a broader set of functions. Similarly, collaboration management systems mirror some functionality of design rationale systems, as each contribution in the system requires users to add some comment on what changed with this contribution, resulting in a fine-grained list of rationales for any software written for example.

Furthermore, the workflow associated with collaboration management systems, which is based on issues, merge requests and milestones has a focus on reviews of conducted work and of writing down what was done and why, which is the same information recorded as in design rationale systems. The difference between design rationale systems and collaboration management systems lie on the one hand in the reason why the information is recorded: For collaboration management systems, recording what was done is simply part of the daily work processes and necessary to complete any given task. For design rationale systems recording the rationale of a decision is an extra work step in a separate tool. On the other hand the difference lies in the way the information is structured: while a design rationale system structures the information according to the structure of the system under development, the design rationales in a collaboration management system can be found in a commit-message, a comment on an issue, a response in a merge request etc. Therefore, while showing a

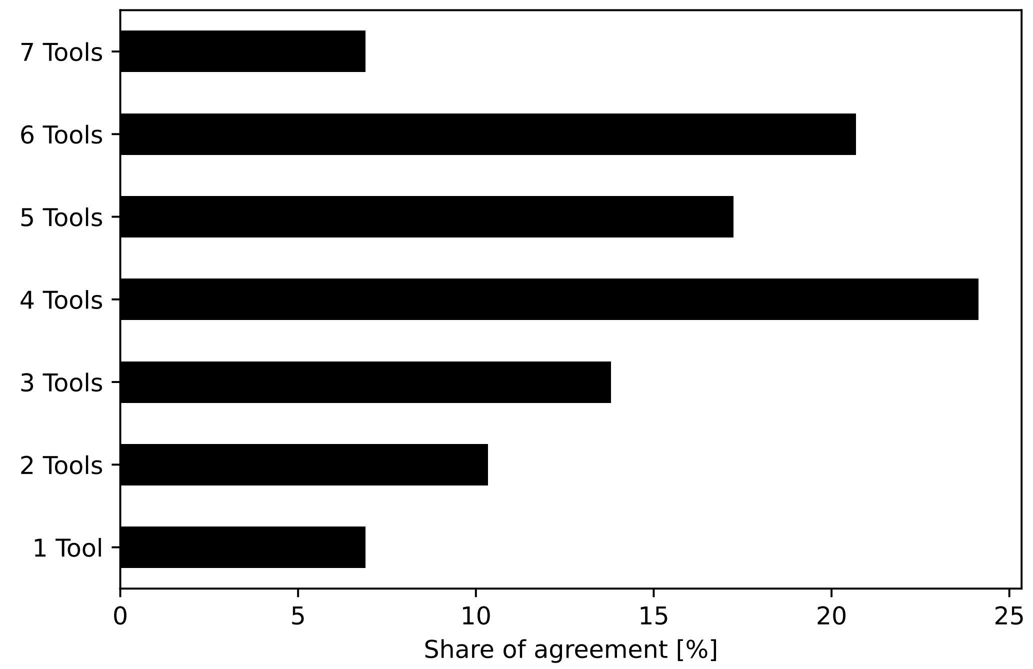


Figure 5-14.: Amount of information management tools in use in small spacecraft development projects

higher degree of completeness compared to design rationale systems and as can be seen in Figure 5-15 are far more successful, the information recorded in collaboration management systems is more difficult to retrieve.

Understanding which information management systems are in use does not yet allow drawing conclusions as to the impact they have on a team's work. As other communication paths are available, such as meetings, emails and direct verbal communication, an assessment of their respective share in the developers' communication was in order. The participants were asked to order the communication paths displayed in Figure 5-16 according to the popularity in their team. Afterwards points were awarded depending on the rank provided for a communication technology, following a linear regression (6 points for the first place, 5 for the second etc.). Figure 5-16 shows that meetings prevail as a dominant communication path, followed by direct communication. Third on the list are emails, closely followed by communicators. Considering that only 17 participants are using communicators, whereas every team employs emails, meetings and conversations, this is an interesting result and clearly shows that communicators are a priority for a holistic approach to information management. Collaboration management systems or issue trackers land only the second to last place, although 26 out of 29 participants stated that such systems are in use in their project. Thereby although widely accepted, their share in daily communications is rather low.

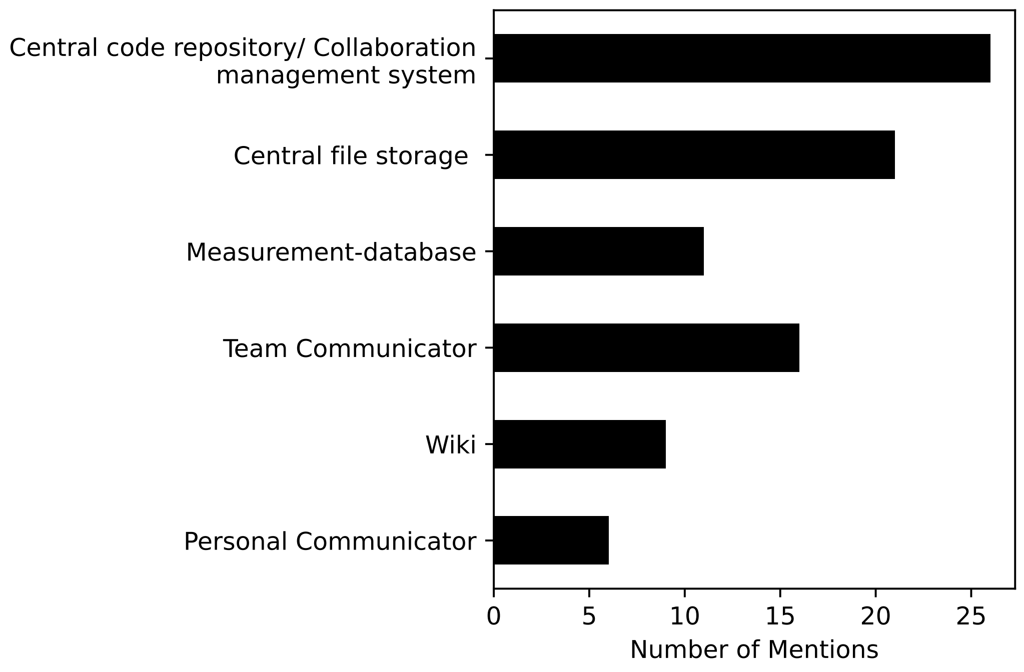


Figure 5-15.: Popularity of information management tools in small spacecraft development projects

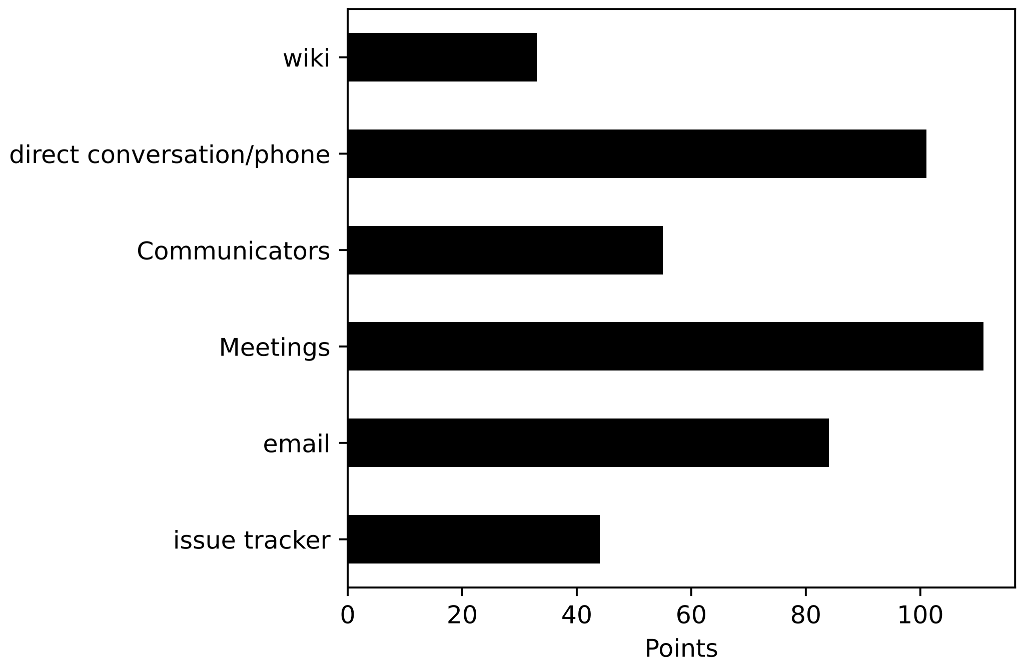


Figure 5-16.: Popularity of communication paths in small spacecraft developments

5.3.5. Objective 4: The Vicious Cycle of Project Delays

Assessing the vicious cycle of project delays' validity requires to first assess which projects are behind schedule. A unique characteristic of small spacecraft projects is the inevitability of the launch deadline. Small spacecraft are – except if being launched with a micro-launcher – part of so-called ride-share agreements. In a ride-share agreement multiple spacecraft are launched aboard the same rocket. Therefore, a delay of the launch due to a single spacecraft being delayed is usually not an option. This leads to implications on the severity of project delays: In case a spacecraft's launch is already scheduled, a delay of only a few weeks can be severe for the overall project. A delay of several months in earlier project phases may have little impact on the success of the mission, especially during phases where the costs for personnel are still rather low such as Phase 0 and Phase A, as launches can be booked with only a few months lead time. Therefore, the question regarding the current project delay was phrased relative to the point of time the spacecraft has to be handed in for integration with the rocket, as this is the last time the developers team can expect to have access to the spacecraft on ground. Figure 5-7 shows the results to the question, with 5 out of 29 participants reporting their project to be on time, a third of participants reporting major delays of over 25% and the remaining participants estimating their project delay to less than 25% but not on time.

For the following assessment, the participants were split in three groups, according to their amount of delay. Group 1 are those developers whose projects are on schedule. Group 2 are the developers with minor delays, i.e. less than 25%. Group 3 are the developers who report major delays.

The theory of the vicious cycle of project delays implies that more direct communication takes place in projects with considerable delay, as the project's documentation is being neglected and does not yield sufficient and trustworthy information.

No correlations could be found between the amount of exchange happening digitally and the project delay. Also, no correlation between the amount of information management systems and the project delay could be found. The only parameter that showed a correlation to the project's delay is the current state of documentation. With a rising delay, a tendency to more outdated documentation can be seen in the data. Figure 5-17 depicts the correlation. Due to the low amount of participants in the survey only the qualitative assessment "the higher the project's delay, the larger the share of outdated documentation" is sustained by the data. This is in accordance with the findings of Grudin and Poole on the deferment of documentation as response to pressure on deadlines and staff shortages [65].

While this partially supports the theory of the vicious cycle of project delays, other data gained in the survey does not support the theory. No difference in the time spent writing documentation can be spotted between projects on schedule and projects with any kind of delay.

Overall, the survey data neither proves nor disproves the vicious cycle of project delays. More data would have to be gathered, especially due to the fact that the already small sample size had to be split in three groups for the analysis again, which further reduced the force of expression of the study in regard to the vicious cycle of project delays.

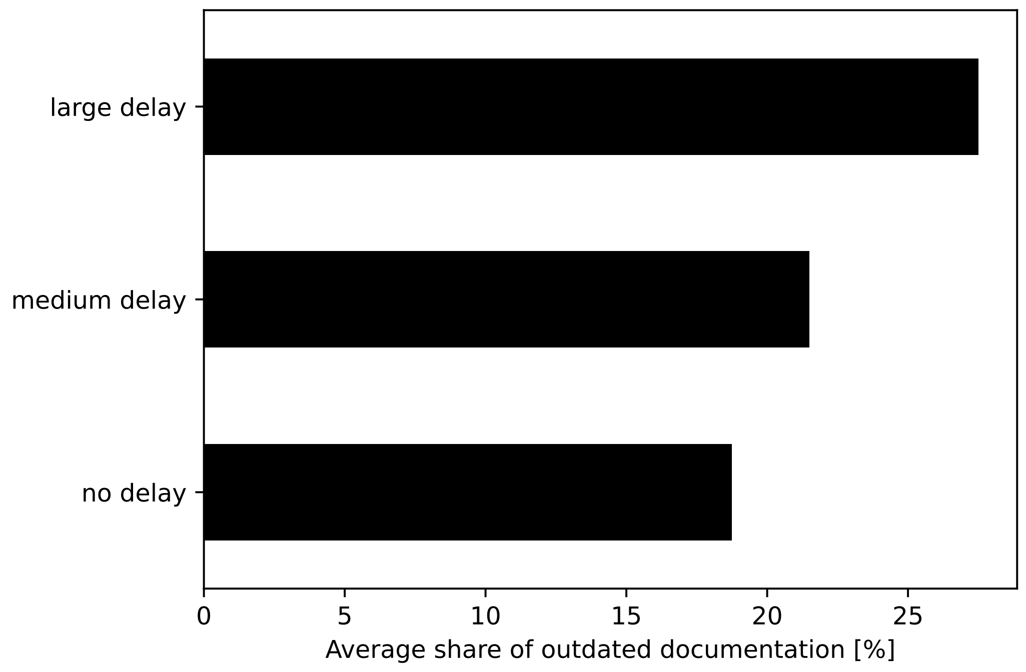


Figure 5-17.: Project delay vs. share of outdated documentation

5.4. Summary

The survey provides valuable insights into the informational processes of small spacecraft developments. It could be shown, that the amount of time spent acquiring information is lower in average in small spacecraft developments compared to the data of Berquand et al. [8]. It could further be shown, that all small spacecraft development projects consider some share of their documentation to be outdated. While the vicious cycle of project delays could neither be proven nor disproven by the survey, a correlation between project delay and the share of documentation which is outdated could be established. Most importantly for the further design of a tool to integrate information across all information management systems is the collection of information management systems used in the projects as well as their relative share in the teams' communications.

6. Proposal of a Graph Schema for SysML

Major sections of the following text are already published by the author of this thesis in [22]. This encompasses the graph schema for structural diagrams as well as activity diagrams and state machines. The aspects of the schema relating to use cases, requirements and their traceability throughout the model have not been published before.

This section provides an overview of the development of the graph schema and modeling guidelines. In order to analyze a system by combining SysML and graph analysis, a graph schema has to be defined, with the help of which the SysML model can be transferred from its editing tool into a labeled property graph. A graph schema explains how nodes in a graph can be labeled and which relation types can exist between specific nodes.

6.1. Existing Graph Schemata for SysML

Two schemata are mentioned in the literature, which – before progressing to the questions set on SysML models for graph analysis – shall be discussed briefly.

[135] proposes a schema focusing on requirements analysis. Goal of the schema is to assess completeness, consistency and correctness of requirements in a SysML model built in MagicDraw. The author provides a detailed schema for the requirements related elements blocks, test cases and requirements. The model falls short of considering any other element of the System Modeling Language. Also, the questions on which the schema builds are kept quite simplistic and do not exploit the strengths of a graph database. The analyses are centered on:

- What percentage of requirements are not completely defined?
- What percentage of requirements are not satisfied or not verified?
- How many elements with duplicate names exist?

None of these questions requires a graph traversal of more than one relation, i.e. they could be answered just as well by reading the elements into a table-based database or by employing table methods provided by MagicDraw. [135] further brings up the idea of applying graph algorithms to the model, which would allow finding critical elements in the system, or elements that have the largest influence on others and consequently applies the betweenness centrality algorithm on the requirements of the system, to find out which requirements have the largest influence on others. While this yields information for the first time that could not be obtained as easily by other means, he falls short of defining what routes and elements should be selected for the algorithm, which makes a major difference in the results.

The second graph schema found in the literature is maintained by the company Intercax as part of their Syndeia software suite (see [157]). First proposed by Bajaj in [132], the idea behind Syndeia is to generate a "Total System Model", which they specify as a model that allows to link information across various information sources in a graph database. The MBSE model is one of the information sources integrated into the total system model, alongside product life cycle management, Computer Aided Design systems, Databases such as MySQL or Neo4j, simulation tools or application life cycle management systems such as Jira or git ([157]). The software is commercially distributed and was for example applied on the Large Synoptic Survey Telescope for Verification and Validation purposes ([158]). Multiple publications were made on the software, its applications and involved challenges (see [133], [134], [159]). However, no detailed specification of the graph model itself or data of the use case systems is provided in any of the publications, making it difficult to reproduce any of the results described in the publications. In [134] they show detailed graph query results, but fall short of providing the schema that allows to query the system. While the lack of a detailed specification may be attributed to the commercial distribution of the software, a second lack in the publications on Syndeia is a SysML modeling strategy to go with the graph schema. Analogous to Petnga's work in [135] the queries presented by Bajaj [134] on the SysML graph are of a simplistic nature and do not exploit the potential of the graph database. The queries range from *show me any element connected to Element X* to *show all behaviors of Element X* and *get all paths between two nodes*. The last query is the only one in the publication that exploits the potential of a graph database compared to relational databases. Combining a graph schema with an appropriate modeling strategy allows for far deeper analyses and a higher quality of the information drawn from the model as is shown in the following.

6.2. Questions and Analyses to be Enabled by the Schema

According to [160] the aim of a graph schema should be to enable answering all questions that can be foreseen to be put to the graph with a minimum of required syntax. Therefore, this section defines questions on specific aspects of SysML models, such as structural or behavioral information. The questions provided in the following are the result of five years of personal experience in systems engineering for small spacecraft and testing and operations of small spacecraft and further detail the knowledge intensive interdisciplinary tasks in later project phases described in Section 2.3. The list is of course not exhaustive, but sufficient to help drafting the graph schema and modeling guidelines.

6.2.1. Analyzing the Structural Part of a SysML Model

SysML diagrams can be separated into three groups; structural diagrams, behavioral diagrams and requirements modeling. The following questions can be set to the structural part (i.e. Block Definition Diagrams, Internal Block Diagrams and Parameter Diagrams) of a SysML model:

1. What is component X composed of?
2. What types of ports are used over a certain range of equipment?
3. What elements belong to a certain class?
4. What systems employ a certain type of component?
5. What component supplies system X with power?
6. How is information Y processed within a certain subsystem?
7. How is information Y processed globally?
8. Which components draw power from a certain supply component?

The following questions show the might of a graph analysis, as a complete fallout analysis can be performed with the same type of queries:

9. What is the source of telemetry Y and what could influence its measurement?

10. Given an anomaly on a specific subset of a system's telemetry, which components are most likely to have caused it? Which components can be ruled out?
11. Given a failure of component X, are there any alternative ways of acquiring data usually processed by component X?
12. What happens if component X breaks?
    a) Which systems will not work nominally anymore as they process data coming from component X?
    b) Which components will be offline in case of an electrical short in component X?
    c) Which components will suffer from a loss of input, as they depend on data processed by any of the components offline due to the electrical short in component X?

6.2.2. Analyzing the Behavioral Part of a SysML Model

Behavioral Diagrams such as state machines, activity diagrams and sequence diagrams describe the response of a system to certain events and conditions. They are used for modeling expected behavior, required inputs and task sequences to reach certain states of the system or to produce a certain output, compare [17]. Analogous to this are the questions that can be answered by analyzing behavioral diagrams:

13. Which conditions lead to state A?
14. What is the shortest path from state A to state B?
    a) While condition C cannot be met?
    b) Without changing the state of equipment D?
    c) Which conditions have to be fulfilled for this path?
    d) Which activities are performed along this path?
15. Which functions/activities require object E?
16. Starting at activity F, is there a way to reach activity G? Which activities G are on that route and which alternatives could be taken?
17. Which is the shortest route from activity F to activity G?
18. Which conditions have to be met on the route from activity F to G and which inputs provided?
19. Which activities lead to the production of object E?
20. Which inputs are required to produce object E?
21. In case condition C cannot be met, which states of the system cannot be reached anymore?
22. In case condition C cannot be met, which outputs cannot be acquired anymore?

6.2.3. Requirement and Use Case Diagrams

Requirements build up the specification of the system's expected performance and are connected to blocks and activities that have to fulfill these requirements. They can usually be traced back to use cases relating to specific stakeholders of the project, such as customers, legal entities or operators via *trace*-relations. Another source of requirements are legal regulations the system has to comply with. These can also be connected to the specific requirement employing a *trace*-relation. The SysML Specification further allows for requirements to be connected to test cases that verify the requirement [161]. The SysML Specification also declares relations between requirements, which structure them and depict respective dependencies, such as containment or derive relations.

Closely related to the definition of requirements are use case diagrams. They are used to define how external entities (e.g. actors) may use the system to accomplish their system-related goals and therefore are often the origin of operational and performance requirements[17], [161]. Apart from the use cases themselves, the main element of use case diagrams are the actors. Actors are connected to use cases via *communication* relations. Use cases are related among each other via *include*, *extend* and *generalization* relations [161].

Requirements are the only part of the SysML specification where the literature already proposes a graph schema. In [135], Petnga analyzes the requirement fulfillment of a publicly available SysML model by transforming the SysML model into a graph model and consequently querying the graph. Petnga aims to provide a methodology for quality assessment of SysML based requirements via graph technology, specifically aiming at the traceability and completeness of requirements. He provides three categories of defects to be investigated; Incompleteness, Inconsistency and Incorrectness and asks questions related to the requirements such as[135]:

- Are there any two requirements with the same title but different IDs?
- Are there any requirements without "satisfy" or "verify" relations, i.e. which are not assigned to any part of the system or are not covered by tests
- Are there any components/functions assigned to the wrong subsystem?

The first two questions stated by Petnga do not require the transformation into a graph database, as they do not query any chains of relations between the SysML elements and could be answered by generating tables in the modeling tool as well. According to NASA's Systems Engineering Handbook, the following tasks additionally are to be handled by requirements management over the whole system life cycle [140]:

- Ensuring bidirectional traceability
- Change management w.r.t. established baselines
- "Identify, control, decompose and allocate requirements across all levels of the Work Break Down Structure"[1]
- Assess and conduct requirements change requests
- Comparison of requirements with product verification and validation results

The following questions were derived from the above presented tasks outlined in [140]:

23. Which requirements must component X satisfy directly?
24. Which requirements must component X satisfy for being a component of system Y, or any other higher level system?
25. What components must directly or indirectly satisfy requirement A?
26. Which requirements are affected by a change of requirement A, either because they are directly related or because they apply to the same component?
27. Which tests does component X have to fulfill due to the requirements it shall satisfiy?
28. Which test documents exist relating to a certain requirement?
29. Which stakeholders have to be included in a specific requirements change process?
30. Who has to be notified about a requirements change?

[1]NASA Systems Engineering Handbook, p. 150

31. Which requirements can be traced to use case U?
32. Which requirements cannot be traced back to a use case and stakeholder?
33. Which use cases are related to a specific stakeholder?
34. If a stakeholder leaves the project, which use cases and requirements can be deactivated? Which components of the system may become obsolete?
35. If the extension point of a use case is deactivated, which requirements and components become obsolete?

6.2.4. Summary of Analysis Questions

A set of questions were defined which can be answered by traversing the graph of a SysML model. The questions range from simple statements to detailed and complex analyses that would typically require developers to read, ingest and combine tens of diagrams, and to repeat this time intensive process for any new parameter, telemetry value, state or component of the system that the question is applied to. By transforming the model into a queryable graph, a query for each type of analysis only needs to be written once and can then be reapplied for every new parameter, telemetry value state or component of the system. The definition and validation of these queries may also take some time for the more complex questions such as question 7, 12b and 12c in Section 6.2.1, question 14 and 21 in Section 6.2.2 or question 24, 28 and 26 in Section 6.2.3. However, the graph approach has the unique advantage of returning definitely correct analyses every time, without the chance of human error in the interpretation of diagrams, once a query is verified.

Certain diagram types were omitted in this section, such as parametric diagrams and package diagrams. Parametric diagrams already provide a means of calculating values over the whole system. As we may at best reach the same level of information again that is already provided by the diagram there is no need to transform them to a graph model for analysis. Packages and package diagrams are a means of hierarchically organizing a SysML model to enable user rights management, ease readability of the model and define specific system views for a variety of users. In a graph transformed model this hierarchical organization is not necessary anymore, as elements are found by queries which traverse the graph instead of manually traversing the product tree.

6.3. Requirements for the Graph Schema

The following requirements are imposed on the graph schema:

1. The graph schema shall enable the transfer of SysML bdd, ibd, req, uc, act and stm diagrams to a labelled property graph.
2. The graph schema shall enable a bidirectional transformation. I.e. once information is transferred to the labelled property graph, it shall be possible to reconstruct the corresponding SysML model elements and their relations.
3. The graph schema shall enable users to answer the questions phrased in Section 6.2 with simple Cypher queries.
4. The graph schema's vocabulary shall be kept as small as possible.
5. The graph schema's vocabulary shall preserve the declarative nature of the Cypher query language.
6. The graph schema shall enable the setup of Cypher queries with parameters that can be reused across the same SysML model and can be applied to other models that keep with the modeling guideline.

Requirement 1 leaves out package diagrams, parametric diagrams and sequence diagrams. This is in part due to the brevity, as recreating the whole SysML specification as graph schema for labelled property graphs is too large an undertaking. The selection was made in accordance with the added value. Requirement 2 is necessary to ensure the accuracy of the information displayed in the labelled property graph. If a transfer back to SysML is possible that contradicts the original model, the graph can be misunderstood.

Requirements 4 and 5 shall ensure a shallow learning curve and fast introduction for new users. The value of a system meant to increase efficiency in information retrieval would be greatly diminished if using it would require a protracted learning phase. Requirement 6 is the ultimate means of achieving higher efficiency in information retrieval: If information retrieval processes need to be developed only once and can be reapplied to any similar problem by any person with access to the model, a major reduction of redundant work becomes possible.

6.4. Layout of the Graph Model

As explained in the previous section, the graph model should be simplistic enough to keep the learning curve shallow and actually increase efficiency in model analysis. Another reason to keep the model as simple as possible is the chance of inadvertent omission of results. If a query returns a result that is too detailed, or includes a piece of information the user did not look for, the user running the analysis can still decide to exclude this information by hand, order the results based on any calculable measure of interest or apply stricter filters on the results. The other way around may prove disastrous, as one cannot include a piece of information by hand one does not know about. Another reason to stick to a simplistic schema is the intent of general applicability. A wide variety of modeling tools and modeling strategies are employed in the industry and academia and while most of these tools rely on SysML as common core, other modeling languages and methods exist and should not be neglected. While the transfer of for example the European Cooperation for Space Standardization (ECSS) standards 10-23 and 10-25 or the Capella Engineering tool shall not be discussed in detail, a simpler schema eases the definition of transfer-schemata.

The following paragraphs keep to the layout of Section 6.2 and compare different possibilities for the schema and how this affects the queries necessary to answer the questions outlined in Section 6.2. After weighing the options, a decision for a specific implementation is presented at the end of each section.

The schema builds upon the labeled property graph model, described in Section 2.2.9. Each paragraph describes the definition of node-labels and relation-types as well as properties stored on relations and nodes. The labels were chosen with a focus on readability. The idea behind this is to maintain a shallow learning curve and preserve the declarative nature of the graph query language Cypher. Representations that read like a sentence are easier to understand and remember. For example, `(Mobile Charger) -[:CLASS]-> (Power Converter)` does not make it apparent yet which is the class and which the element. `(Mobile Charger) -[:IS_OF_TYPE]-> (Power Converter)` makes it clear that the Mobile Charger is an element of the class Power Converter.

6.4.1. Graph Schema for Structural Aspects of a SysML Model

Since the SysML is a graphic modeling language, most concepts can be transferred straight-forwardly (compare [161]):

- SysML Blocks become nodes with the label `:BLOCK`.
- SysML Ports become nodes with the label `:PORT`.

- Instances of Blocks become nodes with the labels `:BLOCK` and `:INSTANCE`
- Generalizations become `:IS_OF_TYPE` relations.
- Aggregations and Compositions, i.e. the associations that structure SysML Blocks hierarchically, become `:IS_PART_OF` relations.

Additionally, the following features must be transferred to answer the questions described in Section 6.2.1, which are discussed in the following:

- Ports attributed to block instances
- Connectors
- Itemflows

One of the ideas behind the graph schema for SysML is to avoid the need to know every aspect of the system that was ever modeled but to be able to retrieve any information via queries. Since aggregations and compositions are both used to structure a set of blocks hierarchically, they both get the same relation type `:IS_PART_OF`. Details on the relation type can be retrieved via the property `type`. The `:IS_PART_OF` relation is also used to connect ports to their respective blocks.

Instantiations in SysML are not always straight forward. While every block in the diagram becomes instantiated at the moment an internal block diagram is built, the ports connected to the blocks are not instantiated. Furthermore, the block hosting the internal block diagram, which is represented by the diagram-frame is not instantiated when an internal block diagram is built.

This becomes relevant when looking at SysML Connectors. Connectors are used to depict connections between instances of blocks, over which physical or non-physical items can be exchanged, such as electrical power, data or physical momentum. A concept closely related to connectors are itemflows. An itemflow describes what items are transmitted via a certain connection. The transmitted items themselves are modeled as Blocks (see [161]). To distinguish them, they get the additional label `:FLOWITEM`. Figure 6-1 shows the block definition diagram to the illustrative example of an internal block diagram with itemflows in Figure 6-2. The diagram depicts four block instances, of which three are connected over respective ports. The connectors both carry the flowitem X. Note that the instance "second B", which is an instance of Block B does not interface any connector.

The graph schema has to depict the information in Figures 6-1 and 6-2 unambiguously. Figure 6-3 shows the representation of the block definition diagram in Figure 6-1 in the graph. Figure 6-4 shows the relation of instances and blocks. Figure 6-5 shows the graph-transformation for the internal block diagram in Figure 6-2. The design is explained in the following and weighed against alternatives.

The itemflows shall be traceable through the whole system. One possibility to translate the information into a graph is to simply create a connector from block to block and add the flowitem's name and ID as properties on the connector. However, this would lead to the following complications:

- (a) the node of the flowitem has no connection to the flow as the connector runs from source block to target block and cannot be connected to a third node,
- (b) the flowitem would not be traceable anymore, if a block containing the flowitem is used to describe the itemflow instead of the flowitem.

The next simpler solution would be to directly use the flowitem as node between the ports it flows from and to. However, this only works until a second connector carries the same flow, as it would become untraceable which relation belongs to which connection. The solution in Figure 6-5 does not have these impasses, as a connector node is created for every SysML connector. This construct is often referred to as hypernodes, which is why the node carries the label `:HYPERNODE`.

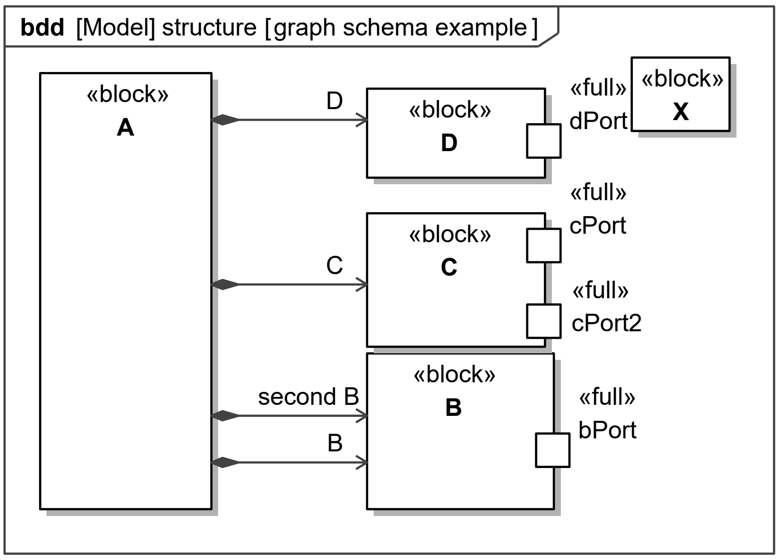


Figure 6-1.: Block Definition Diagram (bdd) to Figure 6-2

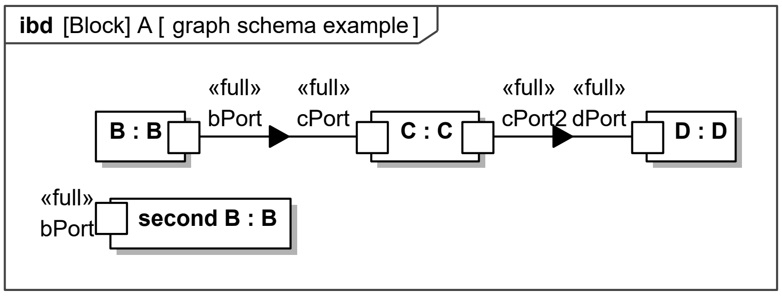


Figure 6-2.: Example of an itemflow in an internal block diagram

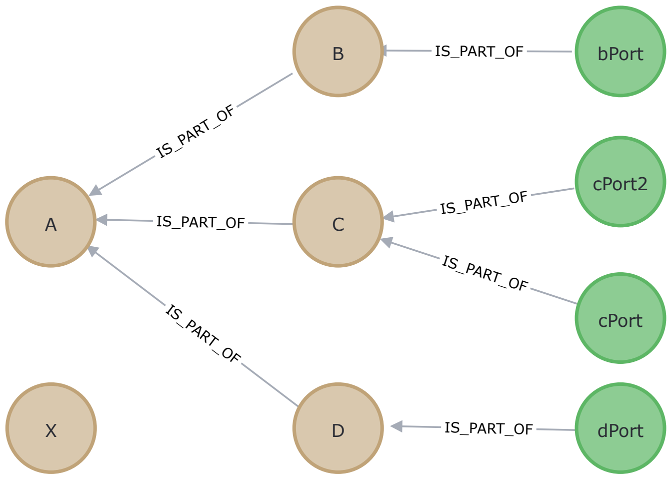


Figure 6-3.: Graph Representation to Figure 6-1. Blocks are depicted in brown, ports in green

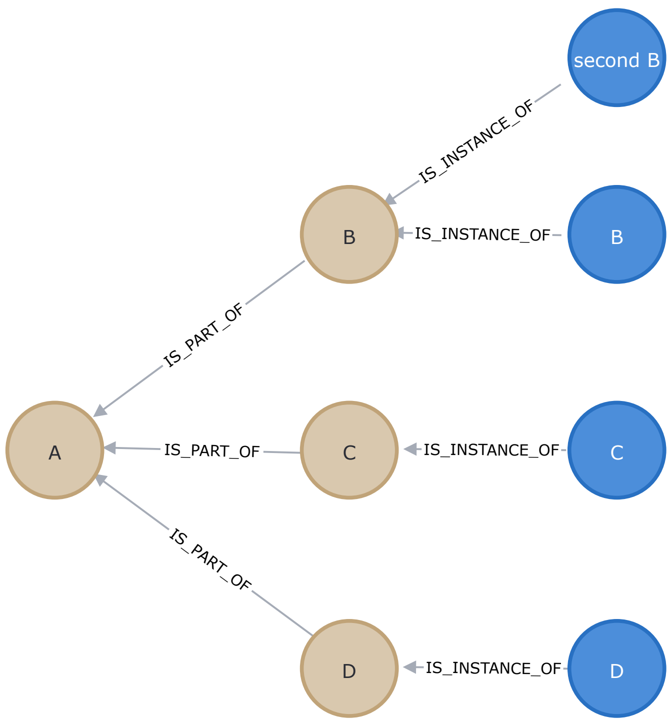


Figure 6-4.: Graph Representation of the instances for Blocks. Blocks are depicted in brown, Instances in blue

Similarly, the simplest solution to handle ports would be to omit port instantiation and only store the relation between port and block, as well as port and block-instance. In the example shown in Figures 6-2 and 6-5 however, this approach would lead to a loss of the information, whether the instance "B" or "second B" carries out the connection, as both would be connected to the port "bPort". Hence, additionally to the elements taken from the MagicDraw model, port instances are created during the translation from MagicDraw to Neo4j to discern which port instance belongs to which block instance. While ports can formally be instantiated in MagicDraw, requiring users to conduct this instantiation by hand and relating all port instances by hand would mean extra modeling effort that can be omitted. Figure 6-6 shows the complete graph schema for SysML Structure diagrams and to which the transformation of the internal block diagram in Figure 6-2 shown in Figure 6-5 is compliant. The graphic is interpreted as follows: Nodes are depicted in rectangulars, while relations are presented as connections between the rectangulars. The graphic defines which combinations of labels and relations between node types are allowed by the schema. For nodes with multiple labels (for example `:BLOCK:INSTANCE`) all relations defined by the singular labelled node (in this case `BLOCK`) are also allowed. The relation types are written on the connections. Arrows originating at the node they point to show relations defined between two nodes of the same type. Additionally to the labels and relationtypes defined in Figure 6-6, all nodes carry an `id`- and a `name` property, allowing for unique identification and easy reference. The names are taken over from the SysML model. Where no name is provided in SysML `NULL` is entered in the graph.

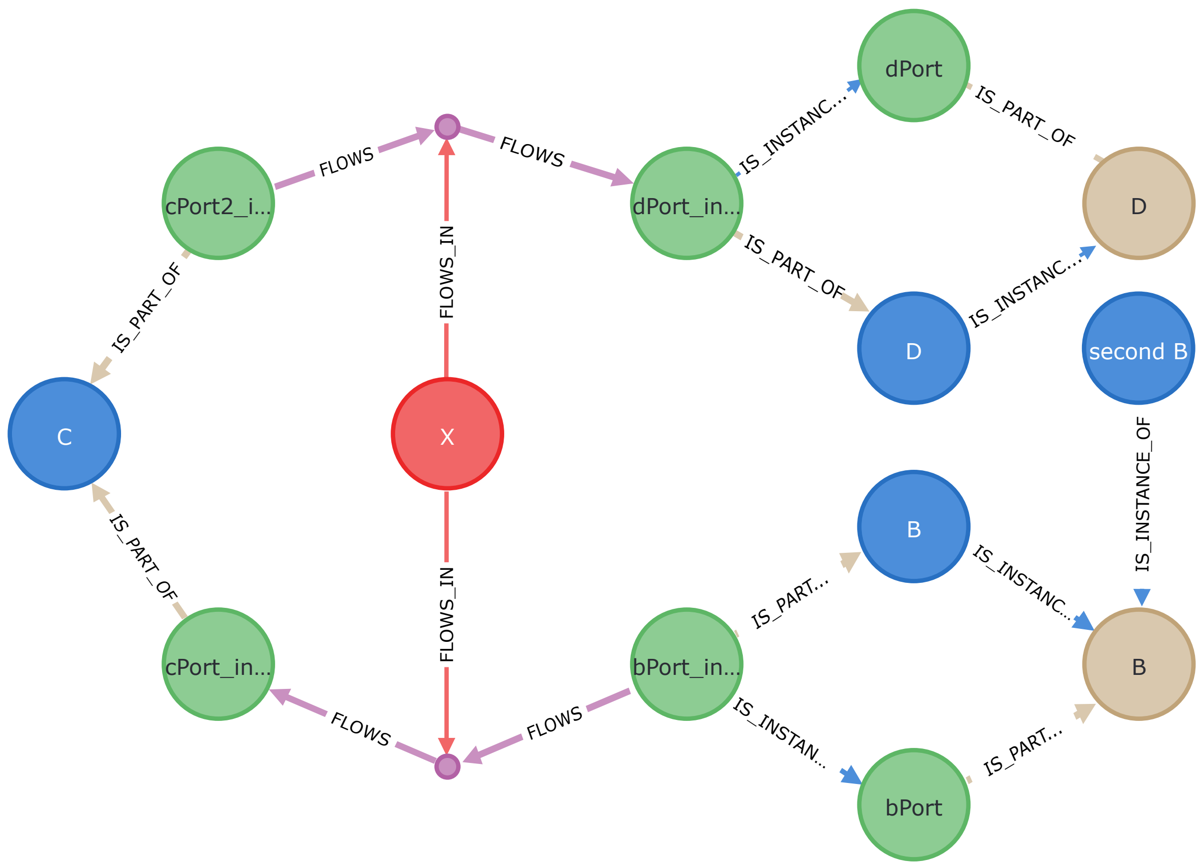


Figure 6-5.: Graph Representation of the information related to Figure 6-1 and 6-2. Blocks are depicted in brown, instances of blocks in blue, hypernodes in purple, ports and port instances in green, flowitems in red. Note: As the graph itself contains all information, this is merely an excerpt showing specific information but not bound to the limits of any SysML diagram type

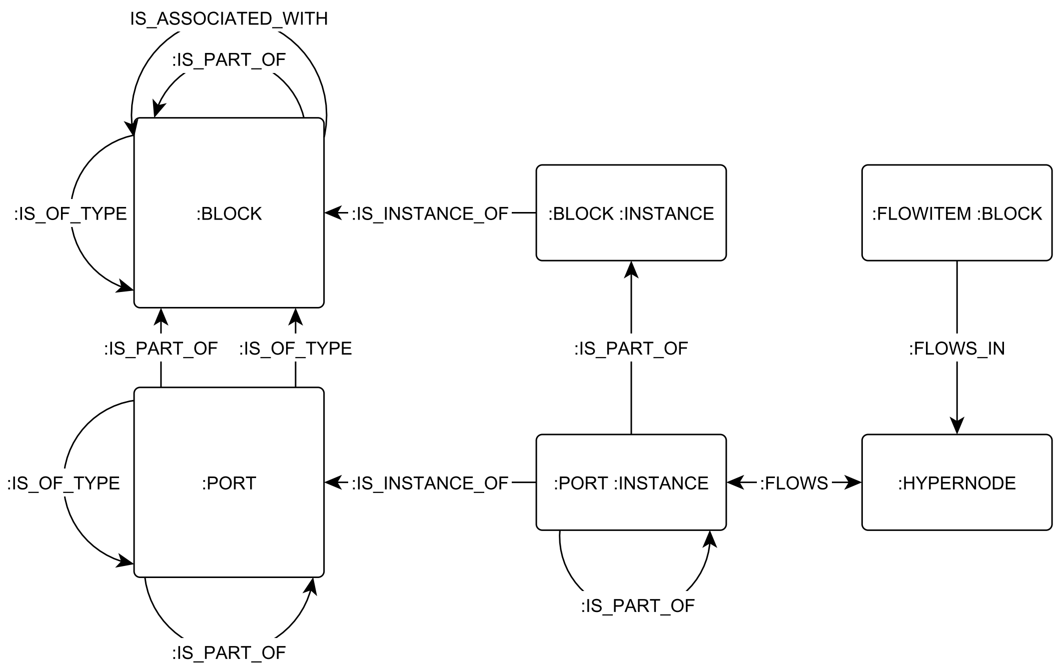


Figure 6-6.: Proposed graph schema for Structural SysML Diagrams

6.4.2. Graph Schema for Behavioral Aspects of a SysML Model

Behavior is modeled in SysML via Activity Diagrams, State Machines, Sequence Diagrams and Use Case Diagrams ([17]). This section will focus on Activity Diagrams, State Machines and Sequence Diagrams.

Graph Schema for Activity Diagrams

Figure 6-7 shows a simple activity diagram, consisting of a Start and End node and 2 activities named Action 1 and Action 2 as well as their translation into a graph.

The information contained by the control flows (dashed arrows in the SysML act diagram, green arrows in the

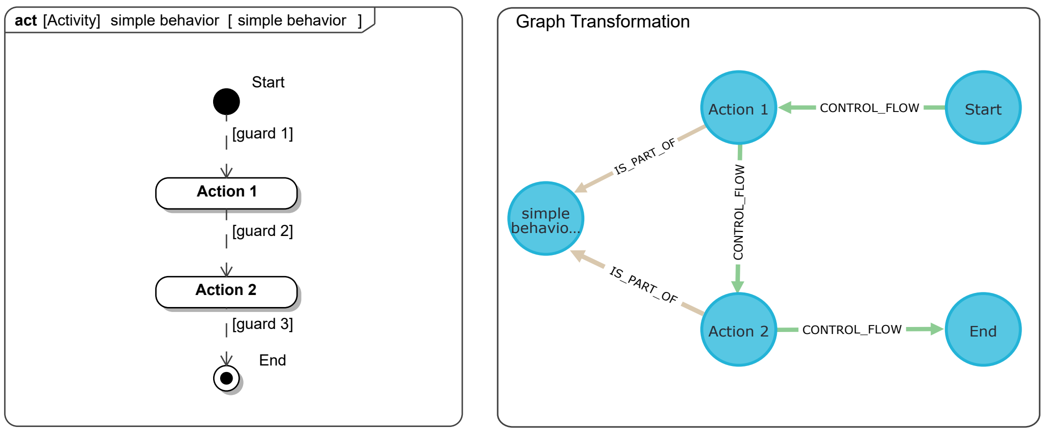


Figure 6-7.: Simple activity diagram and its graph transformation

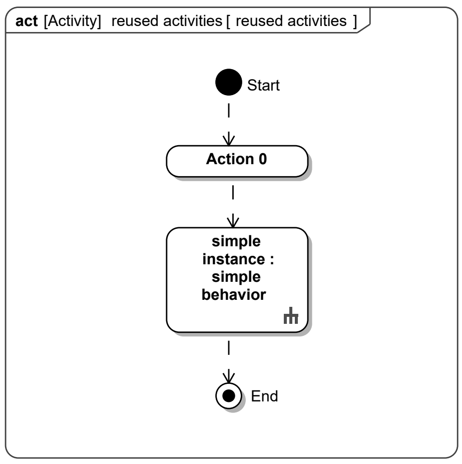

act [Activity] reused activities [reused activities]
Start
Action 0
simple instance : simple behavior
End

Figure 6-8.: Nested activities

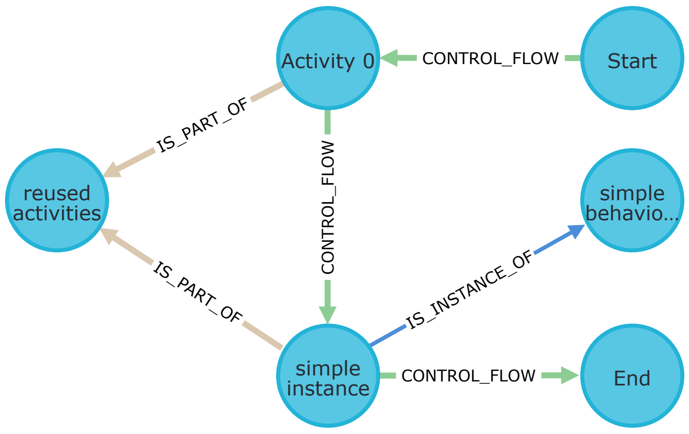


Figure 6-9.: Nested activities transformation Alternative 1

graph transformation, see Figure 6-7) is stored as properties of the `:CONTROL_FLOW` relations, which are not depicted in Figure 6-7. Any activity diagram can be seen as an activity by itself. Therefore, the node to the very left of the graph transformation in Figure 6-7 represents the diagram itself. This becomes a necessity when dealing with nested activities. Figure 6-8 shows the activity *simple behavior* from Figure 6-7 nested in another activity with the title *reused activities*. Semantically, this means *simple behavior* is instantiated in the *reused activities* diagram and the complete content of *simple behavior* is run through at the time the activity is called.

Figures 6-9 and 6-10 show different possibilities of the diagram's transformation. Figure 6-9 shows a simpler transformation that is closer to the SysML's own syntax. The *simple instance* of *simple behavior* is instantiated and connected via the corresponding control flows as depicted in Figure 6-8. Figure 6-10 shows a more complex transformation. Each activity and node contained in the *simple behavior* diagram is instantiated together with the respective control flows. The control flows that connected to *simple instance* are redirected to the *Start* and *End* node instances of *simple instance*. Thereby the complete activity flow is generated and can be queried to understand, which activities have to be performed in order to achieve a certain result.

While both transformation schemata have their advantages, we decided in favor of the variation shown in Figure 6-10, as it is easier to understand a complete flow of activities and keeps queries simpler.

Another construct of SysML Activity Diagrams is the use of blocks as inputs and outputs of activities as

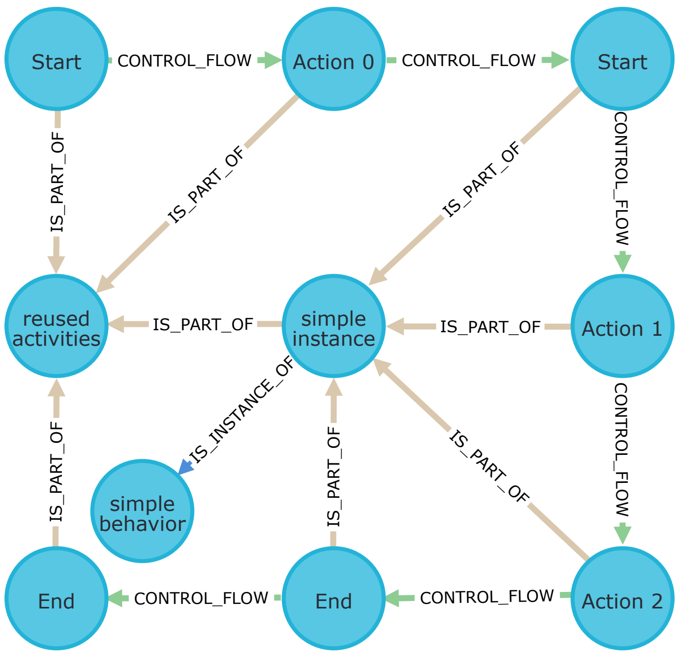


Figure 6-10.: Nested activities transformation Alternative 2. *Note:* `:IS _INSTANCE_OF` relations and respective nodes are omitted in this view for better readability

depicted in Figure 6-11. Blocks are depicted in activity diagrams as rectangles with pointed corners. They are connected to activities in the diagram via object flows (solid arrows in Figure 6-11).

Figure 6-12 shows the diagram's translation into the graph. To realize the usage of blocks in activity diagrams, two additional relation types are introduced: `:OBJECT_FLOW`, which is a direct translation of its SysML counterpart, and `:IS_USED_IN`, which represents the connection between the respective block (*A*) and the `:ACTNODE`s (*in, buffer, out*) of the object flow. The `:ACTNODE` label is also used for Initial and Final nodes, as well as Decision nodes, Fork nodes and Merge nodes. To discern the different types, the property `nodetype` is defined, which carries the respective information.

This brings us to the declaration of node labels for activity graphs. Figure 6-13 summarizes the graph schema for activity diagrams. Discerning activities from other nodes such as Initial, End, Fork or Buffer nodes in an activity diagram is necessary to answer the Questions 16 to 20 in Section 6.2.2. Therefore, additionally to the `:ACTNODE` label, an `:ACTIVITY` label is defined, which is used for the actual activities. To reduce the effort to build queries, nodes with the label `:ACTNODE` or `:ACTIVITY` also carry the label `:ACT`, which marks any node from an Activity Diagram. For any instantiation, the `:INSTANCE` label is provided, together with the relation `:IS_INSTANCE_OF`. The `:BLOCK` label used in Figure 6-13 is the same one as in the graph schema for block definition diagrams and links the two diagram types.

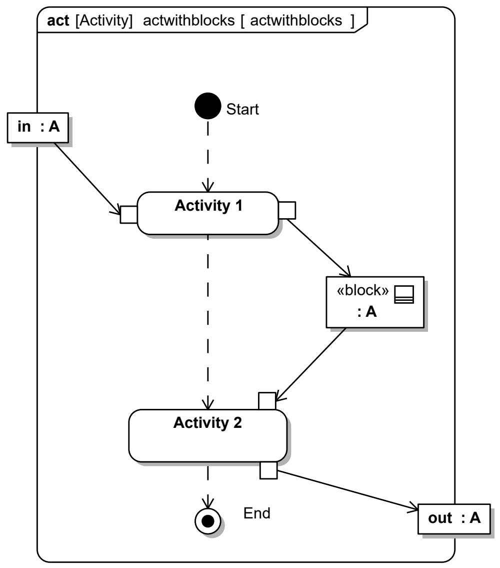


Figure 6-11.: Activity diagram with blocks

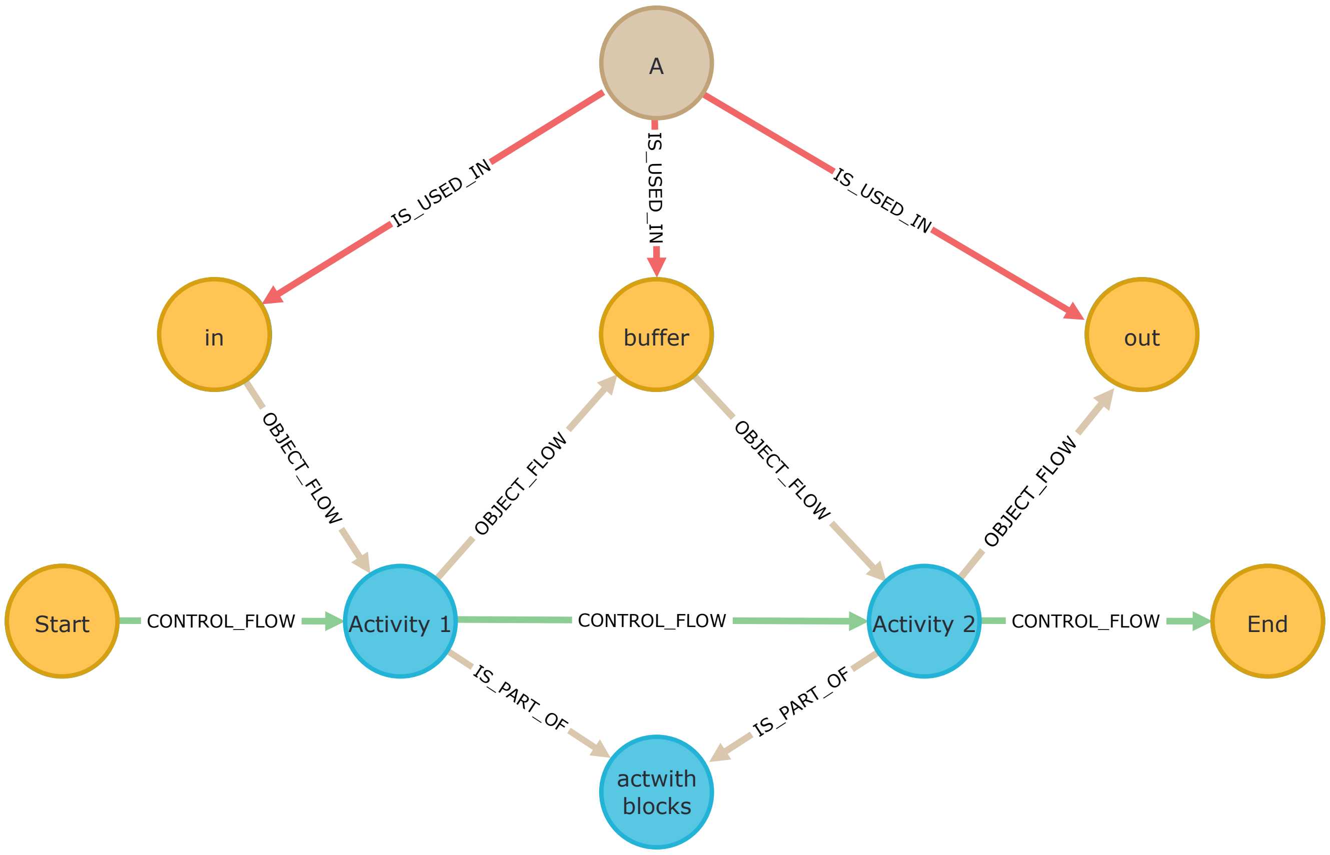


Figure 6-12.: Graph transformation of the activity diagram with blocks in Figure 6-11. Yellow: `:ACTNODE`, blue: `:ACTIVITY`, beige: `:BLOCK`

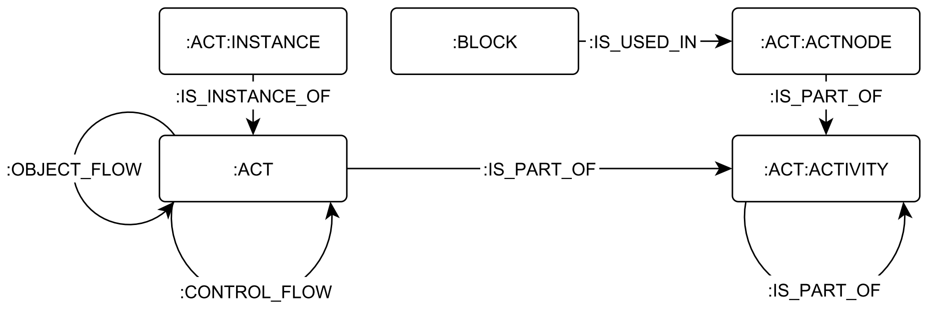


Figure 6-13.: Graph schema for activity diagrams.

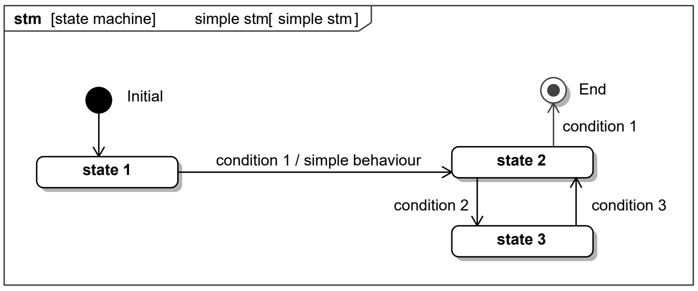

stm [state machine] simple stm[simple stm]
Initial
End
condition 1
state 1
condition 1 / simple behaviour
state 2
condition 2
condition 3
state 3

Figure 6-14.: Simple state machine

Graph Schema for State Machines

The second important part of behavioral modeling is state machines. Figure 6-14 shows a simple state machine with three states, *state 1, state 2, state 3* and a set of transitions. According to [161, p. 162], transitions can carry triggers, guards and activities. Triggers and guards specify when the transition becomes active, while the activity simply states that an activity shall be performed when the transition activates. Figure 6-15 shows a possible transformation of the state machine in Figure 6-14 into a graph. However, the schema applied in Figure 6-15 does not allow for complex queries on transition conditions and behavior such as Question 14 in Section 6.2.2. This is resolved by a `:HYPERNODE` construct, equivalent to the `:HYPERNODE` in Section 6.4.1, which is linked with the state that was left and the state which is entered via a `:TRANSITION` relation. Triggers for state transitions, such as condition 1, 2 and 3 in Figure 6-14 carry the label `:TRIGGER`, which may be added additionally to any other label the conditioning node carries. `:TRIGGERS` relations link the `:TRIGGER` with the `:HYPERNODE`.

Additionally, to the trigger events, so-called guards can be specified to refine the conditions under which a state transition is activated. The definition of trigger events as separate nodes is sufficient to answer the questions defined in Section 6.2.2 though. Therefore, guards are for now treated as properties of the `:HYPERNODE` (`guard`-property). For an exemplary transformation of Figure 6-14 see Figure 6-16. Note how *condition 1* is used in both transitions and the activity *simple behavior* is also linked to the transition, enabling us to answer questions such as Question 14a, c (What is the shortest path from state A to state B, while condition C cannot be met and which conditions have to be fulfilled for this path?), 21 (In case condition C cannot be met, which states of the system cannot be reached anymore?) or 22 (In case condition C cannot be met, which outputs and signals cannot be acquired anymore?).

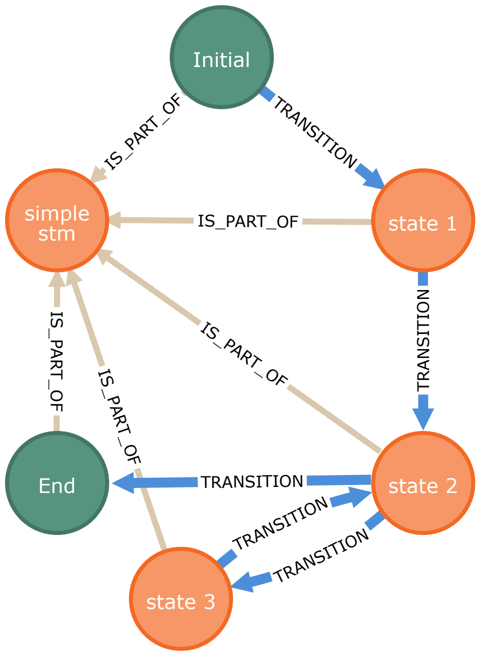


Figure 6-15.: Possible graph transformation of the state machine in Figure 6-14. `:STATE` nodes in orange, `:PSEUDOSTATE` nodes in green, `:TRANSITION` relations in blue, `:IS_PART_OF` relations in brown

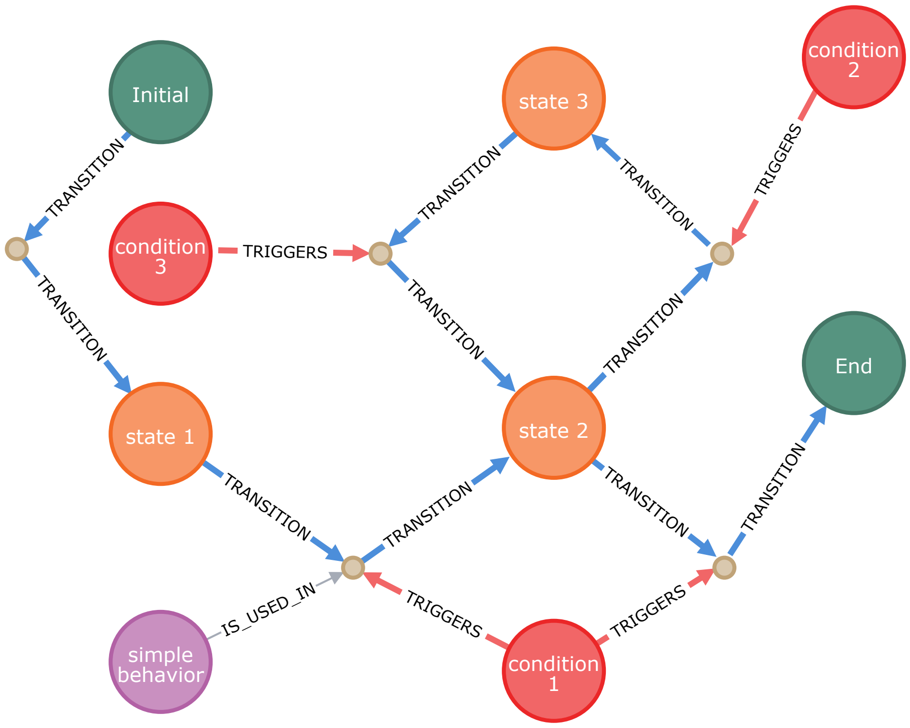


Figure 6-16.: Graph transformation of the state machine in Figure 6-14. `:STATE` nodes in orange, `:PSEUDOSTATE` nodes in green, `:HYPERNODEs` in brown, `:ACTIVITY` nodes in purple, `:TRIGGER` nodes in red. *Note:* The node "simple stm" is not shown for better readability

Figure 6-17 and Figure 6-19 show more possibilities to be considered for the transformation. Figure 6-17 depicts state machines with substates in two variations. This opens up possibilities for the flow of transitions, as the transition entering *superstate* can either be drawn to *superstate*, *start superstate* (or *entry*), or both. The same applies for the transitions to leave the superstate. Arguments can be made for and against all three variations:

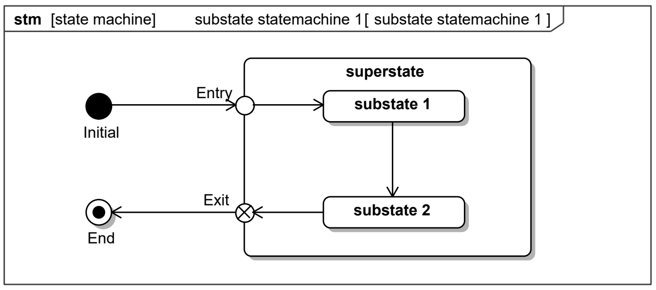
- Drawing the transition from *initial* to *superstate* enables us to answer the question *"How can I enter or leave superstate?"*, but makes it more cumbersome to follow the flow of transitions through the project, as for every state we now need to query whether it has a set of substates and if so include these in the list of states and transitions. Which level of detail is appropriate may be a difficult to answer question especially since the graph queries shall be usable without prior knowledge of the complete SysML model.
- Drawing the transition from *Initial* to *Entry* enables us to follow the transitional flow through the whole diagram, without ever querying any other relation than `:TRANSITION`. A disadvantage of this approach is that it becomes more difficult to query how to enter and leave the *superstate*.
- Drawing both transitions leaves us without any cumbersome queries for either way of modeling, but results in a graph that is harder to understand, since the transitions in the graph would show that you are either in the *superstate* or in *substate 1* or *substate 2.*

Following a transitions network through a set of state machines is a more complex task than answering the question of possible entries and exits to a single state. Therefore, I decided to stick with the second option from above. Figure 6-18 shows the graph transformation of the state machines in Figure 6-17.

An interesting feature of state machines in SysML is their connection to activities. Activities can be set to be performed on transitions, entry, do [while] or exit of a state [161, p. 161]. Such activities may for example log the information "state was left", or saving the results at the end of a measurement accumulation. Such a state machine is displayed in Figure 6-19 on the right. Employing activities is a basic feature of state machines and depicting this kind of crosscutting connection from one diagram type to the other is one of the strengths of graph databases. *State 2* in Figure 6-19 shows activities at various positions within a state (entry, do, exit). The activities used in the state machine have to be defined elsewhere and are then only used in the state. Using a set of activities within a state implicitly creates a new set of control flows between the entry, do and exit activities. The graph schema covers this implicit connection by explicitly linking the three elements with `:CONTROL_FLOW` construct (see Section 6.4.2). Another relation is necessary between the state and the activities performed within it. This is covered by a `:IS_PART_OF` relation. While it could be argued, that a new relation type instead of reusing a relation from the structural aspects of SysML makes sense, goal of the schema is to keep it as simple as possible. Creating new relation types for the same kind of information as in the structural part of the SysML is not necessary as the context provided by the node labels (`:BLOCK` in the structural part of the schema, `:ACTIVITY` and `:STATE` in the behavioral part) resolves any ambiguity.

The diagram on the left side of Figure 6-19 shows the reused state machine *reuse instance: simple stm*. The concept of nesting and reusing defined behavioral elements was already described in the beginning of this section for activities. The same logic applies here. Porting the instance of the State Machine (stm) that was already created by MagicDraw over into the graph allows for clear and understandable semantics. Therefore, instances of states are referred to with the label `:INSTANCE` additionally to the `:STATE` label. Apart from the `:STATE` nodes, other elements of the state machine also have to be transferred into the graph, such as initial nodes and end nodes, decision nodes, etc. For these elements the `:PSEUDOSTATE` label is provided.

Figure 6-20 shows the graph schema for state machines in detail. Nodes of the type `:STATE` and `:PSEUDOSTATE` can be connected to `:HYPERNODE`s via the `:TRANSITION` relation. `:STATE:INSTANCE` nodes are connected to `:STATE` nodes via the `:IS_INSTANCE_OF` relation type. States can also be part of other states, and therefore same as `:ACTIVITY` nodes can be connected to states via `:IS_PART_OF` relations. `:TRIGGER` nodes define the conditions which are necessary to conduct a transition and relate via the `:TRIGGERS` relation to `:HYPERNODE`s.


stm [state machine] substate statemachine 1[substate statemachine 1]
superstate
Entry
substate 1
Initial
Exit
substate 2
End

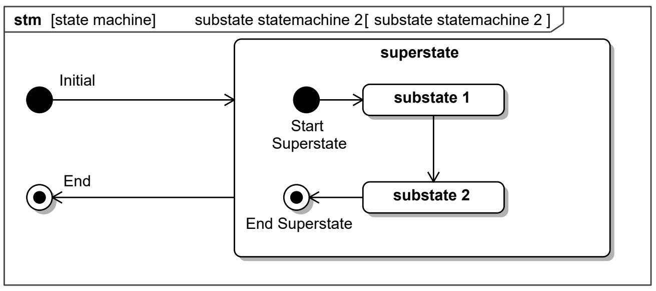

stm [state machine] substate statemachine 2[substate statemachine 2]
superstate
Initial
substate 1
Start Superstate
End
substate 2
End Superstate

Figure 6-17.: State machine with substates in two variations

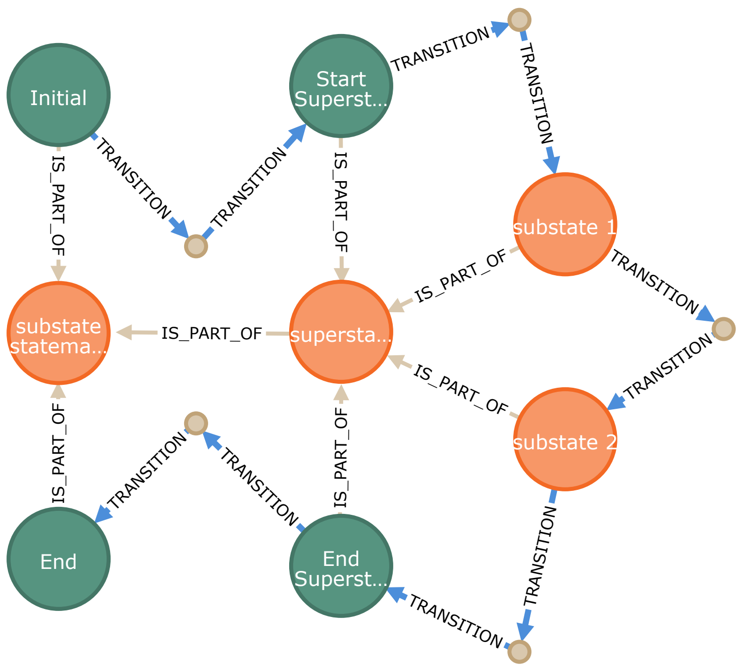


Figure 6-18.: Graph transformation of the state machine with substates in Figure 6-17. Orange nodes carry a :STATE-label, green nodes a :PSEUDOSTATE-label and beige nodes a :HYPERNODE-label. State transitions are shown as blue relations, whereas :IS_PART_OF relations are shown in beige

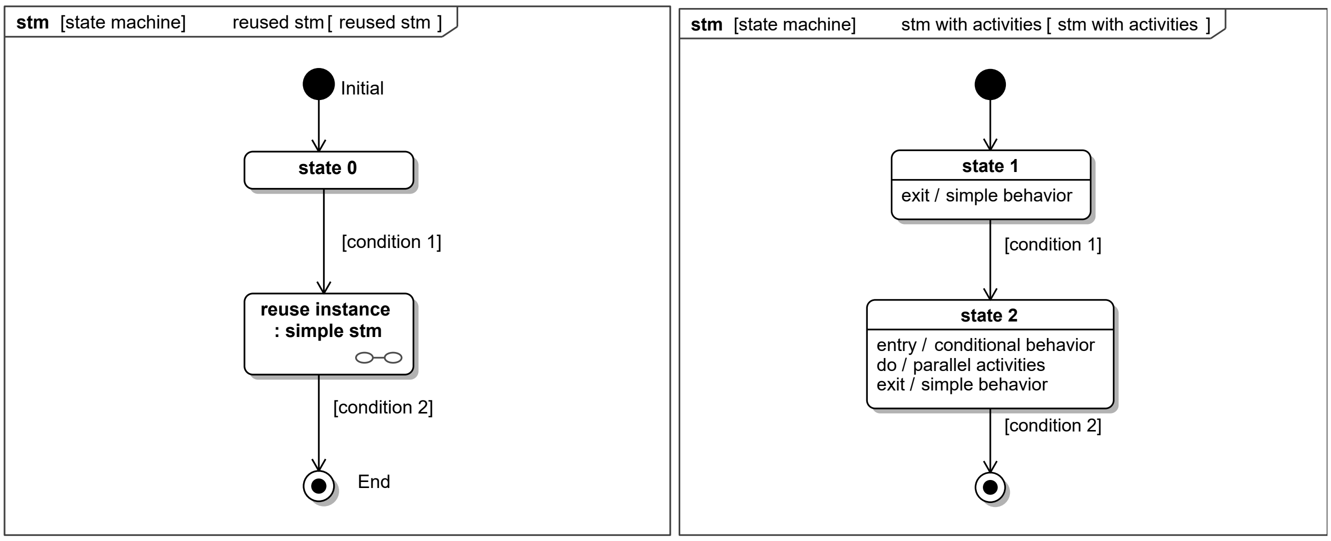

stm [state machine] reused stm [reused stm]
Initial
state 0
[condition 1]
reuse instance : simple stm
[condition 2]
End
stm [state machine] stm with activities [stm with activities]
state 1
exit / simple behavior
[condition 1]
state 2
entry / conditional behavior
do / parallel activities
exit / simple behavior
[condition 2]

Figure 6-19.: Left: State machine with reference to another stm. Right: State machine with activities

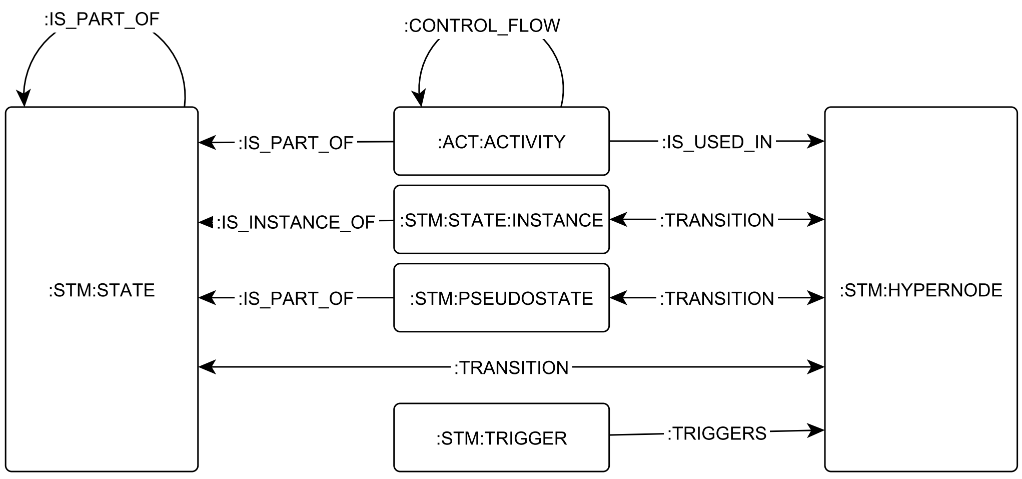


Figure 6-20.: Graph schema for SysML state machines

6.4.3. Graph Schema for Use Cases and Requirements

Although, use case diagrams are categorized as behavioral SysML diagrams, they are listed together with requirements here as the two are closely linked. A major aspect of later phases of a spacecraft development is the verification and validation of the design, i.e. is the system behaving as specified and is it behaving as intended [140], [146], while a focus point of earlier mission phases is to ensure the traceability of the design to the requirements, use cases and ultimately stakeholders involved in the project.

Being able to trace any component under test back through the whole model and being able to derive any requirements linked to the component, its subcomponents, components it interfaces or its containing system plays into the strengths of the graph transformation. The SysML allows for various possibilities how to model and especially relate requirements and use cases. Some of these possibilities are semantically similar. For example the deriveReqt-relation can be employed to describe that one requirement is derived from another. Semantically similar are the trace-relation, being a generalization of deriveReqt that can be applied between any model elements, and the refine relation, which also describes that one model element provides further details on the content of another model element.

A more general purpose relation is the allocated-relation, which according to the SysML 1.6 specification "is a precursor to a more concrete relationship between the elements, their properties, operations, attributes, or subclasses" [161, p.174]. From a graph perspective it is desirable to limit the amount of parallel constructs that describe the same circumstances to a minimum. The idea of a graph transformation is to allow users with minimal knowledge of the SysML model to query the graph. Since allocate, deriveReqt, trace and refine all express the concept of one model element providing further details of another, they are all gathered in the same relation-type, called `:IS_TRACED_FROM`. Otherwise, querying the relations between requirements and use cases becomes unwieldy, with defining any of the four relation types above for any relation between the requirements and use cases just to make sure no relation is unintentionally omitted. Figure 6-21 provides a visual representation of the problem. Acknowledging the slight differences between the types, the property `sysmltype` is set on the `:IS_TRACED_FROM`-relation. This ensures that no information is lost in the transformation while maintaining a lean graph schema. Figure 6-22 shows the graph transformation of the model elements in Figure 6-21.

For the sake of consistency, the communication-relations between use cases and actors, as well as the copy-relation are also modeled as `:IS_TRACED_FROM` and given the properties 'communication' and 'copy' respectively [161].

Other relations of the requirements diagram should not be treated this way. The relations of requirements to blocks and activities are semantically distinct from the relations described above and each other [161]. Therefore, the Satisfy relation as well as the Verify relation are taken over in the graph schema as `:SATISFIES` and `:VERIFIES`. Figure 6-23 shows the respective SysML constructs and their graph transformation.

Similar considerations apply to generalizations and the handling of extension points. In Figure 6-24 "Requirement D" can be traced back to "Actor D". In case the Generalization between "Use Case D" and "Use Case Delta" would be handled with the `:IS_OF_TYPE` relation as described in Section 6.4.1, a graph query relying on `:IS_TRACED_FROM` relations would return that "Requirement D" cannot be traced to any stakeholder and neither can the use cases "Delta" and "Delta.1". On the other hand replacing the `:IS_OF_TYPE` concept here with `:IS_TRACED_FROM` would result in users not being able to query all generalizations in the graph with the same concept and thus be inconsistent. The solution favored here is creating both relations, `:IS_OF_TYPE` and `:IS_TRACED_FROM` between "Use Case D" and "Use Case Delta". To avoid similar problems with the handling of extensions, the relation `:IS_TRACED_FROM` extends is created directly between the use cases instead of between the use cases and the extension point. The `:EXTENSIONPOINT` node is then merely connected to its use case via a `:IS_PART_OF` relation and a `:REQUIRES` relation between the extending use case and the `:EXTENSIONPOINT` node.

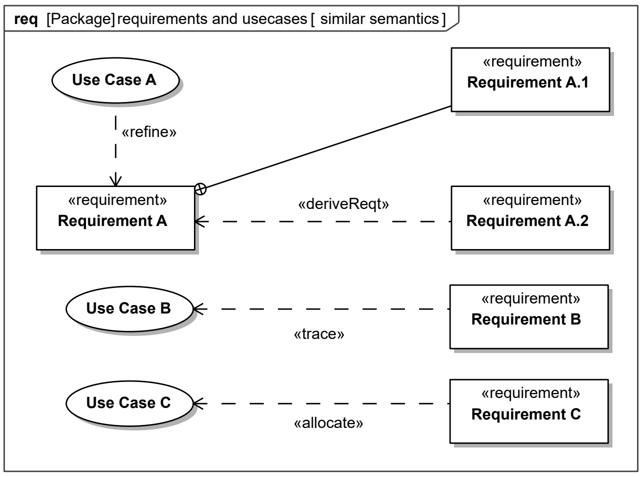

req [Package] requirements and usecases [similar semantics]
Use Case A
«refine»
«requirement» Requirement A.1
«requirement» Requirement A
«deriveReqt»
«requirement» Requirement A.2
Use Case B
«trace»
«requirement» Requirement B
Use Case C
«allocate»
«requirement» Requirement C

Figure 6-21.: Requirements diagram showing multiple expressions with similar semantics

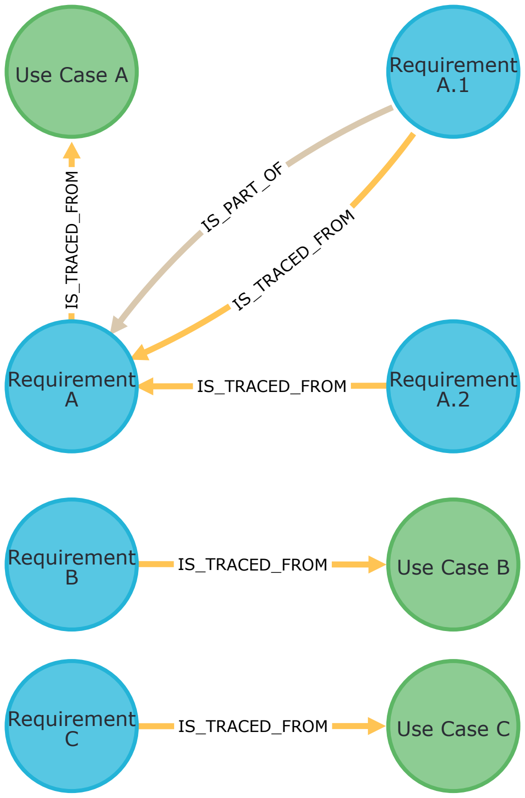


Figure 6-22.: Graph translation of requirements and use cases in Figure 6-21

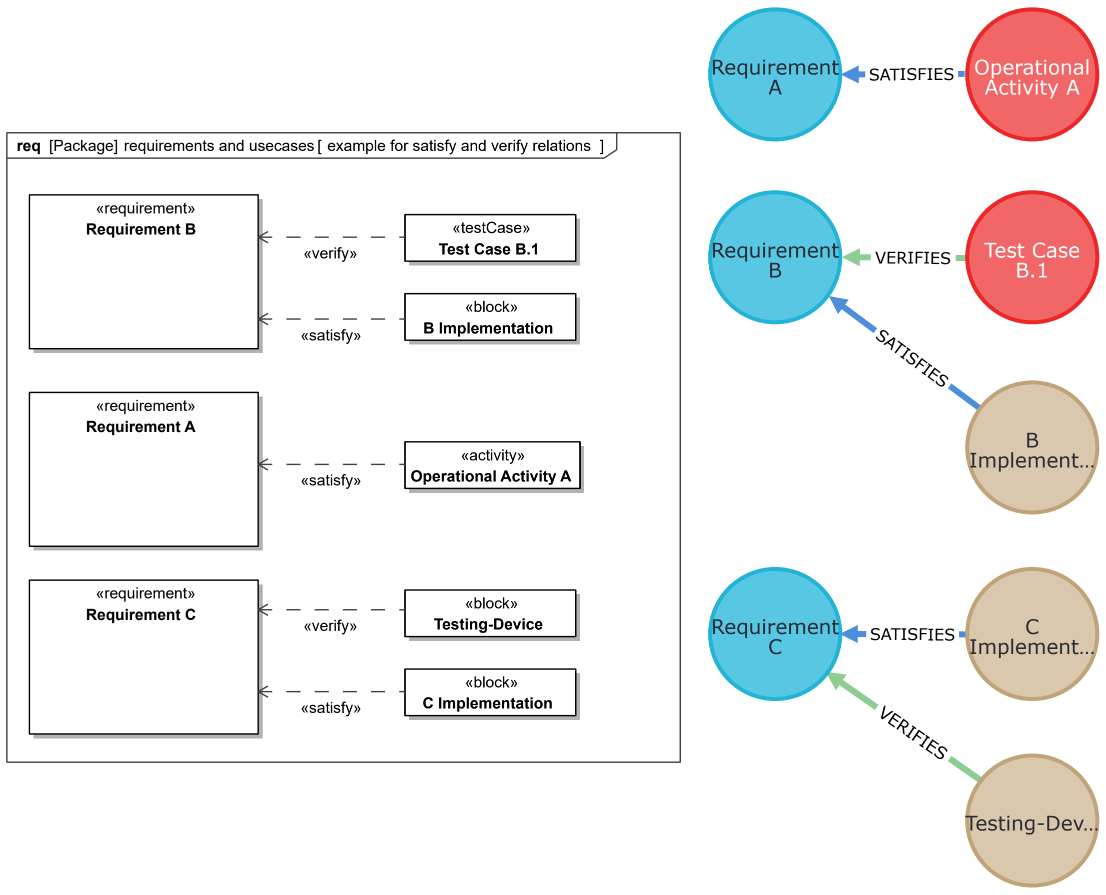


Figure 6-23.: SysML satisfy and verify relations and their translation in the graph schema. Blue: `:REQUIREMENT`-nodes, red: `:ACTIVITY`-nodes, beige: `:BLOCK`-nodes

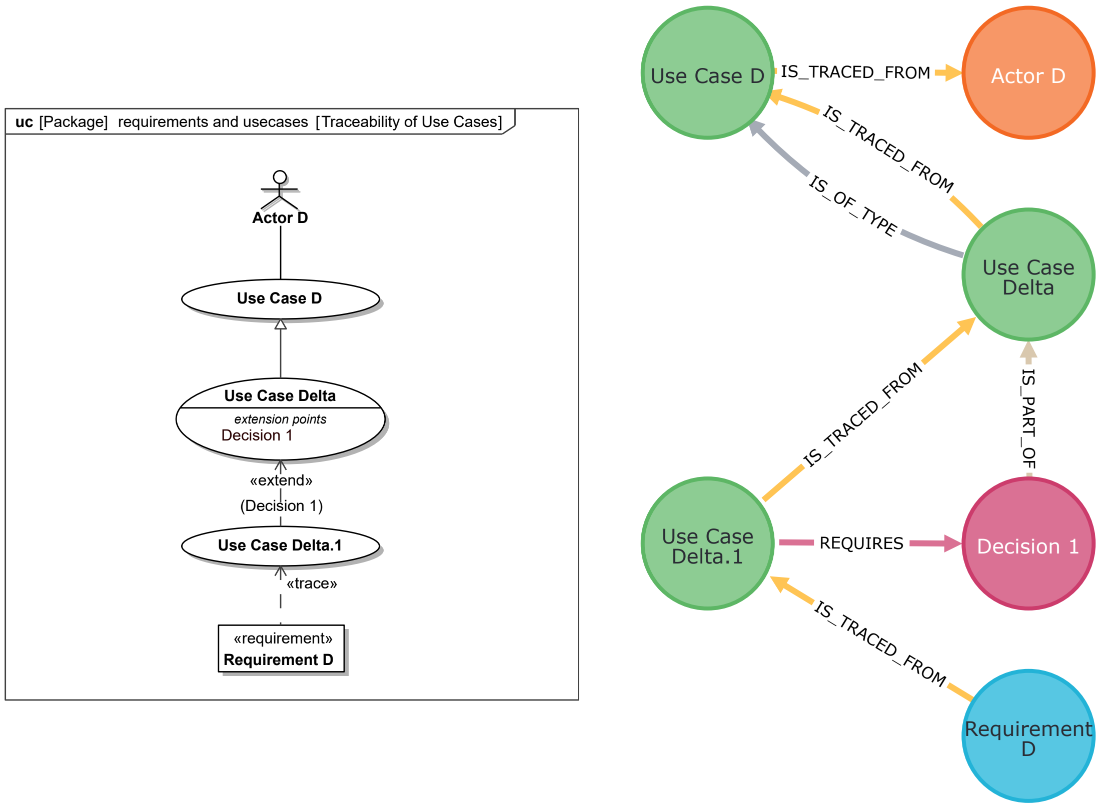


Figure 6-24.: Requirement traceability showcase. The requirement can be traced back to the stakeholder by querying a single relation type

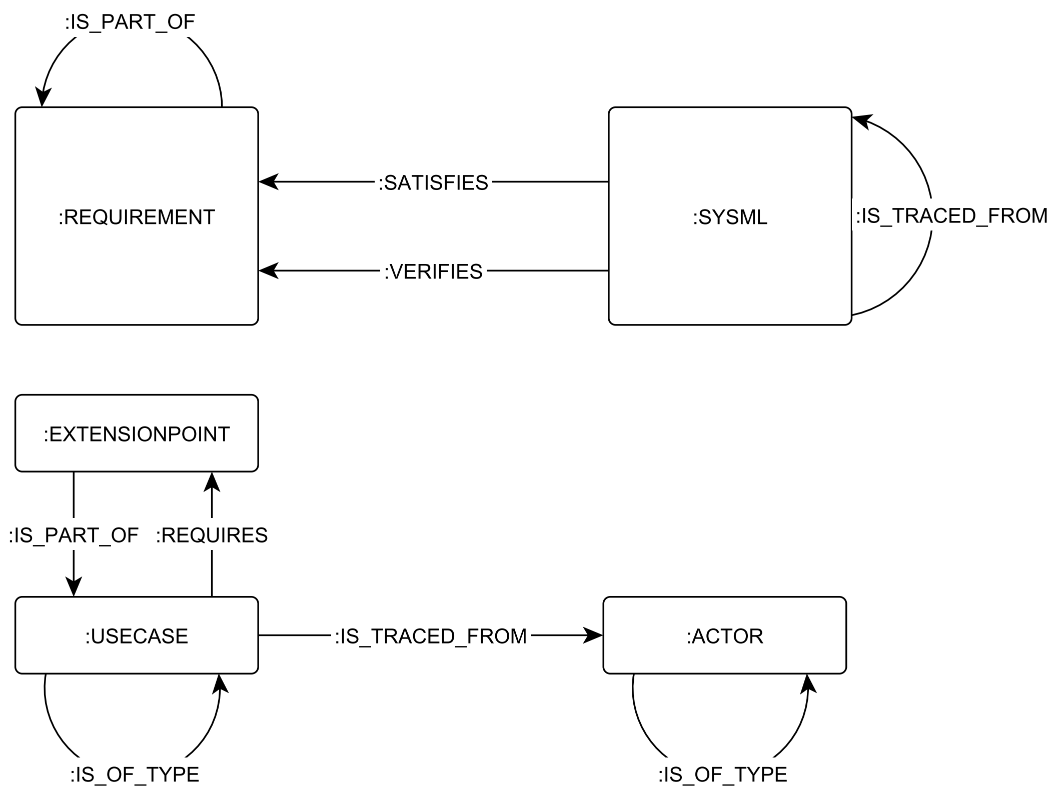


Figure 6-25.: Graph schema for requirement and use case diagrams. Note: Since all nodes created from the SysML model carry the `:SYSML`-label, all relations of the `:SYSML`-node can be realized on any node from the SysML model, including requirements, actors and use cases

Figure 6-25 provides the graph schema for requirement and use case diagrams.

6.4.4. Summary

In the previous sections, a graph schema was laid out to transfer SysML models to a labeled property graph. The information modeled in block definition diagrams, internal block diagrams, requirements diagrams, use case diagrams, activity diagrams and state machines can be translated into a graph with the help of this graph schema. Thereby information spanning the model can be retrieved in a single query without ever having to read or know the diagrams of the SysML model.. The schema is free of a loss of information for any information that is being transferred. I.e. the information in the SysML model – albeit not the diagrams – can be reconstructed from the graph. It has to be noted though, that not all information that can be put in a model transferred by the graph schema. Among others, the following diagrams and concepts are not considered in the graph schema and hence also not queryable in the graph:

- Package Diagrams and the concept of hierarchic structure via packages
- View and viewpoints
- Custom Stereotypes and profiles
- Constraint Blocks
- Parametric Diagrams

- Sequence Diagrams
- The concept of object-oriented modeling.

Extending the schema to allow querying these constructs is of course possible, but out of scope for this thesis.

Being able to query a SysML model via a graph database does not by itself guarantee the queries to return sensible results. The results are still good for simple queries, such as which requirements cannot be traced to a use case. More complex questions, such as which data cannot be acquired anymore if a fuse triggers on the spacecraft, require making assumptions on how the SysML model is set up. Therefore, the following section describes the accompanying modeling strategy, which together with the graph schema enables the definition of complex queries.

6.5. Modeling Guidelines

The following sections provide modeling guidelines for the creation of the SysML models. Goal of the guidelines is to allow powerful graph queries while setting only a minimum of requirements to attain these results. The modeling guidelines presented here shall have minimal interference with any project specific modeling strategy employed in a project. Additionally, the overhead due to the modeling guidelines shall be kept at a minimum. Therefore, the outcome of omitting those rules that create a major overhead or which may not even be fulfillable due to missing information is discussed if appropriate. Parts of the following sections concerning structural and behavioral aspects of SysML are already published by the author in Ref [22], although more examples and additional explanations are provided here.

6.5.1. Definition of Model Scope

The first questions to answer when building a new model is the question of the model's intended scope. This includes the following considerations:

- Defining the purpose of the model
- Defining which systems/subsystems shall be modeled
- Defining the targeted modeling depth
- Ensuring the model's maintainability

Section 6.2 details analyses that can be conducted with the graph schema, provided that the model carries the necessary information. Hence, the model's scope should be tailored to the specific needs. For the three showcases in the following chapters the purposes of the model differed. Therefore, the MOVE-II model's scope encompasses the spacecraft, ground station and operations infrastructure and models all data flows from each sensor down to the telemetry database, as well as all power flows in the spacecraft. As its main application is to conduct fall-out analyses, no modeling of the spacecraft's behavior took place as well as no modeling of the spacecraft's requirements. The analyses on the Extendable Modular Power System (EMPS) model are focused on the connection of sensor-measurements to their physical origins. Hence, here as well as in the MOVE-II model, no requirements or behavior are modeled. The analyses on the Environmental Control and Life Support System (ECLSS) are centered on the derivation of system configurations from various use cases and showing how advantages and disadvantages of these configurations can be pulled from the model's graph with a focus on redundancy of life support systems. Therefore, the model contains in-depth information on use cases and requirements as well as their traceability to each component.

A system can only be modeled to the degree to which it is defined, which limits the depth in earlier mission phases. In later phases considerations of intellectual property often prohibit an in-depth modeling of subsystems delivered by external vendors. Effort and outcome have to be weighed. On a large-scale project it may seem impractical to model a spacecraft down to the last component due to the sheer effort necessary to build up the model. The graph schema and queries accommodate for such considerations and work independently of the modeling depth. However, when working with shallow or incomplete models consider that

the graph can only relay information that is available in the model.

Last but not least, the model needs to be maintained. Once the model becomes outdated, the analyses are not trustworthy anymore. Seeing as available workforce is always a practical consideration leads to the following guideline for maintainability:

Limit the scope of the model to a depth and breadth that ensure the model stays maintainable with the available workforce.

6.5.2. Modeling Guideline for Structural Aspects of the SysML Model

The following section refers to the questions defined in Section 6.2. Any question number cited in the following refers to Section 6.2.

Modeling of Structural Associations

Recalling the questions defined in Section 6.2.1, the Questions 1 (detailing the composition of a block) to 4 (detailing systems that employ a certain component) require a sound use of associations between blocks. I.e.

Every Block that is instantiated should be part of a structure of SharedAssociations and PartAssociations which start at the system of systems and reach any components used in the system.

This is also required for Question 8 (querying components dependent on a certain power supply) or Question 12 (querying the fallout caused by a broken component). As the graph schema unifies the concepts of part associations and shared associations into `:IS_PART_OF`-relations, no distinction is made in the guideline between the two association types.

The same applies for flowitems such as data: modeling flowitems as a set of associated blocks saves modeling effort when pursuing questions such as "How is information Y processed globally?" (Question 7) or searching for likely culprits in case of an anomaly on a specific subset of telemetry (Question 10).

Generalizations

The Questions 2 (querying the types of ports used over a certain range of equipment), 3 (querying the elements belonging to a certain class) and 4 (querying which systems employ a certain type of component) are of interest when the failure of a certain component can be traced back to its working principle and according changes need to be made across the whole system. Furthermore, generalizations can be used to classify flowitems in the model into categories such as currents and voltages or data. This distinction is necessary to answer questions regarding how a certain piece of information is processed (Questions 6, 7, 9, 10 11) or regarding power paths (Questions 8 and 12 (a) to (c)).

All of these questions require the use of generalizations as defined in [161]. A hierarchy in which port types, component types and data types are defined is therefore sensible, but can be limited to the level of detail that allows to answer the above questions. On some systems this may require to refine them to the point where the protocol of the port is defined (for example CAN, USB, Ethernet), while for other systems a simple differentiation between analog and digital ports may suffice.

Formulating a guideline for the use of generalizations,

generalizations shall be used to classify blocks, flowitems and ports by important terms and concepts used throughout the development.

This concept especially comes into focus, when dealing with anomaly detection and failure detection, identification and recovery. For example, the questions 12 (concerning the fallout of a broken component) and 12b (analyzing components offline due to an electrical short in a component) would benefit from employing the following two concepts:

- a *fuse* class, which shows what fuse is the next upstream on the power path, that is triggered by the short and
- a *voltage* or *power* class, which can be used to discern the power paths from data paths or other physical values for example.

In the same way, a *data* class would be beneficial to answer Question 12c (concerning components suffering from a loss of input, as they depend on data processed by any of the components offline due to the electrical short in the specified component), which depends on being able to discern a telemetry value from the physical flow it measures.

Internal Block Diagrams and Modeling of Flowitems

The purpose of Internal Block Diagrams is to show the connections between blocks and thereby define which paths flowitems can take within the model.

To answer questions such as Question 5 (querying the component that supplies a system with power), Question 6 (concerning how certain information is being processed within a certain subsystem) Question 7, Question 8 (querying which components draw power from a certain supply component), Question 9 (querying the source of and possible influences on a certain telemetry value) or Question 10 requires detailed information on the data and power flows within a system. Therefore,

Blocks used as flowitems should be modeled to the same level of detail with which analyses are to be performed later.

To elaborate on this, the associations between flowitems on different levels, such as "Temperature X1 is part of the telemetry of Panel X, the telemetry of Panel X is part of Subsystem Y's telemetry" should be modeled to the same level of depth that is afterwards required for analyses. As the analyses often start with anomalies seen on a single telemetry value this usually requires modeling down to the measurement of every single sensor.

While the flowitems should be modeled to great depth, the itemflows modeled in internal block diagrams do not require the same level of detail. If a certain telemetry shall be traceable all the way back to the sensor generating it, the same level of detail also has to be provided in the internal block diagram. Most modern spacecraft developments rely on commercial of the shelf parts to some degree. The suppliers of these components typically do not provide the necessary information to model on a level of detail allowing to track every data path within their subsystem. Therefore, the graph analysis has to be able to cope with black boxed subsystems, of course following the paradigm that you can only query information which is available in the model. I.e. if only the specification of the telemetry provided from the subsystem is available, but

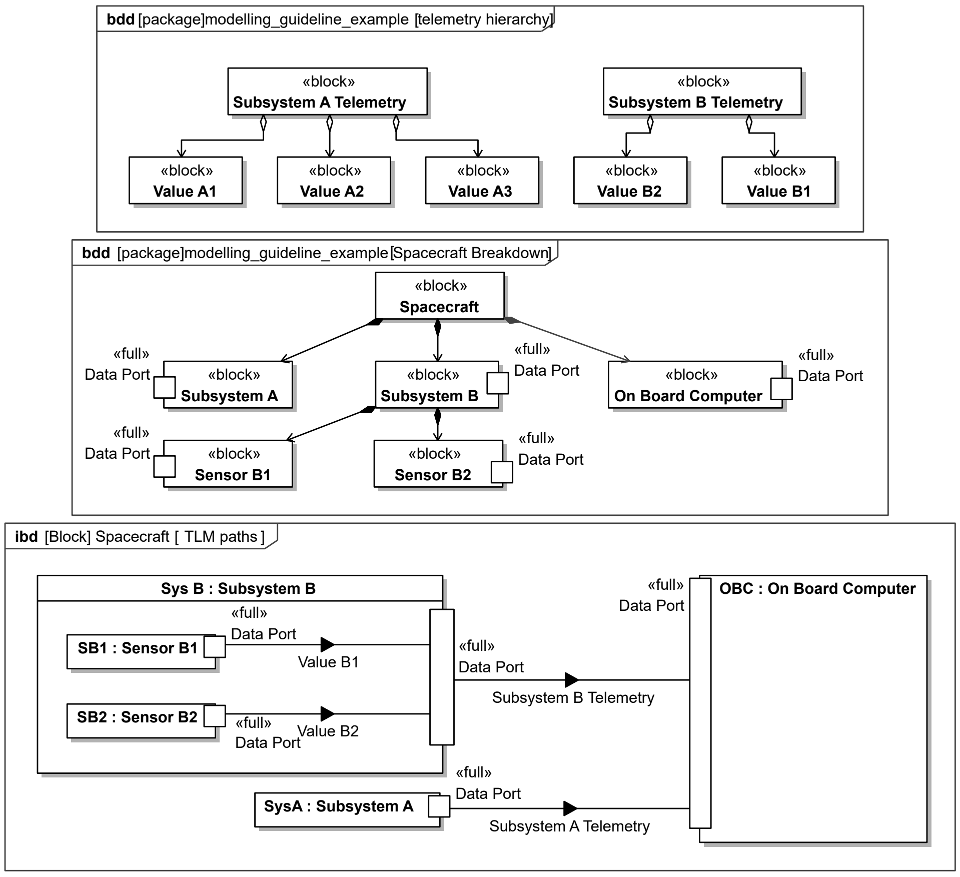


Figure 6-26.: Exemplary application of the modeling guidelines for structural SysML aspects

no information on its internal connections, setting the subsystem as a black box and transmitting a block containing all telemetry suffices.

Figure 6-26 illustrates this in an example. The upper diagram in Figure 6-26 shows the telemetry flowitems of *Subsystem A* and *Subsystem B*. The middle diagram provides a structural breakdown of the spacecraft. The bottom diagram shows the telemetry connections within the spacecraft in the form of an internal block diagram. Note here, that *Subsystem A*, for which no further breakdown is provided simply transmits its telemetry, which according to the upper diagram contains the values *A1, A2* and *A3*.

Depending on the system to be modeled, discerning between physical values and their measurements as flowitems can be a useful addition to the modeling guidelines. An example of this would be an electrical power system, where currents, voltages and the measurements thereof are transmitted.

In such a case the application of generalizations of the flowitems to the two blocks *physical value* and *measurement* resolves any ambiguity. This is especially recommended answering questions such as Question 12 as it allows backtracking power paths to the next fuse and to discern between power paths and data paths.

6.5.3. Modeling Guideline for Behavioral Aspects of the SysML Model

Answering the questions set in Section 6.2.2 also requires some modeling rules. Compared to the structural part of SysML however, these are rather sparse. Modeling state machine conditions with signals, as shown in Figures 6-14 and 6-16 allows answering Question 14 a and c (What is the shortest path from state A to state B while condition C cannot be met? Which conditions have to be fulfilled for this path) and Question 21 (In case condition C cannot be met, which states of the system cannot be reached anymore).

Connecting state machines can be achieved via reusing state machines or employing signals. Both of these practices are encouraged to achieve a higher level of interrelated, queryable information in the model. The same concepts apply to Activity Diagrams. When modeling activities, reusing existing activities to connect the diagrams can be of help to later retrieve information via the graph database.

6.5.4. Modeling Guideline for Use Cases, Requirements and Tests

In general, the modeling of use cases, requirements, tests and components satisfying requirements in SysML pursues the goal of traceability [17]. This is also the basis for this modeling guideline for use cases, requirements and tests. As explained in Section 6.4.3, a variety of constructs that are generally used to create traceability are translated to `IS_TRACED_FROM` relations in the graph. The direction of universal relations such as allocate, trace and refine needs to be specified to allow tracing across the whole graph. For trace, allocate and deriveReqt this means that the relations point from the lower level element to the higher level element. Refine relations are handled in the opposite direction, i.e. pointing from the higher level element to the lower level element. As containment relations between requirements also create an `IS_TRACED_FROM` relation, the same as extend and include relations for use cases, no further rules are necessary for the modeling of requirements. The principle that *the graph can only relay information that is available in the model* also applies to the modeling of requirements and use cases. Traceability on a level that allows answering questions such as Item 27 requires that any relevant test is defined and related to requirements as well as all components related to any requirement they satisfy. However, the following assumption is made with the queries:

Assumption: A requirement applying to a higher level component also applies to its subcomponents.

This holds true for constraints, such as "no magnetic metals may be used" but also for qualification-related requirements, such as "the system has to withstand vibrational loads up to 14g". It does not hold true for functional requirements. However, these can be sorted out by hand or by defining a requirement property that allows to filter functional requirements.

6.5.5. Summary

Overall, the modeling guidelines to transfer SysML models to Neo4j enabling such complex queries as described in Section 6.2 are few. This was intended from the beginning, as the method described in this thesis should be combinable with any other modeling strategy. Parts of the rules described above can even be neglected, if the specific questions related to these rules are not of interest. Examples on how the modeling guidelines presented here can be tailored can be found in Chapter 9

7. Universal Graph Schema for Information Management Systems

One of the advantages of graph databases is their adaptable and easily extendable schema. Additional insights to the system under analysis can be gained by loading the documentation of the mission and its infrastructure and the correspondence of the development team into the database. Connecting this information with the SysML Model of the mission provides further context to the information and shall thereby make it simpler and easier to retrieve and assess. Thereby the database not only allows to conduct analyses on the system but furthermore can be extended to a full scale enterprise search engine. Compared to traditional enterprise search engines, using the SysML Model as a backbone allows the users to overcome the problems stated in Section 2.2.6, as the SysML Model provides further context for the search. Thereby if a certain piece of information cannot be found on the component the user suspected it to be related to, the search can be widened along interfaces, hierarchical levels, etc. In the best case, assessing whether a piece of information is outdated can be achieved by querying whether any newer information is available concerning the same aspect of the system.

As already explained in Chapter 2 and 5, a variety of information management systems come into play here, from traditional text documents and emails to issues in collaboration management systems and chat messages in team communicators. Ideally all of these information sources are treated the same way, as no prior assessment of a piece of information's worth is possible. To achieve this, a common schema is necessary, that can be applied to the information stored in any of these systems. Therefore, it has to be ensured that the necessary meta-information is available in every information management system to apply this schema. Furthermore, the granularity of the information presented has to be considered. Simply knowing that a certain component is mentioned in a document spanning several hundred pages may be of little help, while granting every comment in a chat protocol its own node may prove as too granular.

7.1. Collection of Questions for the Information Management System Graph Schema

To allow the definition of a universal graph schema for information management systems, a set of questions has to be defined first, which ought to be answered by the schema. The collection below serves as a baseline for the rest of the chapter:

1. What information is available concerning Component X?
2. What information is available concerning Component X or any component connected to it newer than a certain date?
3. What information is available concerning Component X or any of its subcomponents newer than a certain date?
4. What components does a specific document describe?
5. Which information management systems hold information concerning Component X?

6. Is a piece of information Y still applicable or is it outdated?
7. Which test documents are available regarding a certain requirement?

7.2. Necessary Meta-Information for an Adaption into the Graph-Database

The graph schema shall be independent of the origin of the data. Hence, a common set of metadata needs to be defined which can be retrieved for every piece of information. Based on the questions above, the following metadata should be retrieved as a minimum:

- Date
- Text
- Internal organization (i.e. for a doc previous/next page or section, for a chat previous/following message)
- Uniform Resource Locator (URL)

The date is necessary to determine the order in which information was entered independent of the information management system in which it was entered. The text can be used to digest information directly in the graph database as well as to create links between the information and the graph of the SysML Model by filtering the text for the names of SysML nodes. The URL allows jumping directly to the original information. The internal organization allows displaying information nodes in their original context. For example one message may contain the name of a SysML model element and thereby can be linked to that SysML element. If the message is part of a conversation, the other messages of that conversation are of interest as well. Therefore, linking information in its original context is a valuable way of increasing accessibility of information.

The date and text resources can be found on practically any information shared in any information management system. This is obvious for the date tag, as any information has at least a specific date at which it was created and may additionally contain the information when it was last updated. The text usually contains the content of the information. However, for video- or audio-recordings of meetings, tests etc. this may not be the case. This has to be considered and is the reason for the next required meta datum: in case a piece of information that does not contain text is referenced somewhere else, it can be found via the piece of information referencing it. The same applies for pieces of information that cannot directly be referenced to an element of the system model. Therefore, depicting the inner structure of an information management system is beneficial.

A URL allows jumping from the search result in neo4j back to the original item in its original context. For most information management systems accessible via a browser, URLs are provided automatically. Where no URL is provided, one can be generated, for example by storing the respective files on a network access storage.

The author of a piece of information can also be retrieved in most information management systems. Including the information would however result in providing users of the graph database with the ability to derive all kinds of personal information on other project members, such as their typical working time or creating rankings on co-workers depending on the amount of activity they show throughout all information management systems. Providing means to gather and interpret information of such sensitivity is not the aim of the research presented here, hence the author of a piece of information is not part of the schema.

7.3. Proposed Graph Schema for Information Management Systems

To distinguish between the representation of information within the development project and the representation of technical components, two new labels are introduced: The `:TEC`-label is applied to all elements described

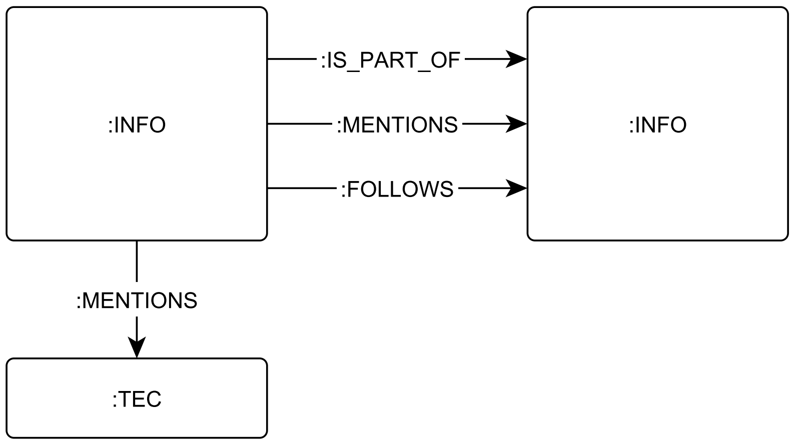


Figure 7-1.: General graph schema connecting information about the system with the model of the system. *Note: The `:INFO`-label is displayed twice for readability*

in Chapter 6. The `:INFO`-label is applied on all informational nodes in the graph database coming from the various information management systems. Additionally, each information management system is provided with its own label.

In the following a general graph schema for information management systems is presented, before its application on a selection of information management systems is discussed.

7.3.1. General Graph Schema for Information Management Systems

Figure 7-1 shows the general graph schema for information and its relation to `:TEC`-nodes. The schema consists of two node types and three relation types. The only relation between `:INFO` and `:TEC` nodes are of the `:MENTIONS` type. These relations shall be used when the content of the `:INFO`-node is related to the `:TEC` node. As information can also reference other information instead of a `:TEC` node, the schema allows the `:MENTIONS` relation also between `:INFO` nodes. Thereby correlations between information management systems can be modelled, for example emails referencing issues in collaboration management systems. It also allows modelling correlations within an information management system, such as the hyperlinks in a wiki or a collaboration management system.

The hierarchy of `:INFO` nodes is presented by the same relation types as the hierarchy of `:TEC` nodes, the `:IS_PART_OF` relation. Apart from this general purpose hierarchical relation, information can also be connected by chronologically following another information node in the same information management system. For example consecutive messages in a team collaborator channel can be modelled in this manner as can be replies to an email.

Details on how the general graph schema can be applied to various information management systems are presented in the following. It may not be possible to directly replicate these examples on other information management systems. However, showing four different information management systems, i.e. Email, wikis, collaboration management systems and team collaborators provides sufficient guidance to reciprocate the schema on other information management systems.

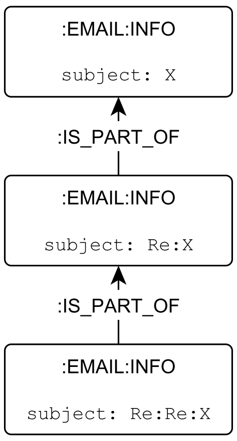


Figure 7-2.: Application of the general graph schema for information on electronic mail, displaying the hierarchical order of electronic mail conversation building on the mails' subject.

7.3.2. Application on Electronic Mail

Figure 7-2 displays the application of the general graph schema for information on electronic mail. Each Email is built translated as a single node. As a reference, the Request for Comment 5322 of the Internet Engineering Task Force is used, i.e. the current standard which defines electronic mail (cf. [162]).

Emails which shall be considered as reply to a former message keep the same subject adding a "Re:" at the beginning of the subject to identify the message as a reply to a former message [162]. Therefore, conversations in emails are easy to identify.

A universal resource locator can for example be provided by regularly exporting emails to a network access storage and reconstructing respective links to retrieve the emails. This extra step slows down the transfer of information and actions required by users are unattractive as they are prone to errors. However, Email suffers from a difficult division of private information and information concerning an actual project. For example an employees email account may not only hold project specific information but also information on their salary, contract or any other personal business. Transferring emails to be searchable by others to a folder by hand allows the users to filter the content they want to share. Adopting so-called customer relation management systems such as Hubspot, Salesforce or Oracle (cf. [163]) would be another possibility how information from emails can be gathered sanitized from undesired personal information.

Applying the schema as depicted above results in a disjointed graph of chains of email messages. Each email is only bound to its previous and following message with the same subject. This clear separation allows for precise queries, as the boundaries of a conversation via email are rather clear: A new subject strongly indicates a new conversation. Connections from the emails to other information or to `:TEC` nodes relate the conversations to other topics. Such relations can for example be identified by the usage of specific technical terms in the text of a message in the conversation or by the usage of the universal resource locator of another piece of information available in the graph.

7.3.3. Application on Wikis

The structure of wikis is already detailed in Section 2.2.4. Each wiki-page is represented as a node. The construct of hyperlinks on pages can be reconstructed well with the `:MENTIONS` relation. Multiple data points can be taken over for the `date` property: The creation date of the page, the last access or the last change. As the gain of adding the date as a property to the information is mainly to help its chronological order, the date of the last change shall be taken over into the graph database. This implies that any information added after the wiki page was changed the last time on the same topic might depict additional information or outdate information in the wiki page. In case no newer information is available however, the wiki page can be assumed up-to-date. As wikis do not build on hierarchical order, no `:IS_PART_OF` relations are used.

As a result, the whole wiki is displayed as an interconnected graph. In case pages or groups of pages exist that do not relate to any other pages within the system, the graph is disjointed. The same principles for relating the graph representation of the wiki with other informational sources in the graph database or `:TEC` nodes can be applied as for emails (see Section 7.3.2).

Compared to email conversations, wiki pages usually contain information in a stand-alone format. Therefore, their specific relation to other pages may not be as important for queries as the wiki-page's relation to any `:TEC` node.

7.3.4. Application on Team Collaborators

As an example on team collaborators, an application of the graph schema on the platform slack is discussed in the following. Messages in Slack are organized by channels, to which members of the development team can subscribe. Any subscriber to a channel may also post in the channel or reply to messages posted by other team members. The replies to a message can either be written as new message in the channel, or as thread-message directly related to a specific previous message. Typically, threads are applied once multiple conversations are carried out simultaneously in the same channel. However since both possibilities prevail, it is difficult to mark out specific conversations in such a system.

The graph schema for such a system duplicates the relations already present in the team collaborator. Messages on channel level follow one another while replies in a thread create a branch from the message chain on channel level.

Two possibilities arise to treat channels: They can either be stored as separate node, with an `:IS_PART_OF` relation from every message within the channel or be stored as property on the nodes. To be compliant with the general graph schema for information, channels would need metadata such as date, text, and a universal resource locator. The last presents no problem, whereas the date of a channel's founding does not imply anything regarding how up-to-date its information is. The text i.e. the informational content represents another challenge to the general informational graph schema, as the actual content of the channel consists of all its messages. However, it might still make sense to take the channel in as an informational node: Channels to discuss technical subjects are often named in accordance to elements of the system under development. Any information discussed in such a channel should therefore relate directly to the respective system element. This information might not be shared otherwise, making it a valuable key for connecting information. However, this information can also be represented by storing it as a property on the message nodes. Thereby only nodes that contain actual information are included in the schema. This approach provides the added benefit of keeping the schema simpler and thereby allowing for more concise queries. Figure 7-3 shows an exemplary graph excerpt for messages in a team collaborator.

7.3.5. Application on Collaboration Management Systems

Collaboration management systems provide the by far most extensive system of relations within the data itself presented in this chapter. Originally designed to facilitate the management of software projects, a typical workflow on a collaboration management system may look similar to Figure 7-4.

Not every piece of information is related to a `:TEC` node and additionally various spellings and synonyms are used by humans to refer to the same thing. Therefore, relating pieces of information with each other provides

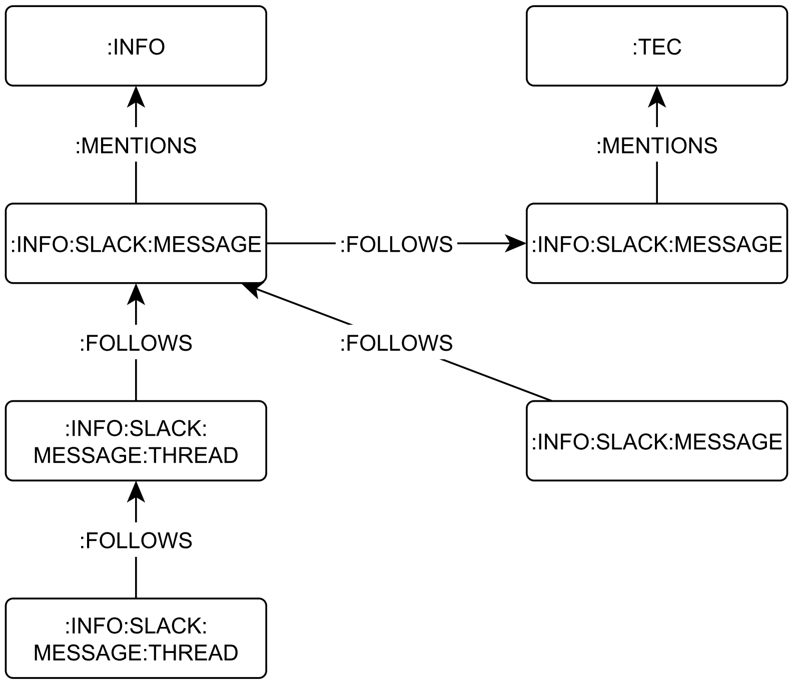


Figure 7-3.: Exemplary application of the informational graph schema on messages in the team collaborator Slack

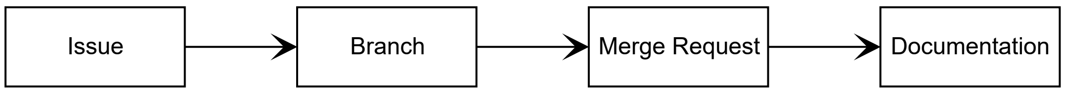


Figure 7-4.: Typical workflow in a collaboration management system

a meaningful technique to gain further access to information on a specific `:TEC`-node. The relations presented in Figure 7-4 can be used to link information that is in any way related to each other by the workflow itself. Additionally, collaboration management systems already contain their own hierarchy which can be represented by the graph.

A more in-depth analysis of this shall be provided on the example of the GitLab software (see [68]). The software builds on the code-versioning system "git" (see [164]) and incorporates issue tracking, wikis, fine-grained access management, time tracking and further features, extending the system from a mere code versioning tool to a fully fledged collaboration management suite. The central organizational element of GitLab are repositories, i.e. folders containing files of the system under development and their complete history on a line-by-line basis. Repositories can either be directly attached to a user profile or to a group. For larger projects the second option is more conventional. Apart from containing the files of the actual implementation of the system, repositories can be split into multiple branches, allowing parallel development of multiple aspects of the system that can then be merged into the so-called "main" branch again, which usually provides the integrated, runable and tested version of the files (see workflow in Figure 7-4). Figure 7-5 provides an overview of the hierarchy in a gitlab-group. Additionally, the software allows cross-linking any of its elements:

- Users can be active in repositories.
- Commits always affect one or more files.
- Issues, merge requests and commits can be referred to from each other, e.g. commits can explicitly be mentioned in merge requests or issues in commit messages.

As described in the previous chapters no direct correlations to the authors shall be drawn for data privacy reasons, they shall also not be transferred to the graph. Apart from this, the hierarchy presented in Figure 7-5 can directly be taken over as well as the references already present in the system. Figure 7-6 shows an exemplary implementation of the general graph schema on the collaboration management system GitLab. The implementation presented omits the wikis provided by GitLab with every project, as a wiki implementation of the general information graph schema was already presented in Section 7.3.3. As can be seen the schema is the most elaborate information schema presented yet. The elaborate relations network within collaboration management systems can be seen as very helpful in identifying correlations between nodes. It also requires a thorough understanding of the applied workflow and the resulting graph network.

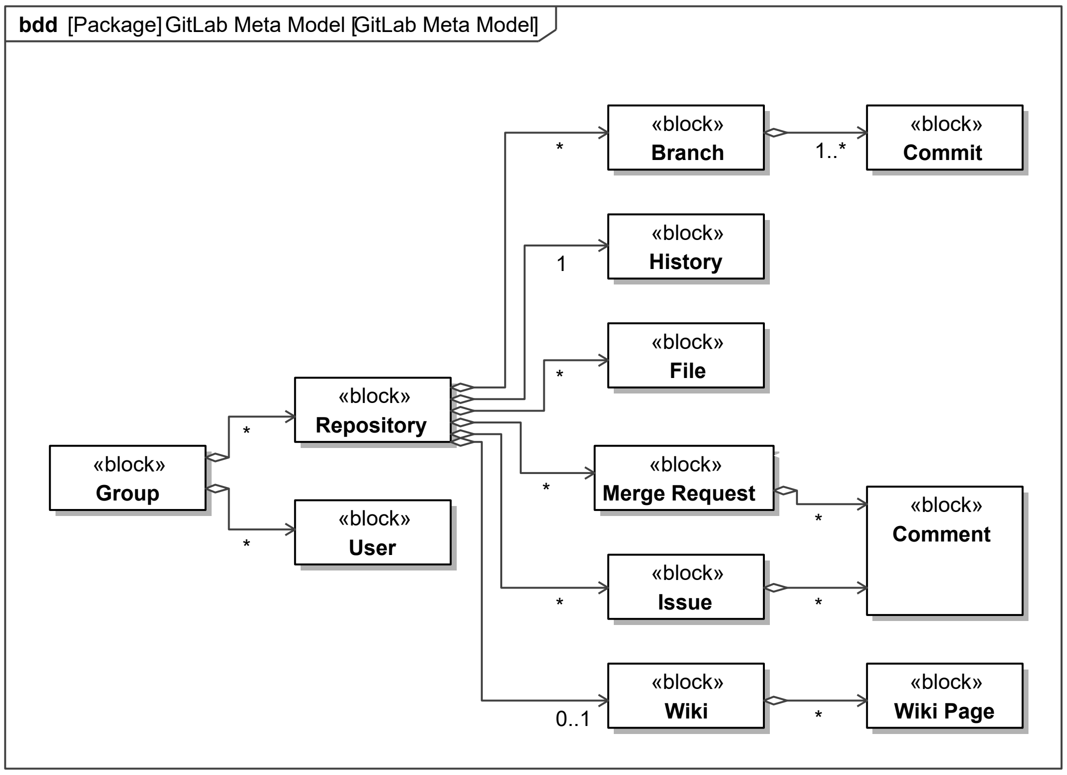


Figure 7-5.: Hierarchy of GitLab elements, exemplary for collaboration management systems.

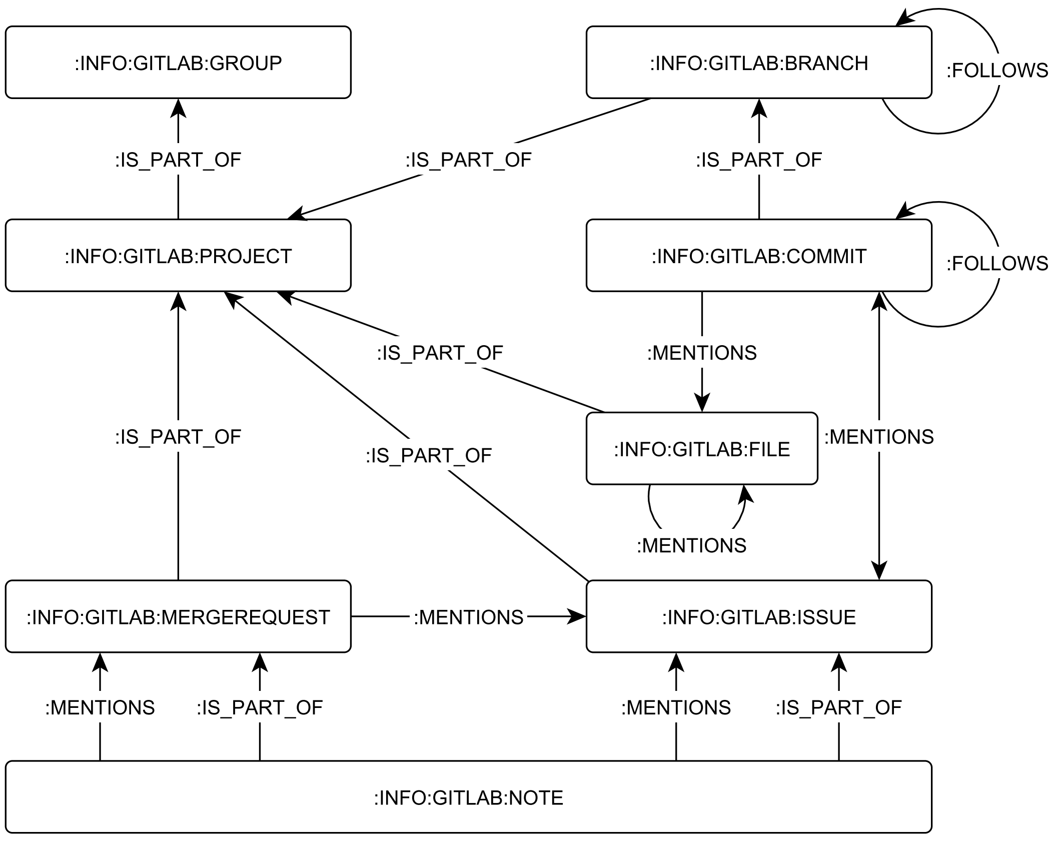


Figure 7-6.: Exemplary implementation of the general graph schema for information on the collaboration management system GitLab

7.4. Connecting Elements of the SysML Model with Information

A general graph schema for information management systems was proposed, which allows linking pieces of information to each other as well as to the respective elements of the SysML model's graph representation. Applications of the graph schema on three different information sources were shown, providing a baseline on how information management systems can be transferred into the graph. A procedure to identify which information refers to which element of the SysML model is still missing and a crucial part for a good connection between information and system model.

The simplest approach is to look for the usage of proper names defined in the SysML model within the text of the information nodes. This approach works well in theory, as discussions, reports and documentation usually work with the same names as provided in the SysML Model. It falls short when words have multiple meanings; A channel of an electrical power system may be mixed up with channels in a team communicator and port names such as "+" and "-" on a battery or "in" and "out" on a flow sensor would result in a high amount of falsely discovered relations. A second shortcoming of the approach are synonyms and abbreviations. Instead of "Microcontroller", "Chip" or "MC" are also in use, as are "thermometer" and "temperature sensor".

Other approaches are thinkable as well, such as identifying named entities in the information and comparing them with names in the system model. Respective programming libraries are available, such as "SpaCy" for python (see [165], [166]). However, comparing these named entities with the system model requires the names to either fit directly – in which case no prior recognition is necessary, as the list of names in the system model already provides a dictionary of the words of interest on its own – or to follow up with more in-depth language processing that allows to identify the base-noun out of any version provided in the text, such as synonyms in use, plural, abbreviated forms etc. While such an approach might reveal additional relations and thereby the Type I errors in the relation search. At the same time it inevitably increases the amount of Type II errors, i.e. there's a higher chance of falsely identified relations.

The second approach is typically employed when it is not previously known which entities are contained in a dataset and hence how to create relations. An example of such an application is the investigation of the Panama Papers (cf. Section 2.2.8 as detailed by Cabra and Kissane [120]). The names of the elements in the SysML model provide a detailed dictionary that can be used to structure the information. Therefore, more sophisticated solutions such as running natural language processing on all information inserted into the graph database is not necessary. However, in case no sufficient amount of relations can be created by matching the names used in the SysML Model, more sophisticated approaches can be used.

8. Implementation

8.1. SysML Graph Translation

The translation of MagicDraw SysML Models into the graph is implemented using Python 3.8. Figure 8-1 provides an overview of the implementation. The `retrieve_SysML_model.py` script reads the SysML-Model generated in MagicDraw 19.0 (`.mdxml`-file) and extracts SysML elements and relations as lists of python `dict` variables. The lists are stored as `.json`-files. This ensures a separation of concerns and enables testing the retrieve functions separately from the insert functions, which insert the extracted SysML elements in Neo4j. It also enables future developers to use only the extraction part of our code and insert the extracted SysML elements anywhere else.

The `insert_SysML_in_neo4j.py` script loads the stored `.json`-files and writes every component via a separate query into the Neo4j database. This procedure is quite inefficient regarding execution times but allows for faster debugging compared to bulk-load methods such as described in [167]. As the execution time on a normal office PC is still under 3 minutes for several thousand SysML elements, the advantage in debugging prevails.

The code is open sourced under MIT License. It can be found under the following link: `https://gitlab.lrz.de/lrt/sysml_graph_analysis_tool/`

8.2. Information Management Systems Graph Translation

An implementation of the graph translation for various information management systems is provided in the repository as well, following the modeling schema defined in Chapter 7. The current version of the software includes extract transfer load functionality for the following information management systems:

- GitLab
- Emails
- Slack
- PDFs

All of them these functions are similarly structured as the SysML extract transfer load code, with an extract script and an insert script and possibly another script to afterwards create relations between the elements of the code.

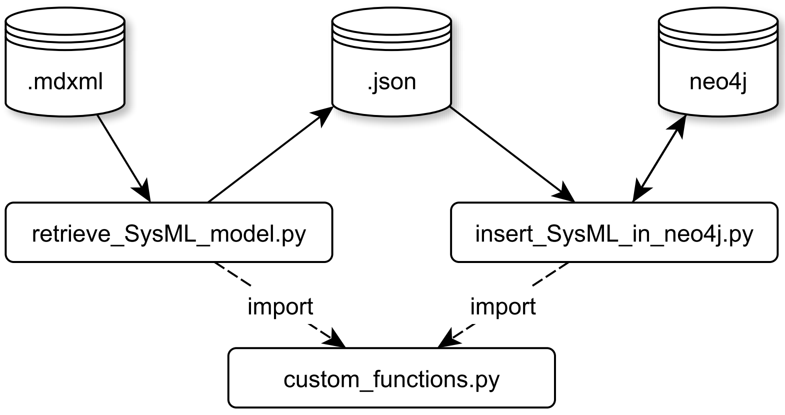


Figure 8-1.: Setup of the extract transfer load software to import MagicDraw SysML data to Neo4j

9. Analyses on the Munich Orbital Verification Experiment II

This chapter shows the application of the modeling guidelines proposed in Chapter 6 on the MOVE-II Space Mission. The spacecraft itself is modeled as well as all ground infrastructure used to run the satellite. After a description of the model and how the modeling guideline presented in Section 6.5 were implemented on the model, the chapter develops a variety of graph queries, which show how the graph schema developed in Chapter 6 can be used to derive information from the SysML Model. In a further step, the documentation available for the MOVE-II Spacecraft, albeit in the form of readme-files for the software, dedicated system documentation or previous review-information is loaded into the database in accordance to the schema developed in Chapter 7 and an assessment on how well the information can be linked with the SysML model of the mission is provided.

9.1. The MOVE-II SysML Model

The focus of the MOVE-II SysML Model lies on the structural aspects of the mission, including electronics and software. The data paths from any sensor on the spacecraft down to the telemetry database are modeled in detail. Additionally, the requirements of the mission are incorporated in the model and linked to any components implementing them. Open spots are left in the requirements modeling to show how graph analyses can be used to identify open potential in a SysML Model.

9.1.1. Overview

Figure 9-1 shows an overview of the model's structure. It is built up by the packages "Satellite Structure", "Ground Infrastructure", "Requirements" and a "Library" package. The package "Satellite Structure" is broken down by subsystem with an extra package "General Flowitems", that contains flowitems which cannot be assigned to any specific subsystem, such as the package "All Data Transmissions", which describes all data flow from the spacecraft to the ground station.

Inside the respective subsystem packages are:

- Package Subsystem X Flowitems: Block definition diagrams that define all flowitems and their subsystem specific packages (both in the "Subsystem X Flowitems" folder)
- Package Subsystem X Hardware: The block definition diagrams and internal block diagrams defining the subsystem's hardware setup. I.e. all data connections between integrated circuits, all power connections and any other relevant hardware component are defined inside this package, together with their respective connections.
- Package Subsystem X Software: The block definition diagrams and internal block diagrams that specify the subsystem's software running on subsystem-specific hardware. In case the software can be broken down into multiple modules, such as driver and logic modules, this is displayed to a certain degree within this package, including the data transfers between the modules.

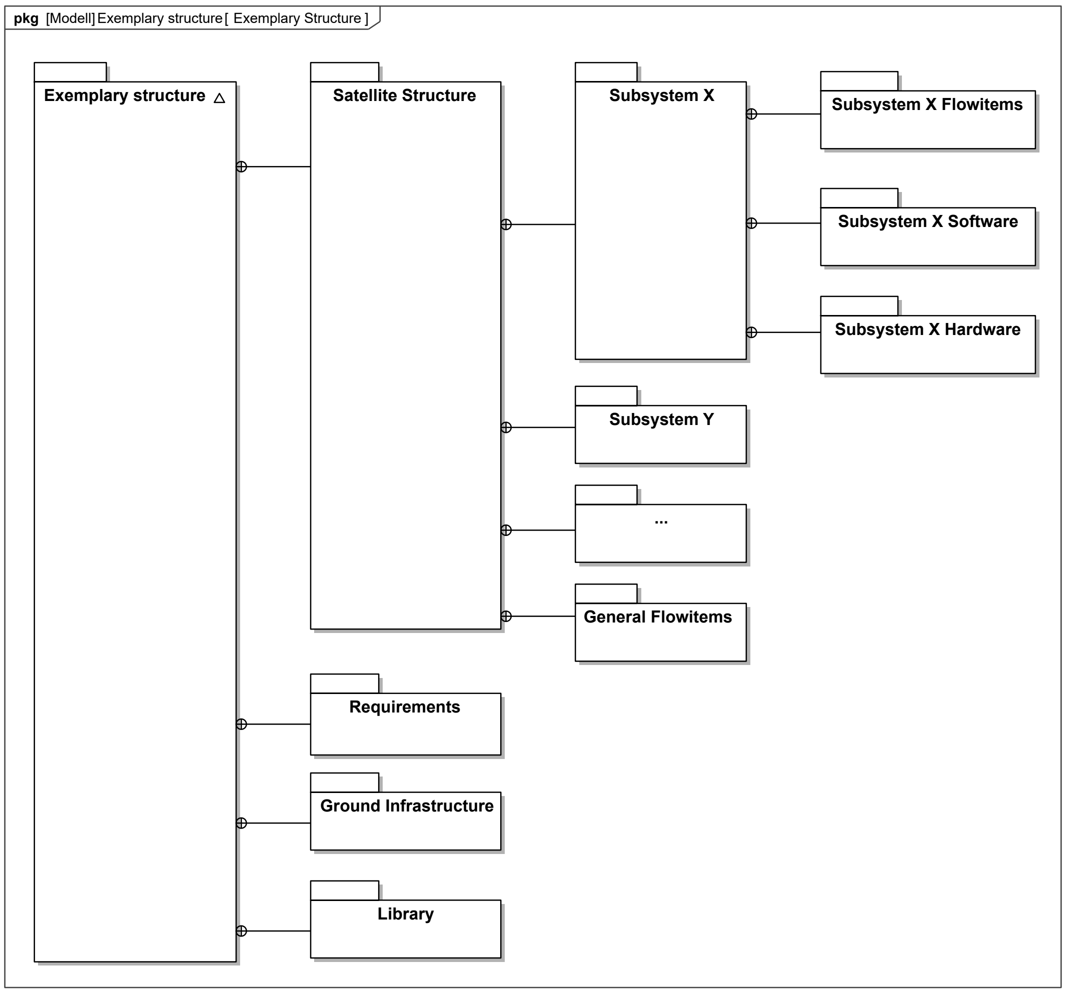


Figure 9-1.: MOVE-II package overview

This breakdown allows to track any telemetry of the spacecraft from their source to the database on ground. Similar content can be found in the "Ground Infrastructure" package. Divided into two sub-packages, "Ground Station" and "Operations System", the model details any connector on the ground infrastructure, as these are prone to environmental influences such as corrosion or server maintenance and hence were the cause of several errors in the past. On the top-level of the model the context diagrams which connect the spacecraft and the ground infrastructure are provided.

The requirements package is broken down according to subsystem as well and attributed to use cases which were built on top of the already existing stakeholder expectations. The model largely relies on information that can be found in the system documentation of MOVE-II, as well as the documentation of the Critical Design Review of MOVE-II (see [168], [169]).

9.1.2. Implementation of Modeling Guidelines

Section 6.5.1 requests the following considerations:

- Defining the purpose of the model
- Defining which systems/subsystems shall be modeled
- Defining the targeted modeling depth
- Ensuring the model's maintainability

The purpose of the model is to show queries along the whole datapath of the system to help identify potential reasons for errors. Apart from this, the power path shall be modeled to analyze fallout-scenarios caused by short-circuits. In the extension of this purpose, the model ideally allows connecting information from various sources.

The scope of the model shall encompass the whole on-orbit operations of the mission. I.e.t̃he spacecraft as well as the ground station and operations infrastructure. It shall not cover the launch segment.

The modeling depth should allow describing the power and data flows throughout the spacecraft. Modeling of microcontroller firmware is neglected, however the on-board computer's software shall also be modeled in detail. Subsystems to which no further documentation is available shall be modeled as black box.

The model's maintainability is assessed as good, since no changes in the spacecraft are expected after its launch. The ground station setup was updated multiple times during the operations, in hardware as well as software. The model as shown here describes the ground station's setup in the year 2019.

Implementation of the Modeling Guideline for Structural Associations

As described in Section 6.5.2, all blocks are part of a structure of SharedAssociations and PartAssociations.

Figure 9-2 shows the hierarchy that connects the "System MOVE" block down to the lowest-level elements "Beacon Parser" and "Beacon Poster" in the Backend of the Operations System.

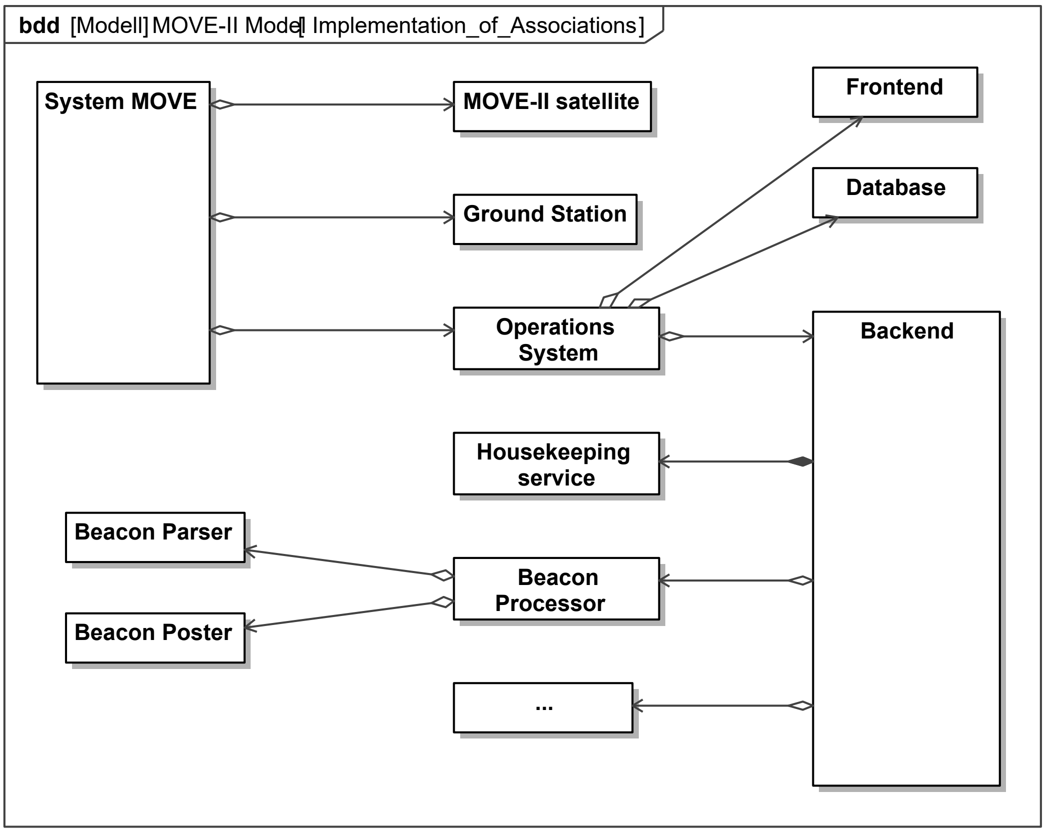


Figure 9-2.: Excerpt of the associations hierarchy of the MOVE-II system, showing the relations over five levels between "System MOVE" and "Beacon Parser"/"Beacon Poster"

Implementation of the Modeling Guideline for Generalizations

The modeling purpose to analyze fallout-scenarios requires the inclusion of a fuse-class, by which components conducting electrical power can be classified as a fuse that can shut down and stop delivering power to all subsequent components in its path (see elaborations in Section 6.5.2). Figure 9-3 shows the components which were defined as fuses.

For the same reason, the model differentiates between two kinds of flowitems: Data flowitems and physical flowitems. This allows to differentiate between the measurement of a voltage or current and their actual flow and is therefore necessary to analyze fallout scenarios. This is resolved via a singular class, the "physical flowitem" class, which is a generalization of all physical flowitems. No data flowitems class was defined, as all flowitems that are not physical flowitems in the model are considered data flowitems.

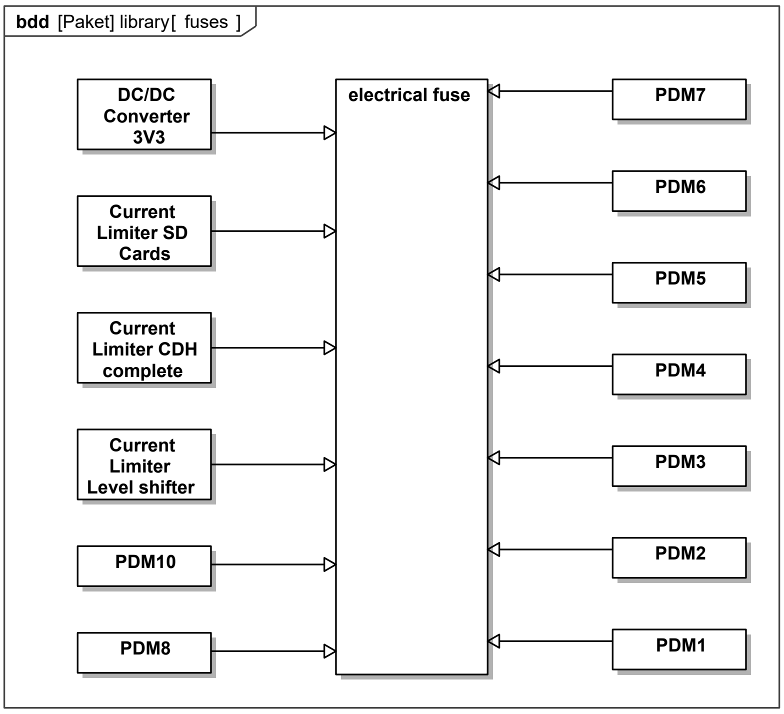


Figure 9-3.: Application of the generalizations guideline, definition of fuses for components in the MOVE-II SysML model

Implementation of the Modeling Guideline for Flowitems

Continuing with the modeling of flowitems (see Section 6.5.2), the level of detail to which the MOVE-II flowitems are modelled can be derived from the purpose of the model. As the model shall enable data queries on all telemetry values accessible in the MOVE-II telemetry database, this defines the list of flowitems to be modeled. Any telemetry value of the spacecraft used by the operators shall be modeled whereas logs of the various software elements aboard the spacecraft for example can be neglected.

Similar to the handling of instantiated blocks with aggregations and compositions, a flowitem hierarchy is built for all data flowitems.

Figure 9-4 shows the exemplary hierarchy for the flowitem "Sidepanel Y- Temperature OW1".

The connection of physical flowitems with data flowitems is realized via undirected associations, as depicted in Figure 9-5.

Implementation of the Modeling Guideline for Internal Block Diagrams

The majority of the information that can be drawn from the MOVE-II Model is implemented in internal block diagrams. The diagrams are generally separated into electrical views and data views, in which the connections are modeled and populated with the respective flowitems. Compared to a typical modelling style, no extra diagrams or views are generated for readability. The diagrams generally contain the complete power connections or data connections of one subsystem, as the purpose of the model is to provide a good basis for analysis in the graph. Any specific views that people are looking for are generated in the graph and hence do not need to be generated as diagrams. Hence, no special views that allow for simple readability are necessary, which reduces the modelling effort. Figure 9-6 shows this on the example of the satellite's power path.

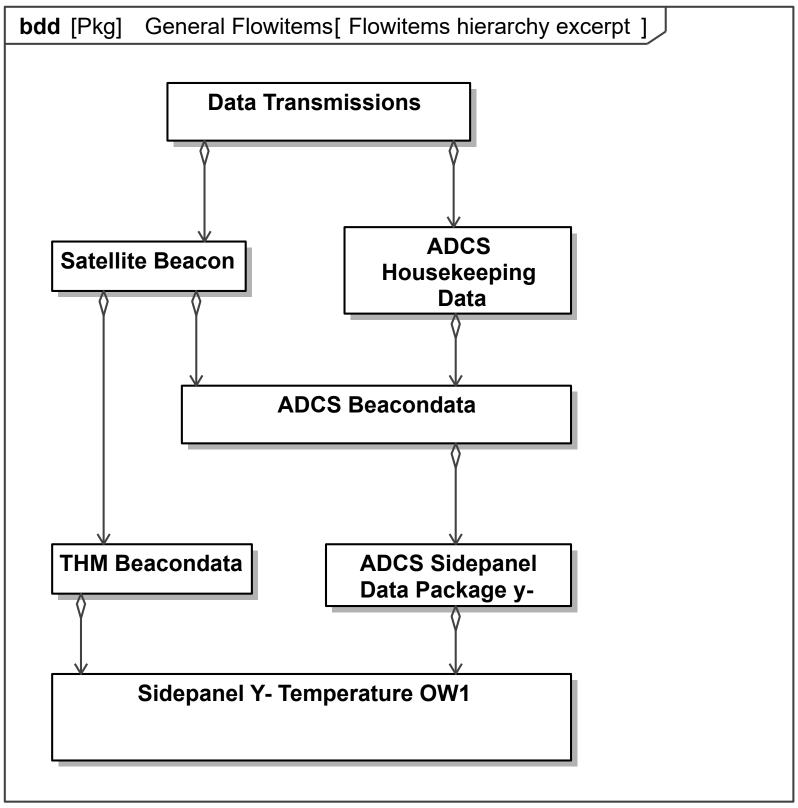

bdd [Pkg] General Flowitems[Flowitems hierarchy excerpt]
Data Transmissions
Satellite Beacon
ADCS Housekeeping Data
ADCS Beacondata
THM Beacondata
ADCS Sidepanel Data Package y-
Sidepanel Y- Temperature OW1

Figure 9-4.: Flowitems hierarchy example for the MOVE-II SysML Model

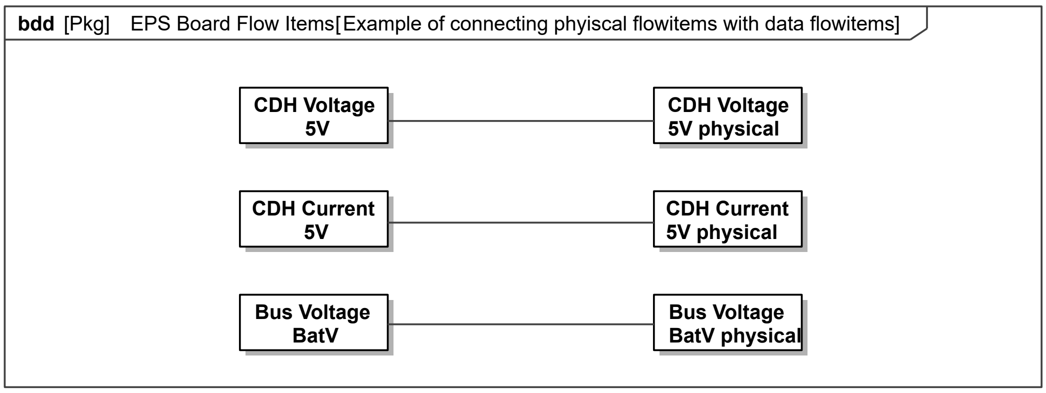

bdd [Pkg] EPS Board Flow Items[Example of connecting phyiscal flowitems with data flowitems]
CDH Voltage 5V
CDH Voltage 5V physical
CDH Current 5V
CDH Current 5V physical
Bus Voltage BatV
Bus Voltage BatV physical

Figure 9-5.: Connection between physical- and data flowitems

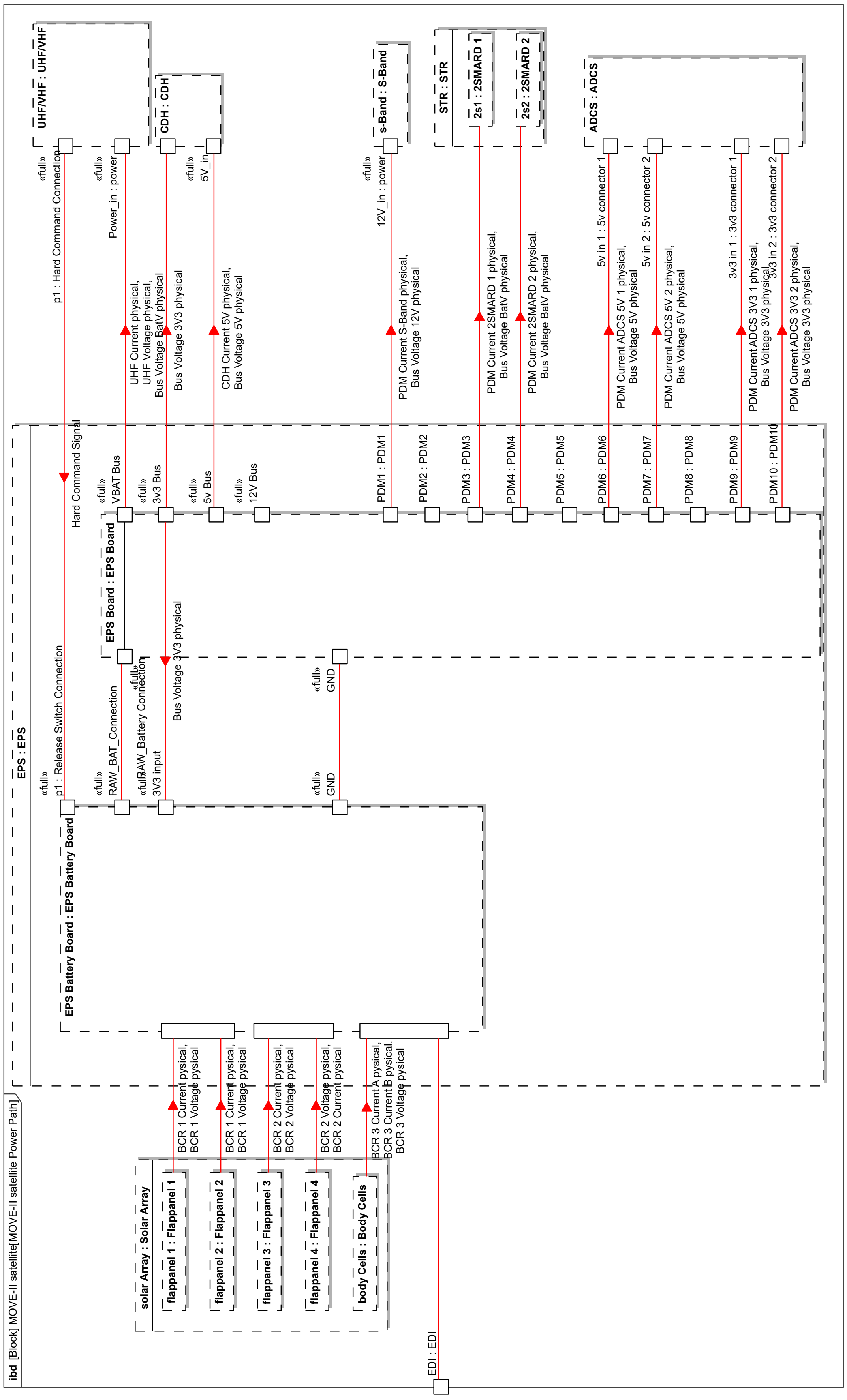

ibd [Block] MOVE-II satellite[MOVE-II satellite Power Path]
UHF/VHF : UHF/VHF
CDH : CDH
s-Band : S-Band
STR : STR
2s1 : 2SMARD 1
2s2 : 2SMARD 2
ADCS : ADCS
EPS : EPS
EPS Board : EPS Board
EPS Battery Board : EPS Battery Board
solar Array : Solar Array
flappanel 1 : Flappanel 1
flappanel 2 : Flappanel 2
flappanel 3 : Flappanel 3
flappanel 4 : Flappanel 4
body Cells : Body Cells
EDI : EDI
«full» p1 : Hard Command Connection
«full» Power_in : power
«full» 5V_in
«full» 12V_in : power
5v in 1 : 5v connector 1
5v in 2 : 5v connector 2
3v3 in 1 : 3v3 connector 1
3v3 in 2 : 3v3 connector 2
UHF Current physical, UHF Voltage physical, Bus Voltage BatV physical
Bus Voltage 3V3 physical
CDH Current 5V physical, Bus Voltage 5V physical
PDM Current S-Band physical, Bus Voltage 12V physical
PDM Current 2SMARD 1 physical, Bus Voltage BatV physical
PDM Current 2SMARD 2 physical, Bus Voltage BatV physical
PDM Current ADCS 5V 1 physical, Bus Voltage 5V physical
PDM Current ADCS 5V 2 physical, Bus Voltage 5V physical
PDM Current ADCS 3V3 1 physical, Bus Voltage 3V3 physical
PDM Current ADCS 3V3 2 physical, Bus Voltage 3V3 physical
Hard Command Signal
«full» VBAT Bus
«full» 3v3 Bus
«full» 5v Bus
«full» 12V Bus
PDM1 : PDM1
PDM2 : PDM2
PDM3 : PDM3
PDM4 : PDM4
PDM5 : PDM5
PDM6 : PDM6
PDM7 : PDM7
PDM8 : PDM8
PDM9 : PDM9
PDM10 : PDM10
Bus Voltage 3V3 physical
«full» p1 : Release Switch Connection
«full» RAW_BAT_Connection
«full» RAW_Battery Connection
«full» 3V3 input
«full» GND
«full» GND
BCR 1 Current pysical, BCR 1 Voltage pysical
BCR 1 Current pysical, BCR 1 Voltage pysical
BCR 2 Current pysical, BCR 2 Voltage pysical
BCR 2 Voltage pysical, BCR 2 Current pysical
BCR 3 Current A pysical, BCR 3 Current B pysical, BCR 3 Voltage pysical

Figure 9-6.: MOVE-II power paths, example of large diagrams reducing the effort to implement the model

To further reduce the modeling effort, the data flowitems are aggregated into packages that can be transmitted instead of transmitting a series of single items. Figure 9-7 shows this on the example of the top-level data paths within the satellite, while Figure 9-8 delivers the respective internal block diagram showing how the data is then transmitted to the ground and further separated into the original telemetry values to be stored in the backend-database. The hierarchy of the telemetry "UKW PA Current" is provided, displaying the mechanics of reducing modeling effort by packaging telemetry. The model contains a total of over 300 itemflows, and over 950 ports, associated with over 640 blocks and over 530 instances of these blocks. It may therefore be considered a model of medium size.

Implementation of the Modeling Guideline for Use Cases, Requirements and Tests

A modeling guideline for Use Cases, Requirements and Tests is proposed in Section 6.5.4. A total of over 230 requirements are modeled for MOVE-II. In accordance with the guideline, the requirements for MOVE-II are traced among each other and to the respective use cases and blocks in the system. To demonstrate queries that analyze missing links within the model, not all requirements are traced completely. Test cases and test documentation for MOVE-II are not modeled but are taken from respective documentation and linked in the graph in Section 9.3.3.

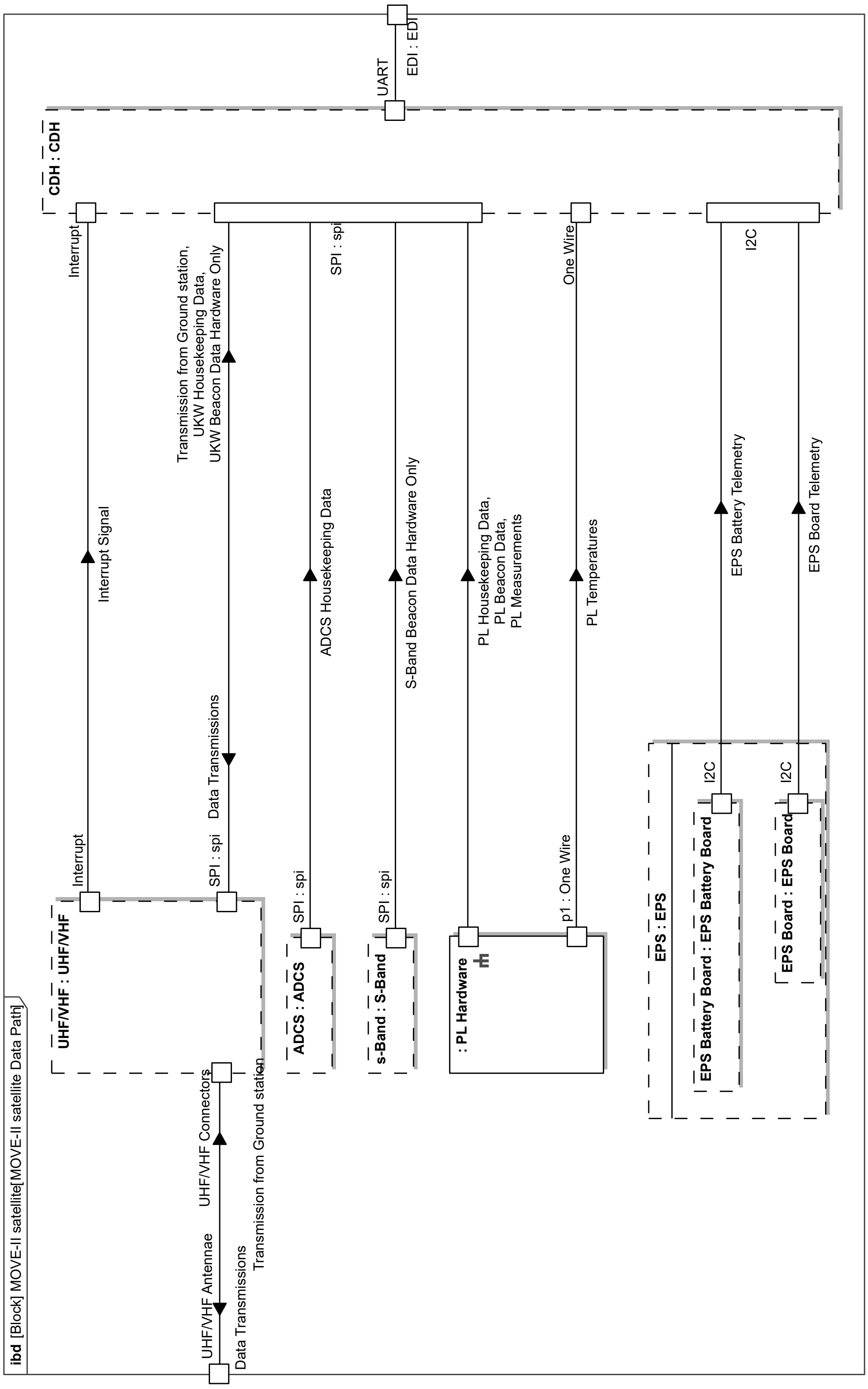


Figure 9-7.: Top level data paths within the MOVE-II satellite, example of aggregating telemetry to reduce modeling effort

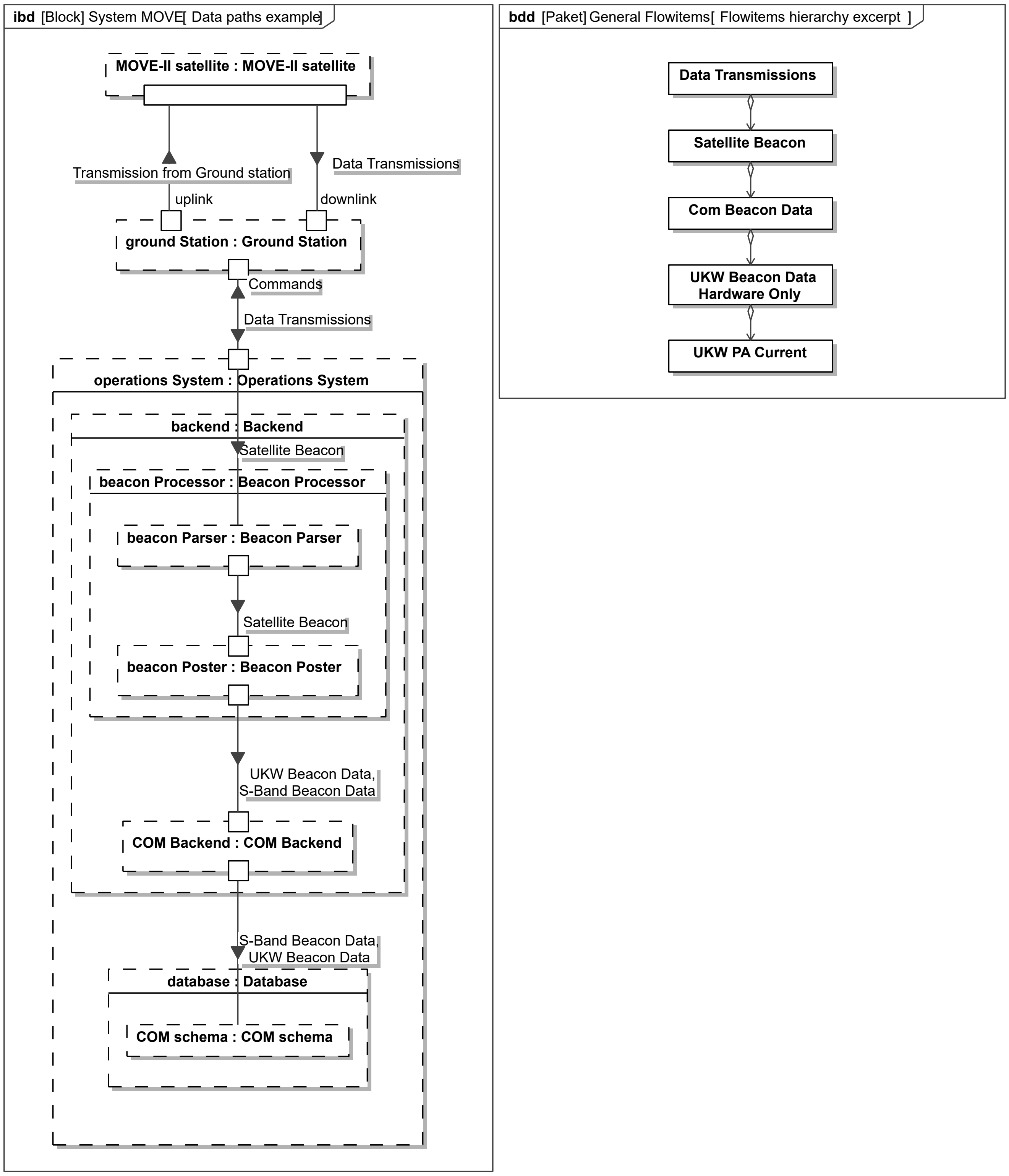

ibd [Block] System MOVE[Data paths example]
MOVE-II satellite : MOVE-II satellite
Transmission from Ground station
Data Transmissions
uplink
downlink
ground Station : Ground Station
Commands
Data Transmissions
operations System : Operations System
backend : Backend
Satellite Beacon
beacon Processor : Beacon Processor
beacon Parser : Beacon Parser
Satellite Beacon
beacon Poster : Beacon Poster
UKW Beacon Data, S-Band Beacon Data
COM Backend : COM Backend
S-Band Beacon Data, UKW Beacon Data
database : Database
COM schema : COM schema
bdd [Paket] General Flowitems[Flowitems hierarchy excerpt]
Data Transmissions
Satellite Beacon
Com Beacon Data
UKW Beacon Data Hardware Only
UKW PA Current

Figure 9-8.: Exemplary data path of the COM Beacon Data

9.2. Structural Analysis

The following text, including the subsections 9.2.1 and 9.2.2, builds on a previous publication by the author (cf. [22]).

The following section describes the application of the schema and guidelines developed in Section 6.4 and Section 6.5. After a short introduction, a selection of questions from Section 6.2.1 are taken and respective queries are explained and performed on the SysML Model of the MOVE-II spacecraft. To conduct these analyses, the SysML Model, which was generated using MagicDraw v19.0 SP4 was transformed into a Neo4j graph database, employing the schema proposed in Section 6.4 and the software implementation explained in Chapter 8. In the following subsections, structural analyses conducted on the MOVE-II model are presented. These analyses build on the questions defined in Section 6.2. Therefore, any question referred to in the following by "Question X" refers to the respective question in Section 6.2.

9.2.1. Data-Oriented Queries

In the following, queries are developed that build on each other and show a rising complexity. They adhere to the questions formulated in Section 6.2 and use the graph query language Cypher (cf. [170]). The queries are shown directly in the thesis as opposed to providing them in the appendix to allow the reader to follow and verify the goal of creating a graph schema that is easy to read and understand (cf. Chapter 2).

Hierarchical Analyses

A list of all subsystems can be generated from the graph database with Query 9-1, which searches for a node with the label `:BLOCK` and `name`-property *MOVE-II satellite* and any further block, which is directly a part of *MOVE-II satellite*, here described with the *subsystem*-variable. Query 9-1 returns the `name` property of all nodes that match the position of the *subsystem*-variable.

```
MATCH (subsystem:BLOCK)-[:IS_PART_OF]->(moveii:BLOCK{name:'MOVE-II satellite'})
RETURN subsystem.name
```

```
subsystem.name
ADCS
CDH
EPS
UHF/VHF
PL
S-Band
Solar Array
STR
```

Query 9-1: Retrieving the parts of a block. *Note:* By adding ∗ after `IS_PART_OF` the query retrieves the parts to unlimited depth.

Query (9-1) shows the solution for Question 1 from Section 6.2.1, regarding the composure of component

X. Query 9-2 answers Question 2, regarding the port types in use over a certain range of equipment. It starts by anchoring the query to the node with a `:BLOCK` label and the name *Ground Station*, asks for any `:PORT` that `:IS_PART_OF` the ground station or of any part of the ground station and queries the port types via the `:IS_OF_TYPE` relations[1]. It returns distinct names of the *porttype* variables and their number of usages, ordering the results in descending order by the number of usages of the port type within the ground station. The result is provided below the query. Note how the declarative definition of relation types fits the declarative language-style of Cypher, allowing for queries that can be understood with little prior knowledge of the language.

```
MATCH (SystemOfInterest:BLOCK{name:'Ground Station'}) <-[:IS_PART_OF*]- (PortInSys:PORT)-[:IS_OF_TYPE]->(porttype)
RETURN DISTINCT porttype.name, count(porttype) ORDER BY count(porttype) DESC
```

porttype.name	count(porttype)
N Connector	8
Ethernet	3
Serial Port	2
USB	2
SMA Connector	2
data port	1

Query 9-2: Retrieve all port types used within a certain range of equipment.

Tracing Data Paths and Analyzing Data Anomalies

One of the first steps in analyzing anomalies is finding all components which potentially participate in the anomaly. Taking a data anomaly as example, any component albeit software or hardware processing the anomaly-holding telemetry is to be found. Instead of consulting a variety of SysML Diagrams to find all components in question, which is cumbersome and prone to errors, Query 9-3 can provide the result, defining any source and target component and the data form in which the telemetry is being transmitted. Note how the use of a parameter allows the reuse of the query for any other telemetry. In this case, the telemetry is *Sidepanel X+ Temperature OW2*, a temperature value on the outside of the spacecraft. Query 9-3 looks for the `:FLOWITEM` with the defined telemetry name and all `:FLOWITEM`s which contain the node with the defined telemetry name. The graph representation to this part of the query is depicted in Figure 9-9.

In the next step, Query 9-3 finds all `:HYPERNODE`s where any of the flowitems `:FLOWS_IN` and calls these *hpn* (see Figure 9-10).

The final step is to look for the pattern of `:BLOCK:INSTANCE` nodes, containing `:PORT`s connected via `:FLOWS` to a *hpn*. An excerpt of this pattern focused on the two elements *Sidepanel X+ Temperature OW2* and *ADCS Sidepanel Data Package x+* can be found in Figure 9-11.

Over the `:FLOWS`-direction, the query differs between *source* and *target* of the flow and consequently returns a table with the results. Of course, the same query can be performed to return a graph instead of a mere table, which might be useful to graphically assist the understanding of the data flow. Figure 9-12 shows the result of this query. Note how at a certain size the graph loses visual interpretability.

[1]The asterisk behind `:IS_PART_OF` defines that an arbitrary number of relations of the type can be followed in this direction.

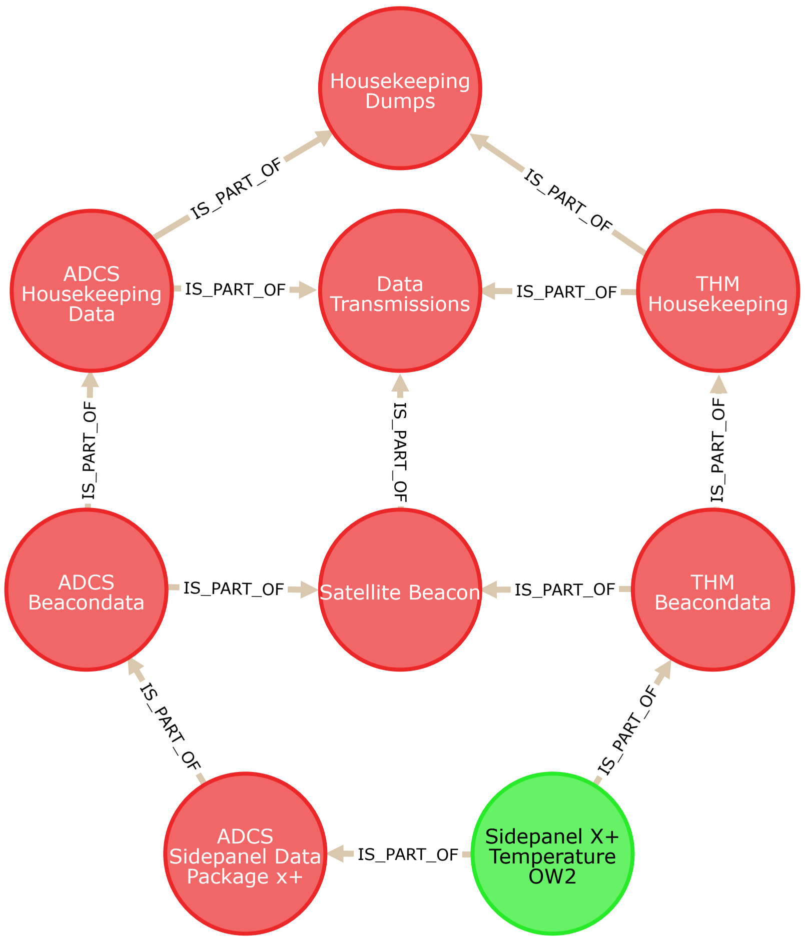


Figure 9-9.: Graph representation to the first part of Query 9-3, retrieving all flowitems carrying the anomalous telemetry. Green: Anomalous telemetry, red: All other flowitems

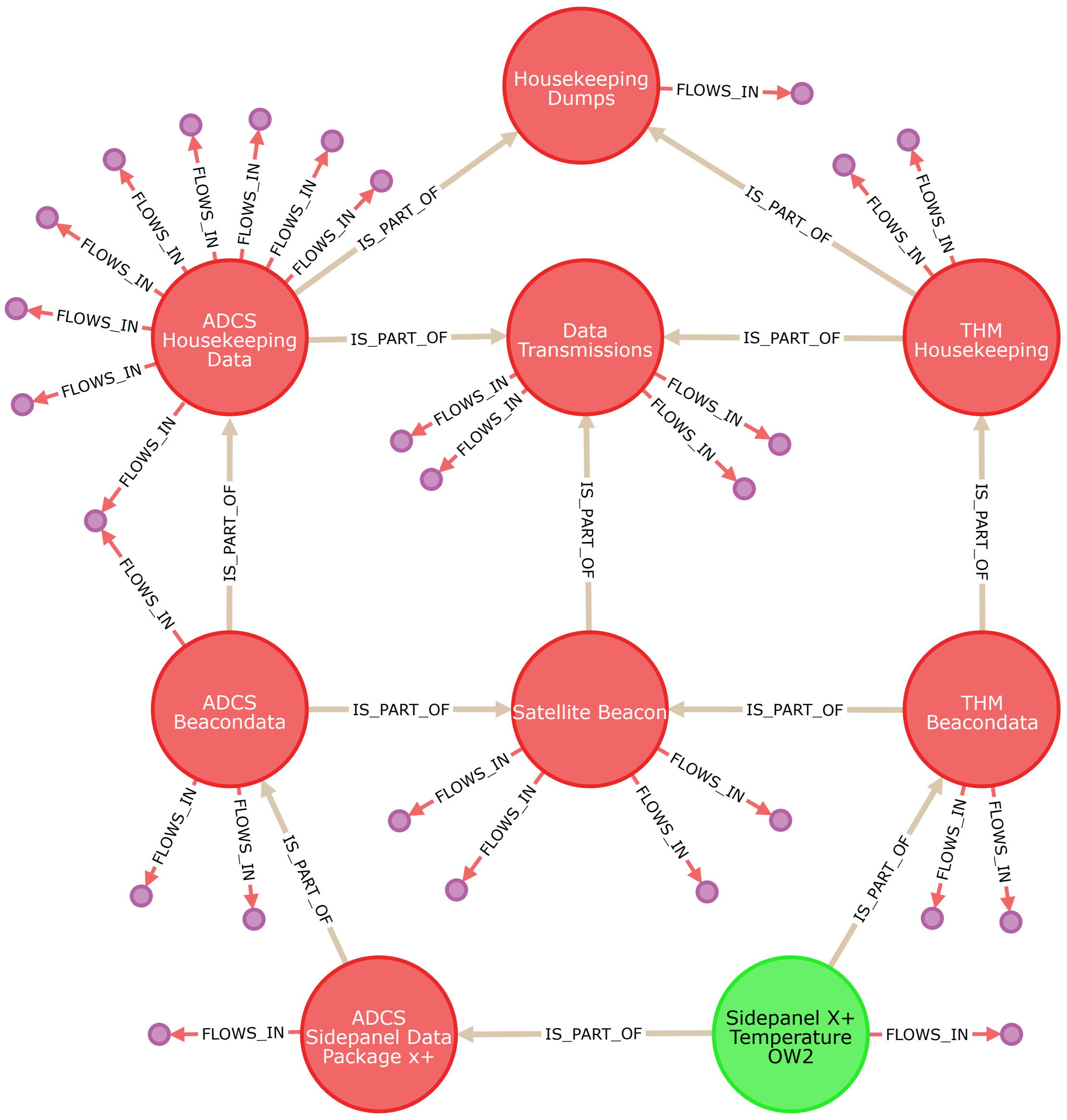


Figure 9-10.: Graph representation to the second part of Query 9-3, retrieving all hypernodes where any of the flowitems of the query's first path flow. Green: Anomalous telemetry, red: All other flowitems, purple: Hypernodes

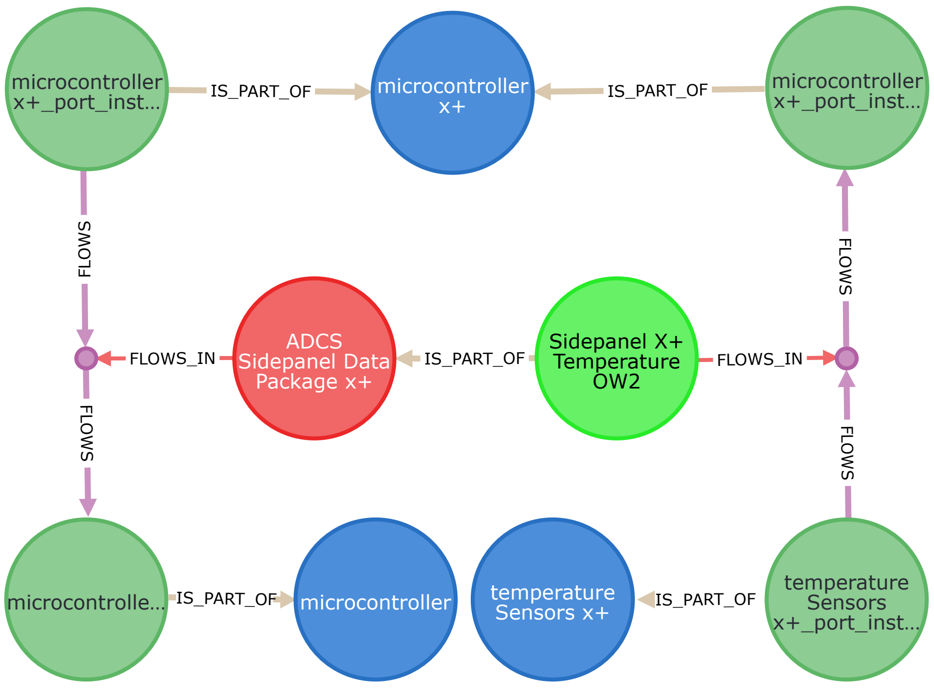


Figure 9-11.: Graph representation of the pattern connecting blocks and flowitems on the example of the flowitems *Sidepanel X+ Temperature OW2* and *ADCS Sidepanel Data Package x+*(final part of Query 9-3).

```
:param searchterm=>'Sidepanel X+ Temperature OW2'
//datapath table
MATCH(telemetry:FLOWITEM {name: $searchterm}) -[:IS_PART_OF*0..]-> (flowitem:FLOWITEM)
WITH flowitem
MATCH(flowitem)-[:FLOWS_IN]->(hpn:HYPERNODE)
WITH flowitem, hpn
MATCH path = (source:BLOCK:INSTANCE) <-[:IS_PART_OF]- (:PORT) -[:FLOWS]-> (hpn) -[FLOWS]-> (:PORT) -[:IS_PART_OF]->
↪  (target:BLOCK:INSTANCE)

RETURN DISTINCT flowitem.name AS processedElement, source.name AS source, target.name AS target ORDER BY
↪  processedElement
```

```
processedElement                source                     target
'ADCS Beacondata'               'beacon Poster'            'ADCS Backend'
'ADCS Beacondata'               'ADCS Daemon'              'beacon Data Collector'
'ADCS Beacondata'               'microcontroller'          'ADCS'
'ADCS Housekeeping Data'        'ADCS Backend'             'ADCS schema'
'ADCS Housekeeping Data'        'ADCS'                     'CDH'
'Sidepanel X+ Temperature OW2'  'temperature Sensors x+'   'microcontroller x+'
[...]
```

Query 9-3: Retrieve the datapath of a certain piece of information within the model. *Note:* The response was cut short here and originally contains an additional 18 lines of response omitted for readability.

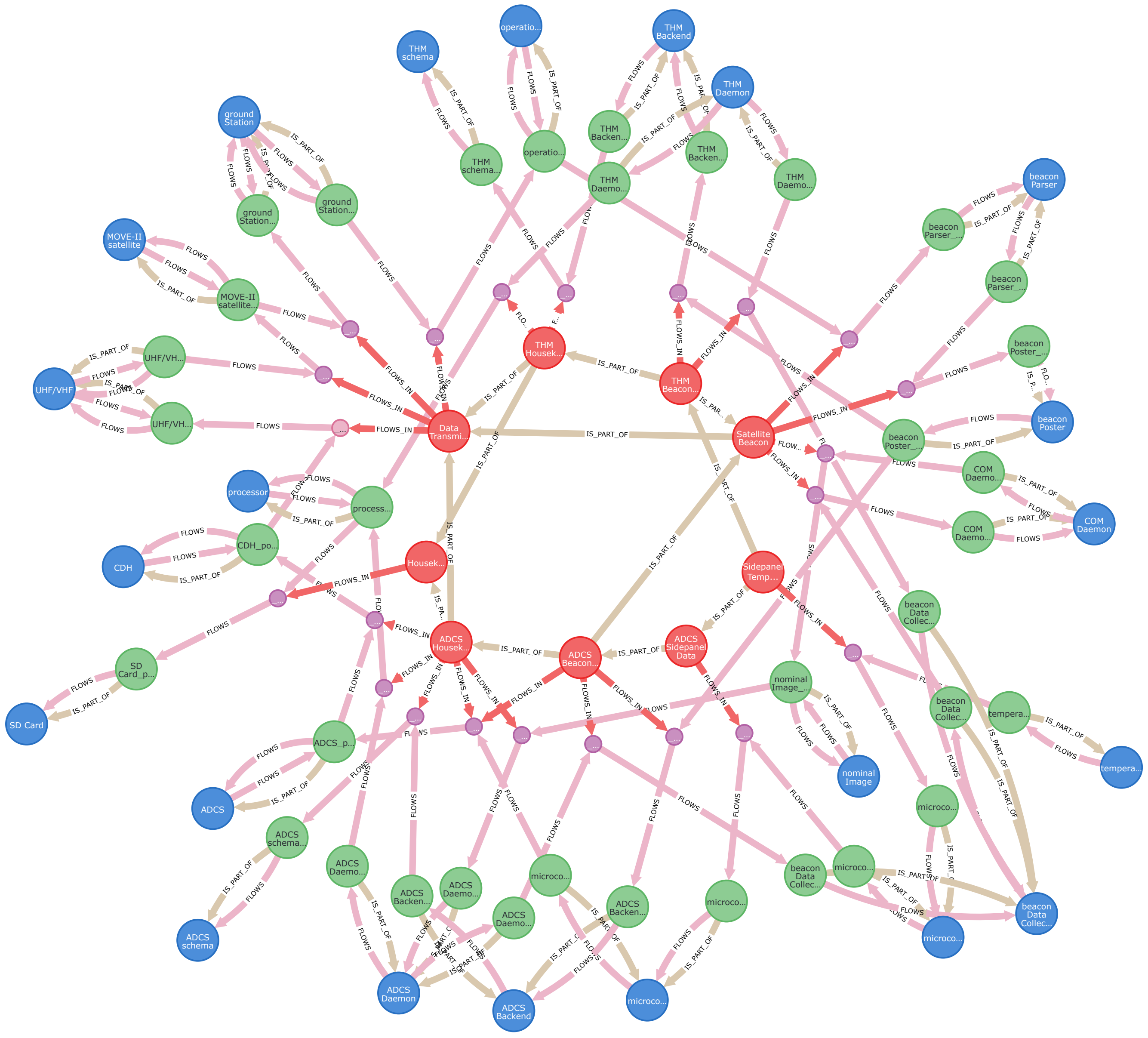


Figure 9-12.: Graph representation of the complete Query 9-3 . Red: :FLOWITEM nodes, green: :PORT nodes, blue: :BLOCK:INSTANCE nodes, purple: :HYPERNODEs

Finding Probable Causes of Anomalies

The next step in finding a data anomaly is to identify other telemetry processed by the components Query 9-3 yielded and checking them for anomalies. A component often either processes any data correctly or none, so while this is just an empirical step, it is an important and useful one. Query 9-4 is an addition to Query 9-3 and yields any other telemetry processed by the same ports.

```
CALL{
    MATCH (telemetry{name: $searchterm})-[:FLOWS_IN]->(hpn) RETURN hpn, telemetry
    UNION
    MATCH (telemetry{name:
↪   $searchterm})-[:FLOWS_IN]->()-[:FLOWS]-()-[:IS_PART_OF]->(component)<-[:IS_PART_OF]-()-[:FLOWS]-(hpn) RETURN hpn,
↪   telemetry
    }
WITH hpn, telemetry
MATCH (suggestion)-[:FLOWS_IN]->(hpn)
  WHERE NOT suggestion = telemetry
return suggestion.name ORDER BY SHORTESTPATH((telemetry)-[:FLOWS_IN|FLOWS*]-(suggestion))
```

suggestion.name
'Sidepanel X+ Temperature OW3'
'Sidepanel X+ Temperature OW1'
'PDM Current ADCS 3V3 1'
'ADCS Sidepanel Data Package x+'
'Sun Vector x+'
'Sidepanel X+ Temperature BMX'
'Gyroscope Data x+'
'Magnetic Field Vector x+'

Query 9-4: Retrieve suggestions for possibly compromised telemetry by checking telemetry which is directly processed by the same components.

This query also builds the basis to answer Question 10 (Given an anomaly on a specific subset of a system's telemetry, which components are most likely to have caused it? Which components can be ruled out?).

The question refers to a scenario, where more than one telemetry value shows an anomaly. To rule out any other component, the query follows the logic "if another input is being processed by the exact same component and port correctly, the error is most likely not in this component." Query 9-5 shows the basic pattern to find components processing the three faulty telemetry packages *$fltm1, $fltm2, $fltm3*, while not processing the healthy telemetry package *$good_tlm*. The result shows a list of ports and components, including their IDs. The query shows, how it is possible to narrow down a list of over 1400 possible suggestions to a mere 6 by applying logic on the graph transformation of the SysML model. Taking a closer look at the proposed components, we find that the ports *p3* and *p4* are proxy ports and therefore no real components. Ruling those out, we end up with one component and three ports as suggested causes of the anomaly. It has to be noted here, that other components or ports could be at fault as well, as not every possible fault path is traceable through the model. However, the query provides a good starting point for the analysis.

Figure 9-13 shows the respective part of the SysML Model, including the item flows, ports and blocks.

```
:param fltm1=>'ADCS Housekeeping Data'
:param fltm2=>'UKW Beacon Data Hardware Only'
:param fltm3=>'S-Band Beacon Data Hardware Only'
MATCH (faulty_tlm1{name:$fltm1}), (faulty_tlm2{name:$fltm2}), (faulty_tlm3{name:$fltm3}), (good_tlm {name:
↪  $good_tlm})
WITH *
MATCH (faulty_tlm1)-[:FLOWS_IN]->()-[:FLOWS]->()-[:IS_PART_OF]->(component)
WHERE EXISTS {(faulty_tlm_2)-[:FLOWS_IN]->()-[:FLOWS]->()-[:IS_PART_OF]->(component)}
    AND EXISTS {(faulty_tlm3)-[:FLOWS_IN]->()-[:FLOWS]->()-[:IS_PART_OF]->(component)}
    AND NOT EXISTS{(good_tlm)-[:FLOWS_IN]->()-[:FLOWS]->()-[:IS_PART_OF]->(component)}
RETURN component.name as suggestion, component.id as id
UNION
MATCH (faulty_tlm1{name:$fltm1}), (faulty_tlm2{name:$fltm2}), (faulty_tlm3{name:$fltm3}), (good_tlm {name:
↪  $good_tlm})
WITH *
MATCH (faulty_tlm1)-[:FLOWS_IN]->()-[:FLOWS]->(port)
WHERE EXISTS {(faulty_tlm_2)-[:FLOWS_IN]->()-[:FLOWS]->(port)}
    AND EXISTS {(faulty_tlm3)-[:FLOWS_IN]->()-[:FLOWS]->(port)}
    AND NOT EXISTS {(good_tlm)-[:FLOWS_IN]->()-[:FLOWS]->(port)}
RETURN port.name as suggestion, port.id as id
```

```
suggestion                     id
SPI level shifter              _19_0_4_9b3028f_1626770558674_761066_49622
CDH_SPI_port_instance          _19_0_4_64d021d_1606812551466_659190_43643 _19_0_4_9b3028f...
processor_port_instance        _19_0_4_9b3028f_1606755365738_580408_43811 _19_0_4_9b3028f...
level shifter_p4_port_instance _19_0_4_9b3028f_1632823793184_527010_45978 _19_0_4_9b3028f...
level shifter_p3_port_instance _19_0_4_9b3028f_1632823808903_57539_46012 _19_0_4_9b3028f...
SPI level shifter_port_instance _19_0_4_64d021d_1609248717945_193014_44390 _19_0_4_9b3028f...
```

Query 9-5: Determine any port or block processing a list of faulty telemetry while not processing healthy telemetry.

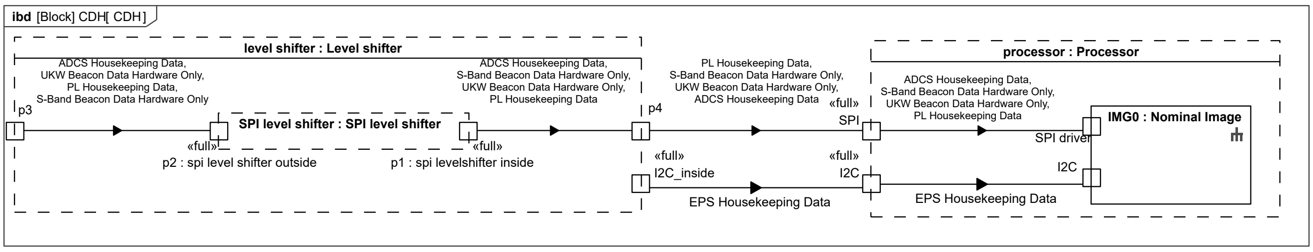

ibd [Block] CDH[CDH]
level shifter : Level shifter
ADCS Housekeeping Data, UKW Beacon Data Hardware Only, PL Housekeeping Data, S-Band Beacon Data Hardware Only
p3
SPI level shifter : SPI level shifter
«full»
p2 : spi level shifter outside
«full»
p1 : spi levelshifter inside
ADCS Housekeeping Data, S-Band Beacon Data Hardware Only, UKW Beacon Data Hardware Only, PL Housekeeping Data
p4
«full»
I2C_inside
PL Housekeeping Data, S-Band Beacon Data Hardware Only, UKW Beacon Data Hardware Only, ADCS Housekeeping Data
EPS Housekeeping Data
«full»
SPI
«full»
I2C
processor : Processor
ADCS Housekeeping Data, S-Band Beacon Data Hardware Only, UKW Beacon Data Hardware Only, PL Housekeeping Data
EPS Housekeeping Data
IMG0 : Nominal Image
SPI driver
I2C

Figure 9-13.: Internal block diagram to Query 9-5

9.2.2. Power-Oriented Queries

The next type of analysis already becomes useful during the design of the spacecraft. In comparison to ground-based systems, robustness against failure is a design goal for any spacecraft, as on-site repairs are usually not possible. Hence, failure propagation is to be kept to an absolute minimum, i.e. the spacecraft's attitude control system, on board computer, power and basic communication systems should not be influenced by a failure in any other part of the system.

The graph schema and modelling guidelines proposed here do not enable the analysis of complex relations such as "the communication system can only work if the spacecraft is pointing to the ground station". The analyses possible with the here proposed graph schema and modelling guidelines center around Question 12 (What happens if component X breaks?).

In case the component X is a physical component, a broken component may trigger an electrical short. Electrical fuses are employed throughout the system to prevent a spread of such an event.

Therefore, one of the tasks to be applied here is to find out which fuse triggers and which components are shut down as well by the fuse. Query 9-7 does exactly this. However, to enable this query, the graph needs to extend by an additional `:FLOWS` relation between any `:PORT` with an incoming or outgoing flow connection and the `:BLOCK` which the port is part of. This is accomplished by the code presented in Query 9-6. Note further that Query 9-7 takes two parameters as input; the *$searchterm* parameter defines which component was shorted, the *$flowlength* parameter defines how many `:FLOWS` connections the query shall follow before it stops searching. It also requires to follow the modelling schema presented in Section 6.5.2, defining *physical flowitems* as a class differing from telemetry in order to enable an accurate tracking of the power path. Limiting the amount of `:FLOWS` connections followed in the query is not technically necessary but increases the performance significantly. As the volatile relations were all built with the property *transient=True*, they can be deleted again after the query is finished.

```
MATCH path = (:HYPERNODE)<-[:FLOWS]-(p:PORT:INSTANCE)-[:IS_PART_OF]->(b:BLOCK)
//limit to ports that do not connect directly to a subcomponent
WHERE NOT EXISTS
↪  {(b:BLOCK)-[:IS_INSTANCE_OF]->(bClass)<-[:IS_PART_OF*]-(subcomponentClass)<-[:IS_INSTANCE_OF]-(subcomponent:BLOCK:INSTANCE)<-[:
MERGE (p)<-[rel:FLOWS]-(b)
ON CREATE SET rel.transient = true

MATCH path = (:HYPERNODE)-[:FLOWS]->(p:PORT:INSTANCE)-[:IS_PART_OF]->(b:BLOCK)
//limit to ports that do not connect directly to a subcomponent
WHERE NOT EXISTS
↪  {(b:BLOCK)-[:IS_INSTANCE_OF]->(bClass)<-[:IS_PART_OF*]-(subcomponentClass)<-[:IS_INSTANCE_OF]-(subcomponent:BLOCK:INSTANCE)<-[:
MERGE (p)-[rel:FLOWS]->(b)
ON CREATE SET rel.transient = true
```

Query 9-6: Code to create volatile [:FLOWS] relations necessary to apply the shortest path algorithm in Query 9-7.

Query 9-7 can also be tuned to show any telemetry values that are not processed correctly anymore due to the short by adding a new `MATCH`-clause based on the variable *shortedcomponent*.

```
    :param searchterm => 'real Time Clock'
Call {
  MATCH p = (rtc{name:$searchterm}) <-[:IS_PART_OF]- (port) <-[:FLOWS]-()<-[:FLOWS_IN]- (shortedvoltage)
↪ -[:IS_OF_TYPE]-> (pfi{name:'physical flowitem'})
  WITH shortedvoltage
  MATCH (fusetype {name:'electrical fuse'}) <-[:IS_OF_TYPE]- (fuse) <-[:IS_INSTANCE_OF]- (fuseinstance)
↪ <-[:IS_PART_OF]- (fuseportinstance) -[:FLOWS]-> (hpn) <-[:FLOWS_IN]- (shortedvoltage)
  RETURN fuseinstance, shortedvoltage ORDER BY length(SHORTESTPATH((fuseinstance) -[:FLOWS*]- (rtc))) LIMIT 1}
WITH *
MATCH p=(fuseinstance) -[:FLOWS*1..$flowlength]-> (hpn:HYPERNODE) <-[:FLOWS_IN]- (shortedvoltage)
WITH hpn, fuseinstance
MATCH (hpn) -[:FLOWS]-> () -[:IS_PART_OF]-> (shortedcomponent)
WHERE NOT fuseinstance.id = shortedcomponent.id
RETURN shortedcomponent.name
```

shortedcomponent.name
'current Limiter SD Cards'
'SD Card'
'FRAM'
'flash'
'GPS'
'processor'
'real Time Clock'
'level shifter'

Query 9-7: Find components affected by an electrical short.

9.2.3. Requirements Oriented Queries

Section 6.2.3 specifies a set of questions related to the handling of requirements and use cases. This fits well with the focus of this thesis into the later phases of the mission life cycle. Verification and validation (i.e. does the system meet the specifications and does it perform its intended tasks) require a thorough understanding of the relations between requirements, between components and requirements and between use cases, stakeholders and requirements.

Querying all Requirements Effective on a Component

Query 9-8 shows all requirements that have to be satisfied by the block defined under *$searchterm* either directly or indirectly. In this case, the component "2SMARD 1" was specified as search term, which is one of the two shape memory alloy based deployment actuators of the MOVE-II spacecraft.

By just slightly altering Query 9-8, one can get the full list of all requirements the component has to satisfy, either directly, or because they have to be satisfied by the assemblies the component is part of. Query 9-9 shows the query's code and the respective result for the component "2SMARD 1". The original response by the system yielded 45 requirements, albeit only 22 are listed here. The assumption made in Section 6.5.4 can be shown to be valid, as most requirements that are to be satisfied by an assembly also affect its parts.

Similar to Question 24, Question 25 queries the same pattern in a reverse manner. To get a full list of all components that have to comply with a requirement, its sub-requirements can be included in the analysis. Query 9-10 provides the respective code for the requirement.

```
:param searchterm=>'2SMARD 1'
MATCH (block:BLOCK{name:$searchterm})-[:SATISFIES]->(r:REQUIREMENT)
RETURN r.title as Title,  r.Text as Text
```

Title	Text
SYS-11.1	All subsystems shall stay in the operational limits of the EPS.
STEX-DLR-02.2	The MOVE-II deployment system for the solar panels shall show a technical progress compared to the First-MOVE system.
STR-10	STR shall ensure the deployment of all of MOVE-II's deployables.
STR-10.4	The deployment mechanisms shall be designed in a way that ensures redundant deployment of MOVE-II's UHF/VHF antennas.
STR-10.2	The deployables shall be deployed no sooner than 30 minutes after ejection from the launch vehicle.
STR-10.3	The deployment mechanisms shall be resettable.
% STR-09	STR (Deployment Systems) shall comply with the power budget of MOVE-II.

Query 9-8: Retrieve all requirements directly related to a component

Requirements Affected by a Change Request

Question 26 details a common question in any project. When a requirement ought to be changed, finding other requirements and components are affected by the change request is of great importance. Therefore, Query 9-11 provides a list of all directly affected requirements to a requirements change. As example the requirement PL-06.1 (*The sun angle measurement accuracy should be <2°and has to be <8°.*) is analyzed.

Query 9-12 provides the requirements affected via one of the components related to the query. Comparing the results of Query 9-11 and 9-12, the second query yields 3 more requirements, two of which start with a %-sign, used to mark inactive requirements in MOVE-II. Therefore, the only active requirement that was added by the query is ADCS-03.1.

Going a step further, Query 9-13 refers to Question 29, return all stakeholders that have to be considered for the requirements change process.

Question 32, asking for requirements that cannot be traced to any stakeholder becomes relevant already in the early phases of a project. Apart from finding requirements without any justification for existence, it also helps to identify deficiencies in the SysML Model. Query 9-14 provides the necessary Cypher syntax to answer this question. Adding the clause `AND NOT req.title starts with '%' RETURN collect(req.title)` at the end of the first line filters all deactivated requirements, resulting in an empty response.

Question 34, asking after the effects of a stakeholder leaving the project is also more relevant to earlier project phases. The queries 9-15 and 9-16 provide the respective syntax. Figure 9-14 shows the graph response to Query 9-16. Although a total of 21 requirements are possibly obsolete, only one component shares the fate: the Release Switch Connection of the Electrical Power System. Analyzing the requirements that are listed as possibly obsolete, a series of requirements linked to the CubeSat Design Specification stick out. Thus, the result of the query can be read as follows: In case the DLR left the project as a stakeholder, it is no longer required to keep with the CubeSat Design Specification. Thereby components that are only necessary to fulfill the CubeSat Design Specification may no longer be necessary. Query 9-15 makes use of another feature of graph databases; the flexible data schema. By adding the property "_active" which discerns between stakeholders still in the project and those that left the project, the graph itself stays the same while new and more complex analyses can be run.

```
:param searchterm=>'2SMARD 1'
MATCH path = (component:BLOCK{name:$searchterm}) -[:IS_PART_OF*0...]->
↪  (comp_and_assemblies:BLOCK)-[:SATISFIES]->(r:REQUIREMENT)
with  length(path) as distance
RETURN comp_and_assemblies.name as ComponentName, r.title as Title, r.Text as Text ORDER BY distance
```

	Component Name	Requirement ID	Requirement Text
1	2SMARD 1	% STR-09	STR (Deployment Systems) shall comply with the power budget of MOVE-II.
2	2SMARD 1	STEX-DLR-02.2	The MOVE-II deployment system for the solar panels shall show a technical progress compared to the First-MOVE system.
3	2SMARD 1	STR-10	STR shall ensure the deployment of all of MOVE-II's deployables.
4	2SMARD 1	STR-10.2	The deployables shall be deployed no sooner than 30 minutes after ejection from the launch vehicle.
5	2SMARD 1	STR-10.3	The deployment mechanisms shall be resettable.
6	2SMARD 1	STR-10.4	The deployment mechanisms shall be designed in a way that ensures redundant deployment of MOVE-II's UHF/VHF antennas.
7	2SMARD 1	SYS-11.1	All subsystems shall stay in the operational limits of the EPS.
8	STR	% STR-01.15 A	STR shall provide at least one (limit) and should provide two deployment switches located on the designated standoff.
9	STR	%STR-10.1	The deployables shall meet the exterior dimensions requirements of the CDS Rev.12.
10	STR	PL-01	No specular or diffuse reflections from the satellite shall reach the SCs surface.
11	STR	STEX-LRT-02	The MOVE-II satellite bus shall be adaptable to different payloads.
12	STR	STEX-PL-01.1	No specular or diffuse reflections from the satellite shall reach the solar cell assembly's surface.
13	STR	STR-00.1	The STR system shall provide the mounting of the satellite bus.
14	STR	STR-00.2A	The STR shall comply with the Rev.13 of the CDS.
15	STR	STR-00.3	STR shall ensure the structural integrity of the satellite during launch and on-orbit operations.
16	[...]	[...]	[...]
17	MOVE-II satellite	%SYS-04.2	The MOVE-II satellite shall not exceed the volume constraints defined by the CDS Rev.12.
18	MOVE-II satellite	ADCS-07	Residual magnetic moments shall be minimized.
19	MOVE-II satellite	ADCS-07.1	The residual magnetic moments of MOVE-II shall be minimized to a value that enables the ADCS to perform with no more than 0.25W (limit value) power consumption due to residual magnetic moments. It should not have to use more than 0.1W (set-point value).
20	MOVE-II satellite	CDH-05	In the satellite's nominal operation state, all subsystem interaction and the commanding shall be conducted by the CDH system.
21	[...]	[...]	[...]
22	System MOVE	STEX-DLR-02	MOVE-II shall advance the development of Smallsat-related space technology, with a significant progress compared to First-MOVE.
23	System MOVE	SYS-06	The MOVE-II satellite shall be able to be operated from the LRT Ground Station.
24	System MOVE	SYS-12	Whenever applicable, hard- and/or software developed for the First-MOVE mission shall be reused and/or adapted in order to minimize development efforts.

Query 9-9: Retrieve all requirements directly related to a component or any assembly it is part of

```
:PARAM reqTitle =>'ADCS-01A'

MATCH path =
↪  (r:REQUIREMENT{title:$reqTitle})<-[:IS_PART_OF*0..]-(req)<-[SATISFIES]-(b:BLOCK)<-[:IS_PART_OF*0..]-(block)
RETURN block.name as BlockName, req.title as RequirementID ORDER BY length(path), block.name
```

	Component Name	Requirement ID
1	ADCS	ADCS-01A
2	3v3 connector 1	ADCS-01A
3	3v3 connector 2	ADCS-01A
4	3v3 in 1	ADCS-01A
5	3v3 in 2	ADCS-01A
6	5v connector 1	ADCS-01A
7	5v connector 2	ADCS-01A
8	5v in 1	ADCS-01A
9	5v in 2	ADCS-01A
10	Mainpanel	ADCS-01A
11	SPI	ADCS-01A
12	Sidepanel X+	ADCS-01A
13	Sidepanel X-	ADCS-01A
14	Sidepanel Y+	ADCS-01A
15	Sidepanel Y-	ADCS-01A
16	Actuator	ADCS-01A

Query 9-10: Retrieve all components satisfying a requirement

```
:PARAM searchterm=>'PL-06.1'
MATCH(req_changing:REQUIREMENT{title:$searchterm})-[:IS_TRACED_FROM*0..1]-(req_affected)
RETURN  req_affected.title as ID, req_affected.Text ORDER BY req_affected.title
```

Requirement ID	Requirement Text
ADCS-03.2	The ADCS must provide sun angle information with an accuracy of 8°during payload measurements and should provide sun angle information with an accuracy of 2°.
PL-06.1	The sun angle measurement accuracy should be <2°and has to be <8°.
PL-06A	The sun angle has to be measured during payload measurements.

Query 9-11: Requirements directly affected by a change request

```
MATCH (req_change:REQUIREMENT {title:$searchterm})<-[:SATISFIES]-(element)-[:SATISFIES]->(req_affected:REQUIREMENT)
WITH DISTINCT req_affected, req_change, collect(element.name) as Element
RETURN req_affected.title AS ID, req_affected.Text AS Text,  Element ORDER BY req_affected.title
```

Requirement ID	Requirement Text
%ADCS-03	The ADCS shall provide an attitude knowledge of at least 2°in each direction.
%PL-06	The sun angle measurement accuracy should be <2°and has to be <8°.
ADCS-03.1	THE ADCS must provide an attitude knowledge of at least 5°in each direction and should provide an attitude knowledge of at least 1°in each direction.
ADCS-03.2	The ADCS must provide sun angle information with an accuracy <8°during payload measurements and should provide sun angle information with an accuracy of <2°.
ADCS-03A	The ADCS shall provide an attitude knowledge.
PL-06A	The sun angle has to be measured during payload measurements.

Query 9-12: Requirements affected by a change request via a common component

```
MATCH (req_change:REQUIREMENT {title:$searchterm})-[IS_TRACED_FROM*0..]->(u:USECASE)-[:IS_TRACED_FROM]->(a:ACTOR)
RETURN DISTINCT a.name AS Stakeholder, u.name AS UseCase
```

Stakeholder	Use Case
Student Team	Solar Cell Degradation Measurement
Payload Provider	Solar Cell Degradation Measurement

Query 9-13: Retrieving Stakeholders possibly affected by a requirements change request

```
MATCH(req:REQUIREMENT)  WHERE NOT EXISTS {(req)-[:IS_TRACED_FROM*]->(:ACTOR)}
RETURN collect(req.title)
```

```
[%ADCS-03, %ADCS-01, %ADCS-06, %ADCS-09,%ADCS-08, %ADCS-00.1, %CDH-06, %ADCS-00.2, %SYS-11,%SYS-03,%SYS-04,
%COM-07, %COM-00.3, %PL-07.1]
```

Query 9-14: Retrieving all requirements not linked to a stakeholder or use case

```
:PARAM leavingStakeholder=>'DLR'
MATCH (a:ACTOR) SET a._active=TRUE
MATCH (lstkh:ACTOR{name:$leavingStakeholder}) SET lstkh._active = FALSE
MATCH(stk:ACTOR{name:$leavingStakeholder})<-[:IS_TRACED_FROM*]-(u:USECASE|REQUIREMENT) WHERE NOT EXISTS
↪  {(u)-[:IS_TRACED_FROM*]->(stk2:ACTOR{active:TRUE})}
return u.name, labels(u)
```

Use Case
Show Responsible Use of Resources
Further Student Education
Advancement of University based Small Spacecraft Development

```
MATCH path = (stk:ACTOR{name:$leavingStakeholder})<-[:IS_TRACED_FROM*]-(r:REQUIREMENT) WHERE NOT EXISTS
↪  {(r)-[:IS_TRACED_FROM*]->(stk2:ACTOR{active:TRUE})} AND NOT r.title STARTS WITH '%'
        set r._plength=length(path)
        WITH DISTINCT r
RETURN  r.title  AS ID, r.Text ORDER BY r._plength, r.title
```

Requirement ID	Requirement Text
STEX-DLR-01	MOVE-II shall be a 1-Unit (1U) CubeSat according to the CubeSat Design Specification (CDS).
STEX-DLR-03	The project shall comply to the Space Debris Mitigation guidelines provided by the DLR (Space Debris Mitigation, DLR-RF-PS-001, Issue 7.0, August 2012).
STEX-DLR-04	The project duration shall be three years, including six months of operations.
STEX-DLR-05	MOVE-II shall facilitate hands-on experience for students of all fields, administratively supported by the LRT.
STEX-DLR-06	The project costs funded by the DLR shall not exceed [redacted].
STEX-DLR-07	Yearly update reports shall be sent to the DLR.
STEX-DLR-08	The MOVE-II mission shall comply with all relevant federal regulations.
SYS-03A	The MOVE-II satellite shall comply with the 13th Revision of the CDS.
SYS-04A	The MOVE-II satellite shall be a 1U CubeSat as defined by the CDS Rev.13.
ADCS-00.2A	The ADCS shall comply with the Rev.13 of the CDS.
CDH-00.2	The CDH system shall comply with the Rev.13 of the CDS.
COM-00.3A	The COM system shall comply with the Rev.13 of the CDS.
EPS-00.2A	The EPS shall comply with the Rev.13 of the CDS.
PL-00.2	The Payload shall comply with the Rev.13 of the CDS.
STR-00.2A	The STR shall comply with the Rev.13 of the CDS.
SYS-04.1	The MOVE-II satellite total mass shall not exceed 1.33 kg.
SYS-04.2A	The MOVE-II satellite shall not exceed the volume constraints defined by the CDS Rev.13.
COM-06B	All transmitters shall wait to transmit a minimum of 45 minutes after the CubeSat's deployment switches are activated from Deployer ejection.
EPS-07A	The EPS shall be able to separate power from the system for the time between final integration and ejection of the satellite.
STR-01A	STR shall meet the exterior dimensions requirements of the CDS Rev.13.
EPS-07.1	Any leakage flows in case of separated power shall be minimized.

Query 9-15: Possibly obsolete use cases and requirements after a stakeholder's exit

```
MATCH path = (stkh{name:$leavingStakeholder}) <-[:IS_TRACED_FROM*]-()<-[:SATISFIES]-(element) WHERE NOT EXISTS
↪  {(element)-[:SATISFIES]->()-[:IS_TRACED_FROM*]->(stk2:ACTOR{active:TRUE})}
RETURN DISTINCT path
```

Query 9-16: Possibly obsolete components after a stakeholder's exit

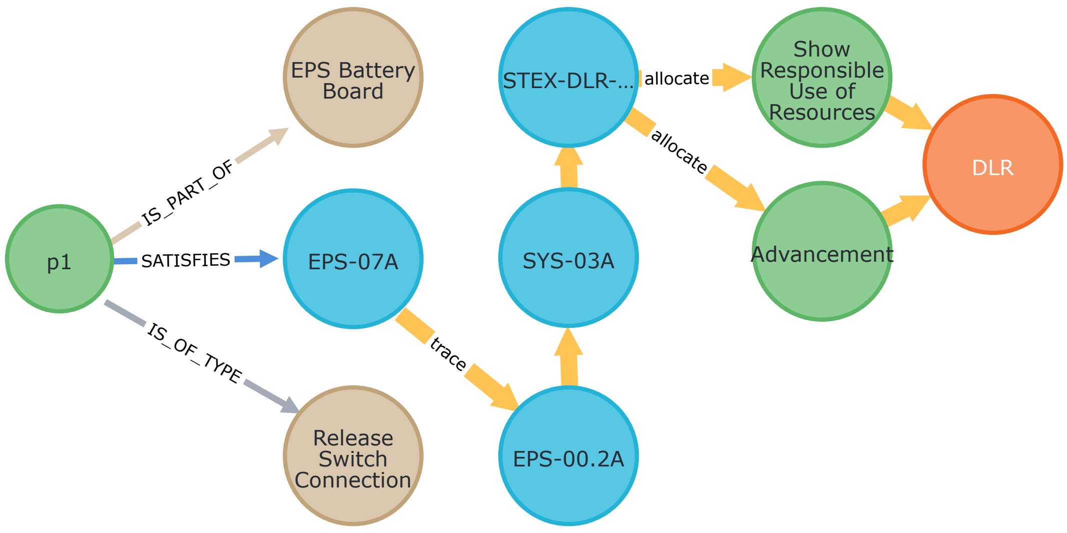


Figure 9-14.: Graph reply of Query 9-16, returning possibly obsolete components after a stakeholder's exit

9.2.4. Summary

Referencing the questions from Section 6.2, Section 9.2 shows the analysis capabilities of translating a consistent SysML model into a graph database. The descriptive nature of the query-language Cypher paired with the phrasing the labels accordingly allows even readers with no previous experience to understand what a query does.

From the questions defined in Section 6.2 the majority of questions regarding the structure of SysML models have been addressed, same as questions regarding requirements, use cases and stakeholders. Analyzing the behavioral aspects of SysML is omitted here for the sake of brevity.

9.3. Analyzing the MOVE-II Informational Environment

The following section gives an overview of the MOVE-II informational environment. As a rather large development project for a CubeSat with at times more than 100 students actively contributing, 3 years of development and testing and another 3 years of operations, the informational environment is diverse. The volunteer nature of the project further contributes to the challenges described in Chapter 1 and Chapter 2 regarding completeness, ambiguity, accuracy and possible communication paths.

9.3.1. Overview over the MOVE-II Informational Environment

Over the years, a variety of systems were used to organize information in the MOVE-II project. Starting with an FTP file-server, and an instance of the issue- and project management system "Redmine", the project progressed to another network access storage, an instance of the team collaborator "Slack", the card-based issue tracking system "Trello" and finally a central collaboration management software (in this case "GitLab"). Additionally, regular reviews took place for which formal documentation was produced and e-mails were used to communicate with external stakeholders.

Employing a system such as described in this thesis to aggregate data that allows to discern personal information would require the formal agreement of every participant. At over 200 this has little chance of success, especially as some participants cannot be reached anymore. This limits the information to be analyzed here to the content that was originally intended as pure documentation. This includes publications and theses written on MOVE-II as well as the documentation on the project, whether for internal use or external review.

The documentation encompasses all formal review documents as well as any documentation written in the redmine-wiki or the collaboration management system, the system documentation of the satellite as well as all formal test reports. This includes the documentation of all 143 code repositories.

9.3.2. Connectivity between Informational Meta System and SysML Model

Identifying connections between the SysML model and the information imported into the graph is a delicate task where no general recipe may be found that is applicable to all and any projects as it strongly depends on the customs of communication within a project.

Most technical projects have a set of technical terms that describe their system. These terms should be defined or at least used in the SysML model. These terms are used to allow others to understand what part of the system is the topic of an information.

The goal here can not be to identify all information ever shared concerning a certain element of the system,

```
MATCH(sysml:SYSML), (m:INFO)
WHERE size(sysml.name)>3 AND apoc.text.clean(m.content) contains apoc.text.clean(sysml.name)
MERGE (sysml)<-[rel:MENTIONS]-(m)
set rel.transient=TRUE
```

Query 9-17: Query to relate MOVE-II informational content with the SysML Model

as this would require the proper name to be used every time. If a discussion is just being continued digitally after the topic is already set over a direct conversation, this may not be possible anymore. Another problem occurring with relying on the names of system elements being used is the usage of acronyms and synonyms. It is possible however, to introduce frequently used terms to the SysML Model as synonyms. This can for example be done via generalizations in the model or a list of synonyms used for cypher-queries.

Relating the information in case of the MOVE-II model was performed by Query 9-17. The call of the function `apoc.text.clean` strips all formatting from a string. This induces small mishaps as for example "Battery Port +" and "Battery Port -" are treated as the same object. The minimum length of name was set to four characters, as anything less would result in relations being made to ports with names such as "in" and "out" throughout all documents to every occurrence of the words "in" and "out".

A prerequisite to Query 9-17 is a unified property `content` which holds the textual content of the `:INFO`-node. Another unified property is the `time`-property, which holds conversions of the time of creation for all information transferred into an `:INFO`-node.

9.3.3. Informational Analyses of the MOVE-II System

After importing the documentation of all MOVE-II repositories as well as a list of theses, papers and other scientific publications on the project, the graph measures a total of 2564 informational elements. Counting the individual commits of the projects as well, the total rises by 60264 to 62828 nodes for the informational environment. Each of these nodes contains information that was added by hand at some time during the project. Opposed to this stand around 2250 nodes created from the SysML Model.

Reducing the model to nodes with a name with more than three characters, the SysML Model's graph still contains 1878 nodes.

A total of 64807 relations were built by Query 9-17 between information and SysML Model. As the model consists of several namespaces and often contains instances with similar names to the blocks they instantiate, only a share of these relations are meaningful, as others simply duplicate relations to elements with very similar or equal names. A third of all commits could directly be related to a node in the SysML Model. Analyzing the amount of connections of each node shows that the most relations are built to SysML elements with too generic names, such as "file" "system" "data" or "Time". Subtracting the relations of elements with duplicate names and generic names reduces the amount of relations by 61%[2]. Table 9-1 provides a more detailed analysis. Similar numbers for blocks and instances are to be expected, as a block and its instance often carry similar names and match on the same texts. The low amount of matches for flowitems is not unexpected, as the names applied to the telemetry packages and flowitems in the model are not in common use. Names such as "SideYMinus Magnetic Field Vector" are simply too unyielding.

While the numbers show that enhanced methods such as natural language processing may be beneficial, an

[2]For the full list consult the digital appendix "amount_of_mentions_per_sysml_element_incl_false_positives.xlsx"

Table 9-1.: Statistics on mentions of SysML elements based on their node type

No. mentions	0	1 or more	4 or more	11 or more	over 21	51 or more	101 or more	200 or more
Blocks	59%	41%	25%	20%	16%	10%	8%	6%
Instances	60%	40%	25%	20%	17%	13%	9%	7%
Flowitems	79%	21%	9%	4%	4%	3%	2%	2%

```
MATCH (component:TEC{name:'RawRESQ'})<-[:MENTIONS]-(i:INFO)
RETURN i.time, labels(i), i.name ORDER BY i.time DESC
```

Query 9-18: Query to retrieve all `:INFO`-nodes connected to a certain component

overall 38% of all blocks being traced into the documentation is a promising outlook and already allows testing queries for information retrieval on graph databases. Additionally, it has to be considered that the information inserted into the database is only a part of the available informational environment of MOVE-II. Due to privacy concerns only information intended for future reference and publication from the start of the project was added to the database. Therefore, no emails, no chat messages, no tickets or merge requests etc. were added as this would require the consent of hundreds of individuals. Similarly interesting to the amount blocks with a connection to the informational environment is the reverse question on the share of `:INFO`-nodes where a connection to the SysML Model could be established. Out of 61964 `:INFO`.nodes, 22105 or 36% could be matched to the SysML Model.

In the following, a selection of the questions defined in Section 7.1 are transformed into queries. The numbering of questions therefore refers to the questions in Section 7.1. Question 1 refers to the most basic situation of retrieving all documentation relevant to a certain component.

Query 9-18 returns a list of all 225 `:INFO`-nodes that mention the term "RawRESQ", which is a specific software component of the MOVE-II Onboard-Computer. The amount of data that can be found for the component raises another problem: Once the data is gathered, functions to prioritize and filter the data are necessary. This can be done by multiple means, such as filtering the file-ending or omitting any commits that contain the term "RawRESQ".

Query 9-19 shows two possibilities for filtering the information retrieved in Query 9-18. It addresses Question 2 and Question 3, which ask for information from a specific timeframe and of a specified type. Specifying relation type and direction in the `MATCH`-Clause allows answering Question 3, regarding information on components and their subcomponents. The `WHERE`-Clause in line 2 and 3 specifies the filters.

```
1  MATCH (component:TEC{name:'RawRESQ'})-[*0..1]-(:TEC)<-[:MENTIONS]-(i:INFO)
2  WHERE NOT component:COMMIT AND
3   i.time>"2021-01-01T00:00:00.000+01:00"
4  RETURN i.time, labels(i), i.name ORDER BY i.time DESC
```

Query 9-19: Options for filtering information from Query 9-18 and widening the search to any connected components.

```
MATCH (i:INFO{id: '21ed6e28129001fb69462b8ab253c5a52bac94c4'})-[:MENTIONS]->(t:TEC)<-[:MENTIONS]-(newinfo)
WHERE newinfo.time>i.time
RETURN newinfo.time, newinfo.name,t.name
```

newinfo.time	newinfo.name	t.name
2018-03-09T13:01:28.000+01:00	commit:Rename Beacon Parser to Beacon Processor	Beacon Parser
2018-03-25T23:29:31.000+03:00	commit:Rename beacon_processor to beacon_parser	Beacon Parser
2018-03-05T16:34:23.000+01:00	commit:add system design documentation	Beacon Parser
2020-10-13T11:17:40.000+02:00	Week_01_2018-12-03_2018-12-09.md	Beacon Parser
2017-11-15T21:07:39.000+01:00	commit:Add PL encryption pattern for housekeeping and beacon parser.	Beacon Parser
2018-03-09T13:01:28.000+01:00	commit:Rename Beacon Parser to Beacon Processor	beacon Parser
2018-03-25T23:29:31.000+03:00	commit:Rename beacon_processor to beacon_parser	beacon Parser
2017-11-15T21:07:39.000+01:00	commit:Add PL encryption pattern for houskeeping and beacon parser.	beacon Parser
2018-03-05T16:34:23.000+01:00	commit:add system design documentation	beacon Parser
2020-10-13T11:17:40.000+02:00	Week_01_2018-12-03_2018-12-09.md	beacon Parser

Query 9-20: Query aiding the search of more up-to-date information

```
MATCH (i:INFO) WHERE apoc.text.clean(i.content) CONTAINS apoc.text.clean("GETESTETE FUNKTIONEN UND KOMPONENTEN")
set i.testdoc = true
```

Query 9-21: Filtering Test Documents

Determining whether a certain piece of information is outdated (Question 6 in Section 7.1) can be aided by the construct in Query 9-20. The results of the query require proper interpretation, however. First, not every link created by Query 9-17 which relates `:INFO`-nodes with `:TEC`-nodes is valid. Second, the fact that new information on a topic exists does not immediately invalidate previous information. However, the list of informational elements retrieved by Query 9-20 provide a good starting point for the assessment.

The last question in Section 7.1 refers to finding any test documents regarding a requirement. This can be done in multiple steps. Query 9-21 provides an extra property for test documents: `testdoc = true`. The query relies on a string in the common template of all formal test documents. Query 9-22 then looks for any test documents with relations to technical components that satisfy the requirement. In this show-case application, the requirement chosen is satisfied by the deployment mechanism on MOVE-II (*STR-10: STR shall ensure the deployment of all of MOVE-II's deployables*).

```
MATCH (i:INFO{testdoc:true})-[:MENTIONS]->(t:TEC)-[:SATISFIES]->(r:REQUIREMENT{title:'STR-10'})
return i.name
```

i.name
"00021-01-TCa-STR_ANT_SIDE_FLAP_2SMA.pdf"
"00020-01-TCa-STR_ANT_SIDE_FLAP_2SMA.pdf"
"00022-01-TCa-STR_ANT_SIDE_FLAP_2SMA.pdf"
"00023-01-TCa-STR_ANT_SIDE_FLAP_2SMA.pdf"
"00023-01-TPr-STR_ANT_SIDE_FLAP_2SMA.pdf"
"170308_testprot_STR_2smard_sw_test.pdf"

Query 9-22: Finding Test Documentation to a specific Requirement

9.4. Evaluation

The showcase application of the graph schema on MOVE-II shows a wide range of capabilities for the schema. The adaptable nature of cypher-queries allows projecting complex patterns of thought onto the SysML Model's graph. Implementing the connection with the information management systems of the spacecraft allows to quickly draw conclusions on complex situations and shows, that the SysML Model of the system under development is a viable basis for an enterprise search engine connecting all information in the development.

Still, a more seamless integration would have been possible if some difficulties experienced in the two sections above had been anticipated.

- Using the function `apoc.text.clean` allows to match texts with slightly different typography that any human would still interpret as the same text. Avoiding false positives requires writing out certain terms instead of using mathematical symbols that are filtered by the function. However, this would not only go for the SysML Model, but also for the every-day communication. Implementing such a guideline would require the entire team to commit to it, which is unlikely. Instead, creating a further block as generalization, that holds neither of the specifications "+" or "-" might be beneficial. The results can afterwards still be filtered again, once the information nodes are found that relate to these elements.
- The names used in the SysML Model should be meaningful and not used in everyday speech. Over 20 000 false positive connections were created by Query 9-17 only based on words such as "time" or "power".
- Additionally, a better algorithm to connect information with `:TEC`-nodes is necessary in order to link all information to the SysML Model, as the current strategy of simply comparing strings is not able to find all references. Besides, a way to treat the renaming of a technical component is required. The modeling method described in the modeling guidelines in Section 6.5 do not require modeling any old denominators of system components, further limiting the amount of identified relations.

In the following, the application of the SysML Graph Schema shall be evaluated on two further systems: The EMPS and a life-support system for a lunar base.

The intent behind these use cases is to show the general applicability of the schema. The EMPS is a payload developed at the Chair of Astronautics under my supervision and flying on the Australian M2 satellite, while the life-support system is based on research of my colleagues Daniel Kaschubek, Laura Grill and Matthias Killian (cf. [171], [172]).

10. Extensibility Analyses of the SysML Graph Schema

After Chapter 9 showed the applicability of the graph schema on small spacecraft projects, the following two showcases demonstrate its applicability on two further models and thereby show the applicability on a wider range of systems than only small spacecraft. The first is the model of a spacecraft equipment with focus on electrical and thermal flows and their effects on one another. The second model's focus lies on chemical flow systems. In difference to the MOVE-II study, the informational environment of the two systems is not part of the analysis.

The EMPS is an electrical power system, for which the relations between different power flows and the relations of power flows, temperatures and their respective measurements are of interest. The model describes a real system, that has been orbited in Q1 2021 and is since being operated by the University of New South Wales, Australia.

The second extensibility case is an environmental control and life support system for a hypothetical lunar base. The focus of the study are the various flows of substances, chemical reactions and ultimately the analysis of redundancy for key functions such as the extraction of carbon dioxide or the production of oxygen.

10.1. The Extendable Modular Power System

The Extendable Modular Power System was developed at the Chair of Astronautics in collaboration with the University of New South Wales' Space Department. The author of this thesis was the principal investigator for the payload development at the Chair of Astronautics. More information on the system can be found in the thesis supervised by the author on the topic written by Florian Mauracher, Rupert Amann and Fabian Schöttel (cf. [173]–[175]) The system was delivered to the University of New South Wales in early 2019 and was orbited aboard the M2 spacecraft of the University of New South Wales in Q1 2021. At the time of writing, the checkout of the payload in orbit is still ongoing.

10.1.1. Targeted Analyses

In the following, a set of relevant analysis questions are derived, which go beyond the questions formulated in Section 6.2.1.

1. Where lie the next sensors that can be read out to verify a power measurement?
2. Which components' power consumption measures a specific sensor?
3. Which temperature sensors should get warm, when a high amount of power flows through a specific component?

Apart from these questions, queries formulated in Section 9.2 are revisited to investigate the reusability of queries formulated for one model on other models.

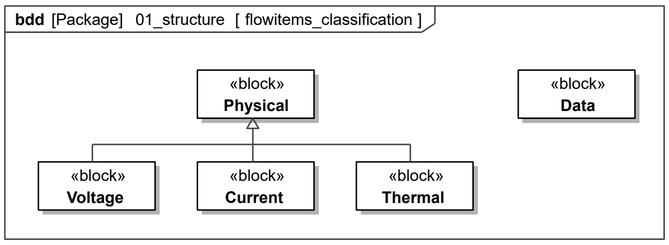


Figure 10-1.: Flowitems classification in the Extendable Modular Power System's model

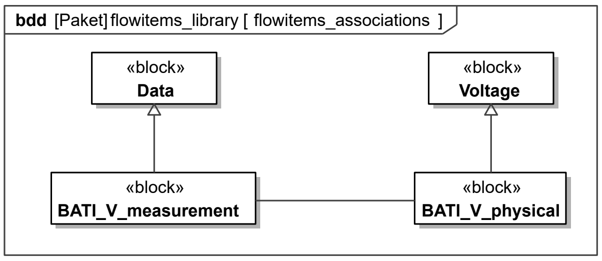


Figure 10-2.: Relation between physical flows and their measurements

10.1.2. SysML Model of the Extendable Modular Power System

The Extendable Modular Power System is integrated in the M2 Spacecraft as an experimental payload. The interest and sphere of influence of the Chair of Astronautics as payload provider end at the interface with the spacecraft. Hence, the model only covers the Extendable Modular Power System itself, not the surrounding spacecraft or the ground infrastructure, as no information about either is available. For comparability to the MOVE-II study presented in Chapter 9, the model also has its focus on the structural aspects. This allows investigating the reusability of the queries developed in Chapter 9 on other systems.

The model contains an implementation-based hierarchical containment structure of the EMPS. It also contains a hierarchical containment and classification structure of the flowitems in the system, which consist of physical flows as well as data flows.

Figure 10-1 shows the classification of flowitems in the model. The actual flowitems are modelled underneath the classes shown in Figure 10-1. An important feature of the flowitems structure is the relation between measurements and physical flows. Relating physical flowitems and their measurements allows querying how a specific physical property can be observed. Figure 10-2 shows the respective resolution on the example of the Battery Input Voltage ("BATI_V"), which is set up using a general purpose association.

The model itself is structured in a library part and a structure part. The library contains the before mentioned classification of flowitems, as well as a classification of components that allows to define ports only once and reuse them throughout the model. Figure 10-3 displays this classification on the example of the switches which regulate the power output on the various channels of the Extendable Modular Power System. Note how the handling of ports described in Section 6.4.1 ensures that each port is automatically instantiated for each role of the SysML Model. The structure-part of the model is organized by packages which each hold a module and its components. Figure 10-4 summarizes the model setup in a package diagram.

Each module holds its own components and internal block diagrams. The internal block diagrams focus on the physical flows and data flows in the system, showing all voltages and currents and their measurements.

Figure 10-5 shows the battery system's internal block diagram as an example.

For the sake of brevity, no behavior is modeled for the Extendable Modular Power System.

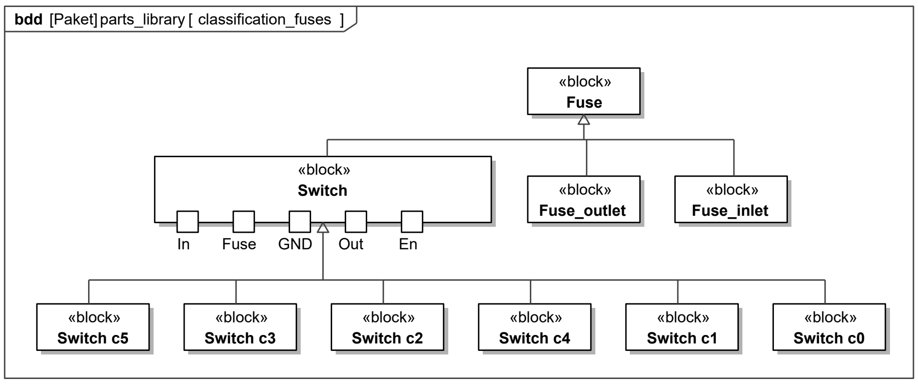


Figure 10-3.: Reducing modeling effort by classification

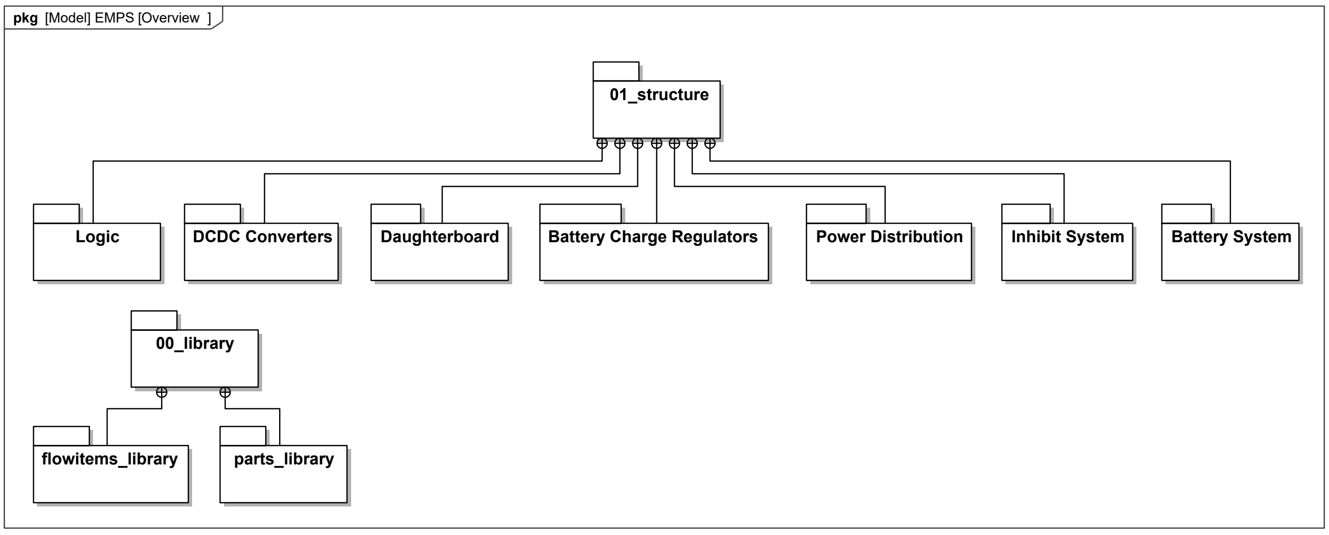


Figure 10-4.: Package structure of the EMPS model

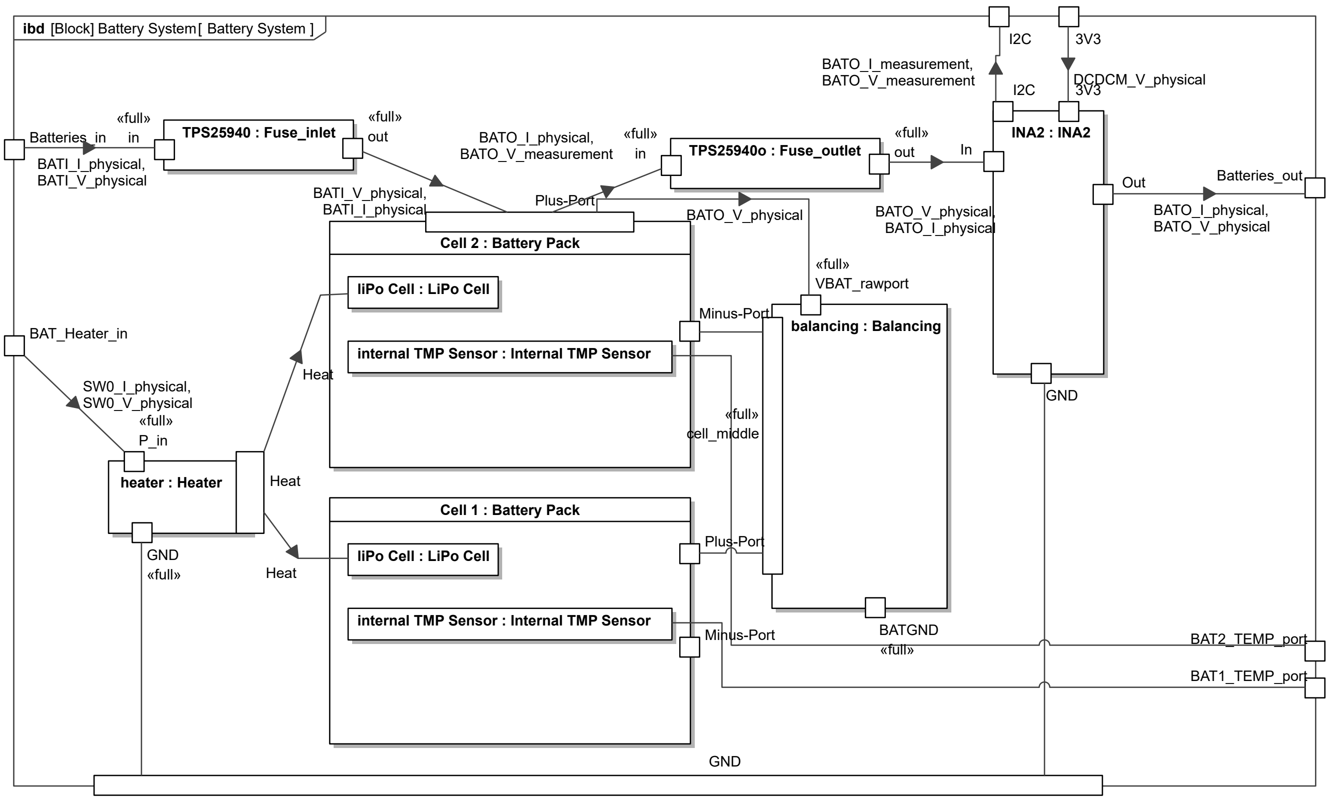


Figure 10-5.: Internal block diagram of the battery system

10.1.3. Graph Analyses on the Extendable Modular Power System

In the following, graph analyses on the Extendable Modular Power System are presented. They are applied on a graph of the system built from the model described above in accordance to the graph schema provided in Chapter 6. The hypothesis presented in Section 3.3 states:

The integration of graph databases with MBSE allows[...] effectively reusing acquired capabilities in the analysis of such systems

While Chapter 9 addresses the reusability of analysis capabilities within the same model, the following section takes the interpretation of the hypothesis a step further and investigates the reusability of queries for the MOVE-II model on the EMPS Model.

Reapplication of Pre-Existing Queries

The following analyses are structured in two parts: The first part is a revision of queries developed on the MOVE-II model, investigating whether queries can be transferred between models. The second part shows the development of new queries specific to the setup of physical and data flows modeled here.

```
:param class=>'Fuse'
MATCH  path = (element)-[:IS_OF_TYPE]->(parent)-[:IS_OF_TYPE*0..]->(c:BLOCK{name:$class})
RETURN element.name, parent.name  ORDER BY length(path), parent.name, element.name
```

element.name	parent.name
Fuse_inlet	Fuse
Fuse_outlet	Fuse
Switch	Fuse
Switch c0	Switch
Switch c1	Switch
Switch c2	Switch
Switch c3	Switch
Switch c4	Switch
Switch c5	Switch

Query 10-1: Retrieving all elements of a class (see Question 3 in Section 6.2)

The following queries from Section 9.2 shall be revisited:

- Question 3, referring to the elements of a class
- Question 4, querying what systems employ a certain type of component
- Question 5, querying the power supply of a system
- Question 12, querying the fallout of an electrical short.

The parameters of these queries are adapted to the new system. For Question 3 the class to be investigated is the "Fuse" Class (cf. Figure 10-3). The query employed for this question is a simplified version of Query 9-2. Query 10-1 shows the appropriate code and response of the system.

The response displayed in the lower half of Query 10-1 procures exactly the same elements as displayed in Figure 10-3, validating the query.

Not all queries can be transferred without adaptions. Depending on the modeling style, slight alterations are necessary. An example for this can be taken from Query 9-2. The MOVE-II model was built at the same time as the first prototype of the code for the extract-transfer-load tool. This earlier prototype was not capable of handling multiple instances of the same port on components of the same class. Therefore, modeling concepts that rely on inheriting ports from a higher class were avoided. The EMPS model was built in a later stage, where handling of port-instances was already implemented. Query 10-2 shows the adapted query, which also works for the MOVE-II Model. Figure 10-6 provides an overview of all ports in the battery system, with ports for which a type was defined highlighted in green.

Regarding the queries listed in Section 9.2.1, which allow tracing data paths and analyzing data anomalies, no adaptions are necessary, apart from specifying new parameters, as the flowitem names in the models differ.

Query 9-3 was reapplied with the variable `searchterm` set to `DCDCM_V_physical`. Table 10-1 shows the result. Query 9-4 returns a list of all flowitems processed by the same components as suggestions for further telemetry that could be affected by an anomaly. However, since the EMPS model includes a large amount of physical flowitems in addition to the data flowitems, a further restriction is necessary to limit the results to only data flowitems. Query 10-3 shows the adapted version, in which the clause `(datatypename:'Data')<-[:IS_OF_TYPE*]-` was added in line 7.

```
MATCH (SystemOfInterest:BLOCK{name:'Battery System'}) <-[:IS_PART_OF*0..]- (b:BLOCK) -[:IS_OF_TYPE*0..]-> (blocktype)
↪  <-[:IS_PART_OF]- (PortInSys:PORT) -[:IS_OF_TYPE]-> (porttype)
RETURN DISTINCT porttype.name, count(porttype) ORDER BY count(porttype) DESC
```

porttype.name	count(porttype)
pwr_in	2
GND	2
I2C	1

Query 10-2: Adapted version of Query 9-2, retrieving all port types within a certain range of equipment

Table 10-1.: Result of Query 9-3, retrieving the data path of a telemetry item on the EMPS model with the searchterm `DCDCM_V_physical`

processedElement	source	target
DCDC measurements	DCDC Converters	logic
DCDC measurements	logic	MSP430
DCDCM_V_measurement	INA5	DCDC Converters

```
CALL{
        MATCH (telemetry{name: $searchterm})-[:FLOWS_IN]->(hpn) RETURN hpn, telemetry
        UNION
        MATCH (telemetry{name:
↪  $searchterm})-[:FLOWS_IN]->()-[:FLOWS]-()-[:IS_PART_OF]->(component)<-[:IS_PART_OF]-()-[:FLOWS]-(hpn) RETURN hpn,
↪  telemetry
        }
WITH hpn, telemetry
MATCH (datatype{name:'Data'})<-[:IS_OF_TYPE*]-(suggestion)-[:FLOWS_IN]->(hpn)
        WHERE NOT suggestion = telemetry
RETURN suggestion.name ORDER BY SHORTESTPATH((telemetry)-[:FLOWS_IN|FLOWS*]-(suggestion))
```

DCDCM_I_measurement

Query 10-3: Adapted version of Query 9-4 filtering for flowitems of type Data

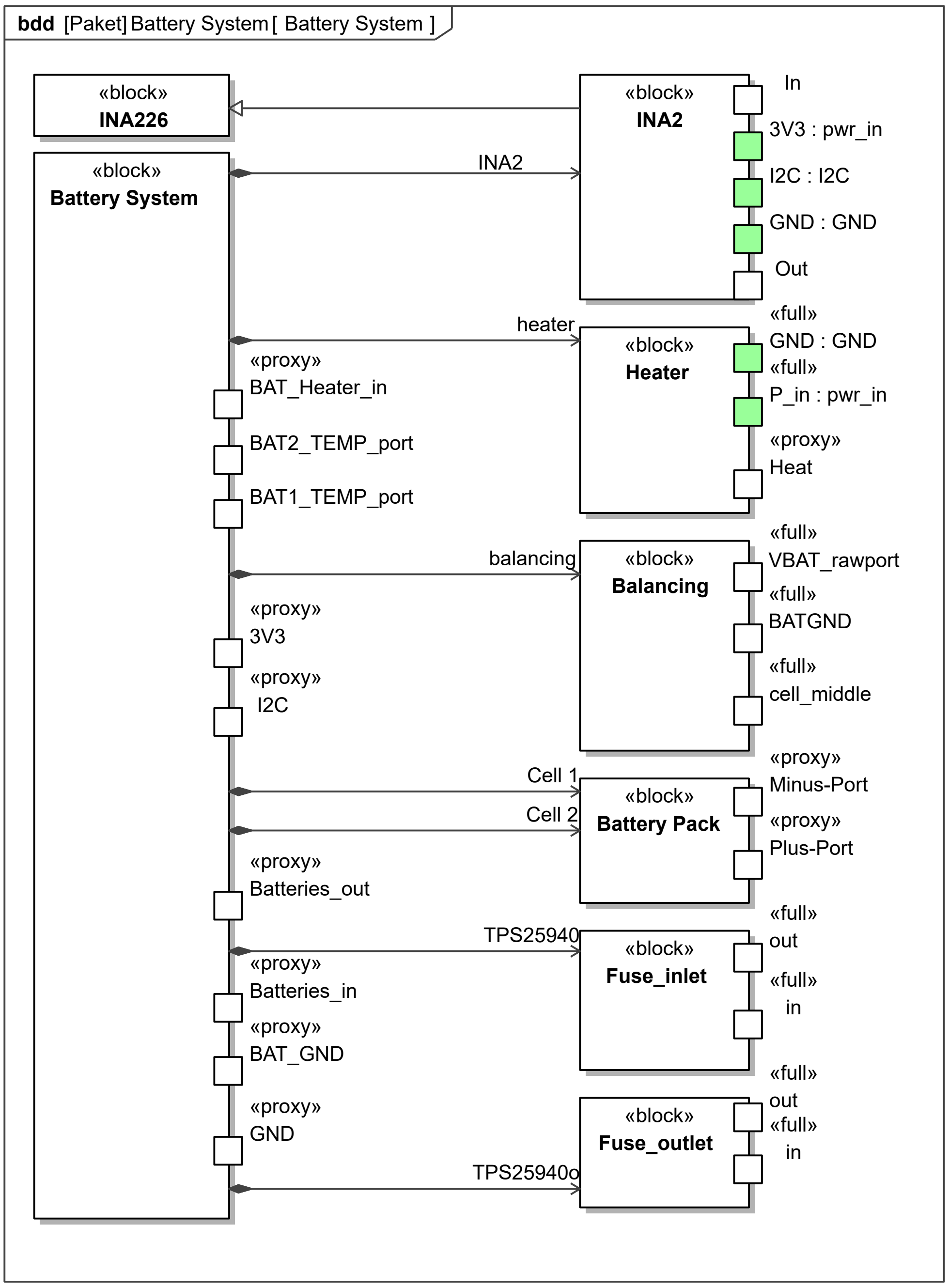


Figure 10-6.: Definition of ports in the Battery System, corresponding to Query 10-2. Ports with defined types are highlighted in green.

Table 10-2.: Results for Query 9-5 on the EMPS Model, which queries a list of fault candidates for a given anomaly pattern in the spacecraft's telemetry.

suggestion	id
DCDC Converters	_19_0_4_64d021d_1645202551438_899875_55291
I2C_port_instance	_19_0_4_64d021d_1645205224151_398668_59291 _19_0_4_6...

Query 9-5 can be applied to the EMPS without any adaptions as well. Table 10-2 shows the list of likely candidates for a fault with the following parameters:

- `fltm1 = 'DCDC5_I_measurement'`
- `fltm2 = 'DCDC5_V_measurement'`
- `fltm3 = 'DCDC3_I_measurement'`
- `goodttlm = 'Switches Measurements'`.

EMPS Specific Queries

In Section 10.1.1 a set of specific questions for the EMPS model were developed. The first of these questions refers to which sensors can be used to verify a measurement. To answer the question, the place where a measurement was taken has to be found, the next physical flows upstream and downstream from the sensor need to be determined, the measurements of these physical flows have to be found and finally the sensors providing the measurements. Query 10-4 shows a possible phrasing for such a query, with results for the flowitem "DCDC3_V_measurement". Figure 10-7 provides an excerpt of the EMPS model that can be used to verify the query. The query yields two false positives (`DCDCM_I_measurement` and `DCDCM_V_measurement`) which result from the power supply of the sensors. The power supply for all on-board components is provided by the power converter DCDCM. It therefore lies upstream on a power path of the sensor INA6, which is the reason for its occurrence in the results of the query.

Query 10-4 relies on the same transient `:FLOWS`-relations as already presented in Section 9.2.2 (see Query 9-6).

The second question in Section 10.1.1 asks for the specific components that draw power from a switch or power converter. This requires to find all down-stream components that take the output of the switch or power converter as input. However, simply following only `:FLOWS`-relations is insufficient, as this would lead the query along data paths as well as power paths. To circumvent the ambiguity, only components that consume outputs of the supply component which are classified as physical are considered. Query 10-5 shows the appropriate code.

To answer the last question formulated in Section 10.1.1, information about the proximity of the thermal sensors to other components would be necessary. As this information is not provided in the model, no answer to the question can be provided.

```
:param searchterm =>'DCDC3_V_measurement'
CALL {
//find the source of the dataflowitem
        MATCH
↪  (dataFlowItem{name:$searchterm})-[:FLOWS_IN]->()<-[:FLOWS]-(portInstance:PORT)-[:IS_PART_OF]->(source:BLOCK)
        WHERE NOT EXISTS {(dataFlowItem{name:$searchterm})-[:FLOWS_IN]->()-[:FLOWS]->(:PORT)-[:IS_PART_OF]->(source)}
        WITH source
//find any physical flowitem at a max. distance of 10 flows downstream (4 flows per one connection block-to-block
↪  means these are max. 3 flows away)
        MATCH path =
↪  (physFlowType{name:'Physical'})<-[:IS_OF_TYPE*]-(physFlow)-[:FLOWS_IN]->(hpn)<-[:FLOWS*..10]-(:PORT)-[:IS_PART_OF]->(source
        WITH physFlow
//find the measurement of the phys flow that is max. 3 block-to-block flows away from the source-sensor
        MATCH(physFlow)-[:IS_ASSOCIATED_WITH]-(alternateMeasurement)-[:IS_OF_TYPE]->(data:BLOCK{name:'Data'})
return distinct alternateMeasurement
//repeat the query for the upstream-direction
UNION
//find the source of the dataflowitem
MATCH (dataFlowItem{name:$searchterm})-[:FLOWS_IN]->()<-[:FLOWS]-(portInstance:PORT)-[:IS_PART_OF]->(source:BLOCK)
        WHERE NOT EXISTS {(dataFlowItem{name:$searchterm})-[:FLOWS_IN]->()-[:FLOWS]->(:PORT)-[:IS_PART_OF]->(source)}
        WITH source
//find any physical flowitem at a max. distance of 10 flows upstream (4 flows per one connection block-to-block means
↪  these are max. 3 flows away)
        MATCH path =
↪  (physFlowType{name:'Physical'})<-[:IS_OF_TYPE*]-(physFlow)-[:FLOWS_IN]->(hpn)-[:FLOWS*..10]->(:PORT)-[:IS_PART_OF]->(source
        WITH physFlow
//find the measurement of the phys flow that is max. 3 block-to-block flows away from the source-sensor
        MATCH(physFlow)-[:IS_ASSOCIATED_WITH]-(alternateMeasurement)-[:IS_OF_TYPE]->(data:BLOCK{name:'Data'})
return distinct alternateMeasurement}
WITH alternateMeasurement
//find the sensors of the alternateMeasurement
MATCH (alternateMeasurement)-[:FLOWS_IN]->()<-[:FLOWS]-()-[:IS_PART_OF]->(alternateSensor) WHERE NOT
↪  EXISTS{(alternateSensor)<-[:IS_PART_OF]-()<-[:FLOWS]-()<-[:FLOWS_IN]-(alternateMeasurement)}
return alternateMeasurement.name, alternateSensor.name
```

alternateMeasurement.name	alternateSensor.name
DCDC3_I_measurement	INA6
DCDC3_V_measurement	INA6
SW4_I_measurement	INA_c4
SW4_V_measurement	INA_c4
SW5_I_measurement	INA_c5
SW5_V_measurement	INA_c5
BATO_I_measurement	INA2
BATO_V_measurement	INA2
DCDCM_I_measurement	INA5
DCDCM_V_measurement	INA5

Query 10-4: Query to find the next sensors for measurement comparisons

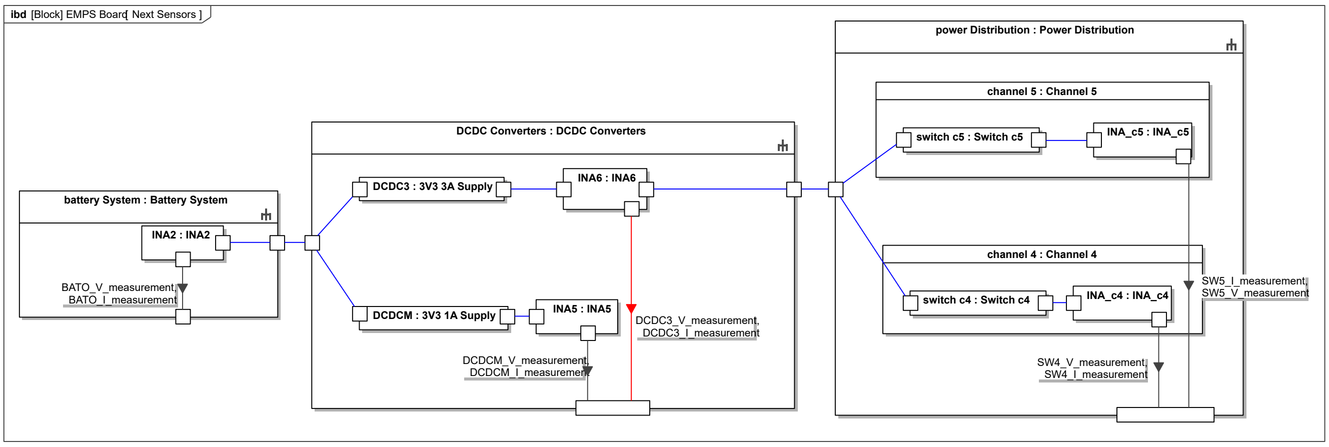

ibd [Block] EMPS Board[Next Sensors]
battery System : Battery System
INA2 : INA2
BATO_V_measurement, BATO_I_measurement
DCDC Converters : DCDC Converters
DCDC3 : 3V3 3A Supply
INA6 : INA6
DCDCM : 3V3 1A Supply
INA5 : INA5
DCDC3_V_measurement, DCDC3_I_measurement
DCDCM_V_measurement, DCDCM_I_measurement
power Distribution : Power Distribution
channel 5 : Channel 5
switch c5 : Switch c5
INA_c5 : INA_c5
channel 4 : Channel 4
switch c4 : Switch c4
INA_c4 : INA_c4
SW5_I_measurement, SW5_V_measurement
SW4_V_measurement, SW4_I_measurement

Figure 10-7.: Excerpt from the EMPS Model for validation of Query 10-4. Port names and itemflows on power lines were stripped off this diagram for readability. Power lines are marked in blue, the red line marks the measurement to be validated.

```
:param supplyComponent=>'DCDC3'
MATCH(supply:BLOCK{name:$supplyComponent}) <-[:IS_PART_OF]- ()-[FLOWS]-> (:HYPERNODE)<-[:FLOWS_IN]-
↪  (suppliedPower)-[:FLOWS_IN]-> (:HYPERNODE)-[:FLOWS]-> ()-[:IS_PART_OF]-> (suppliedComponent)
WHERE EXISTS {(suppliedPower)-[:IS_OF_TYPE*]->(:BLOCK{name:'Physical'})}
RETURN distinct suppliedComponent.name
```

suppliedComponent.name
power Distribution
DCDC Converters
switch c5
switch c4

Query 10-5: Components drawing power from a specific power converter or switch

10.2. Environmental Control Life Support System

The Environmental Control Life Support System modeled for this show-case is taken from Kaschubek, Grill and Killian [171] and the PhD Thesis of Mr. Kaschubek [172]. The systems presented in both publications are combined to an environmental control and life support system for a hypothetical lunar base. The model combines scenarios presented in the two publications to demonstrate how graph transformations of SysML can support analyses of redundancy.

10.2.1. Targeted Analyses

The following analyses shall be performed:

1. Display of possible oxygen supply paths for the Crew
2. Display of possible ways to extract and reduce carbon dioxide
3. Fall out Scenarios, i.e. what happens if a component in the oxygen supply or carbon dioxide removal supply fails

Compared to the previous models (MOVE-II and EMPS), the ECLSS shows major differences. Focus of the model are neither electrical nor data flows but the flows of chemical substances. Furthermore, it is the first system with circular flow systems, as the various chemical substances are recycled and reused in the model. Therefore, the analysis of the ECLSS allow the assessment of a broader application of graph schema and modeling guidelines developed in this thesis.

10.2.2. SysML Model of the Environmental Control Life Support System

Before details of the analyses are presented, the SysML Model of the Environmental Control and Life Support System is presented.

The model includes use cases, requirements and a physical architecture.

Skimming over the use cases and requirements, which are set up similar to their counterparts in the MOVE-II Model and therefore only of lower interest, the physical architecture is made up of a variety of modules and periphery. The following modules are included in the Model in accordance with [171], [172]:

- In-Sitru Resource Utilization (ISRU) Plant
- Biological Environmental Control and Life Support System
- Conventional Environmental Control and Life Support System

All other equipment as well as the crew are directly associated to the Lunar Base Module, also in accordance with [171], [172]. Life support systems require an operational uptime of 100%, as anything less could lead to the death of the crew. The consumables for the crew as well as their waste-products need to constantly be taken care of, which either means a steady stream of supplies from Earth or a recycling system. Current state-of-the-art recycling systems reach recycling rates over 90%. This implies that resupply missions still stay necessary, if less frequently [171].

The needs of the crew can be stated as follows [171]:

- Oxygen supply for breathing
- Carbon dioxide removal from the air
- Water supply

- Food supply
- Taking care of human waste

Each of the modules named above either covers all of these aspects or parts of them.

Figure 10-9 provides an overview of the system, showing the diverse chemical flows between the crew, periphery, and modules. Compared to the previous two models, the main difference in this model lies in the nature of the flowitems and their cyclic flows. In order to assess the possibilities to provide oxygen or water or to extract and reduce carbon dioxide, the various flowitems need to be set in relation to each other. Thereby the paths of flows that can be chemically transformed into the required components can be tracked. Figure 10-8 shows the composition of flowitems in the model. Note how water is made up of oxygen and hydrogen and how ilmenite (an extract from lunar regolith) contains oxygen.

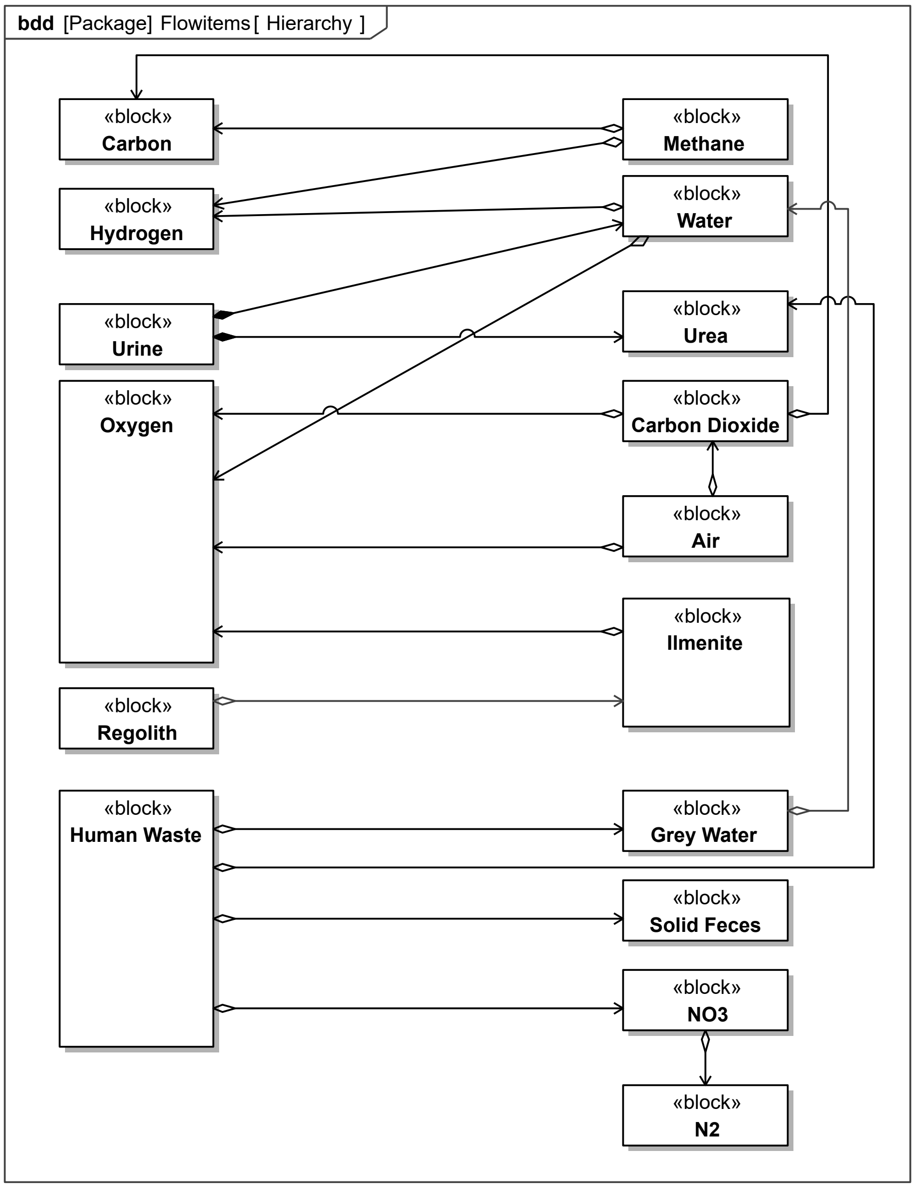


Figure 10-8.: Chemical composition of flowitems in the Environmental Control and Life Support System Model

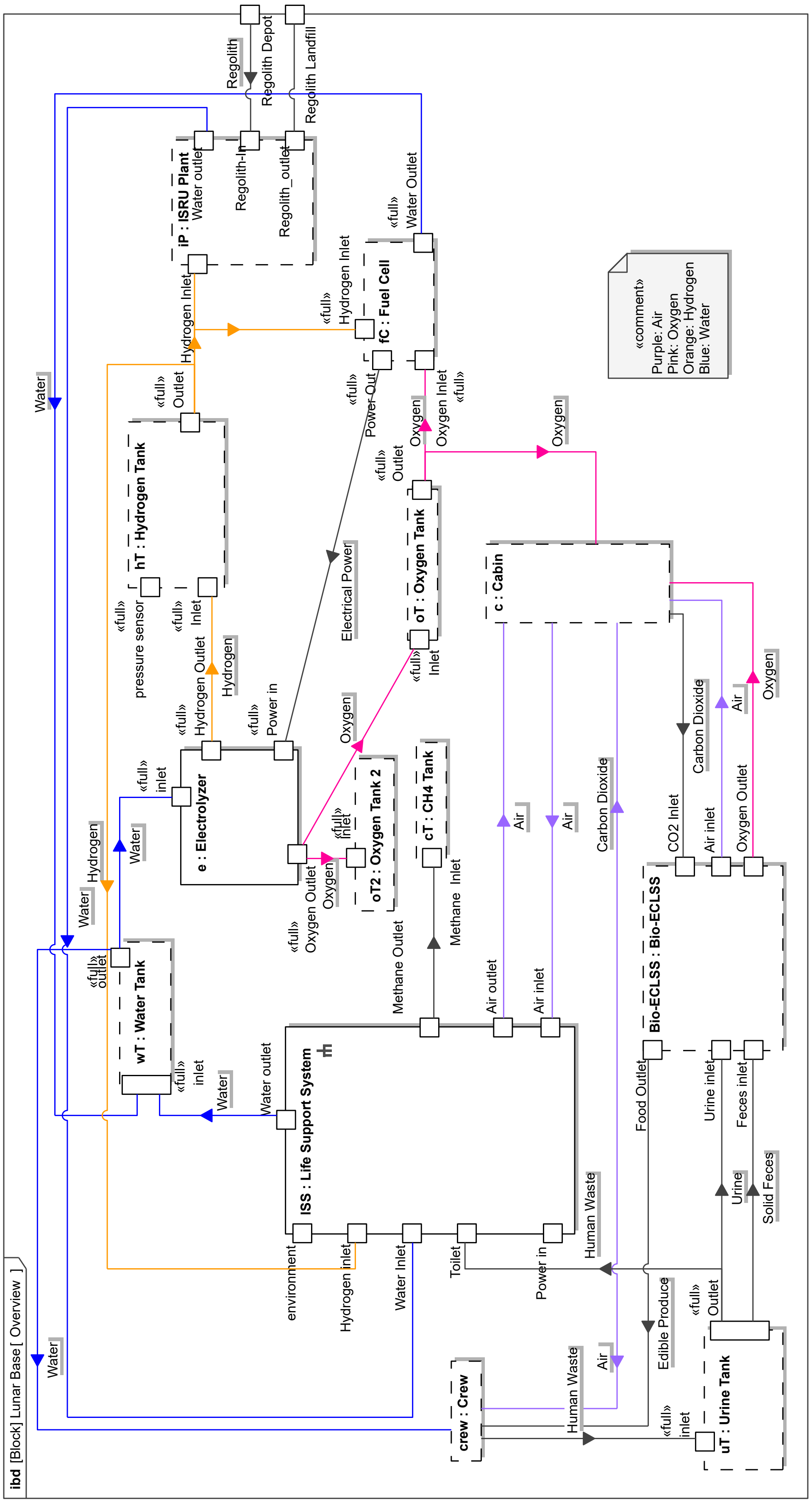


Figure 10-9.: Overview of the Lunar Base

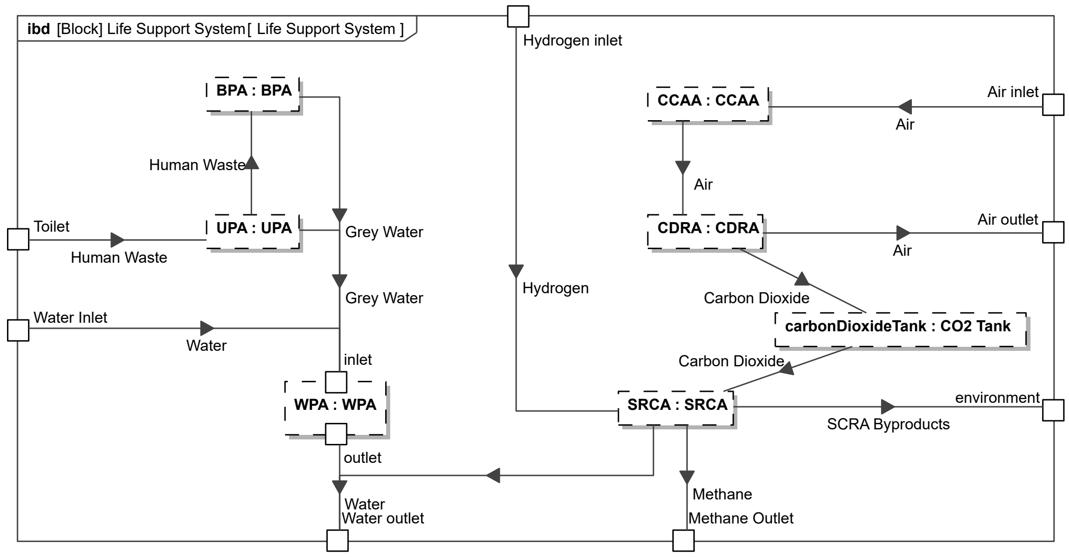


Figure 10-10.: Internal workings of the conventional Life Support System. Descriptions of the abbreviations can be found in the Appendix in Figure A.4

Each module (ISRU Plant, Biological Environmental Control and Life Support System and the Conventional Environmental Control and Life Support System) reuses the flows entering the system and containing chemical reactions that transform the molecular composition of the flowitems. Figure 10-10 and Figure 10-11 show the internal connections of two of the modules. A complete set of internal block diagrams for the Environmental Control and Life Support System can be found in Appendix A.

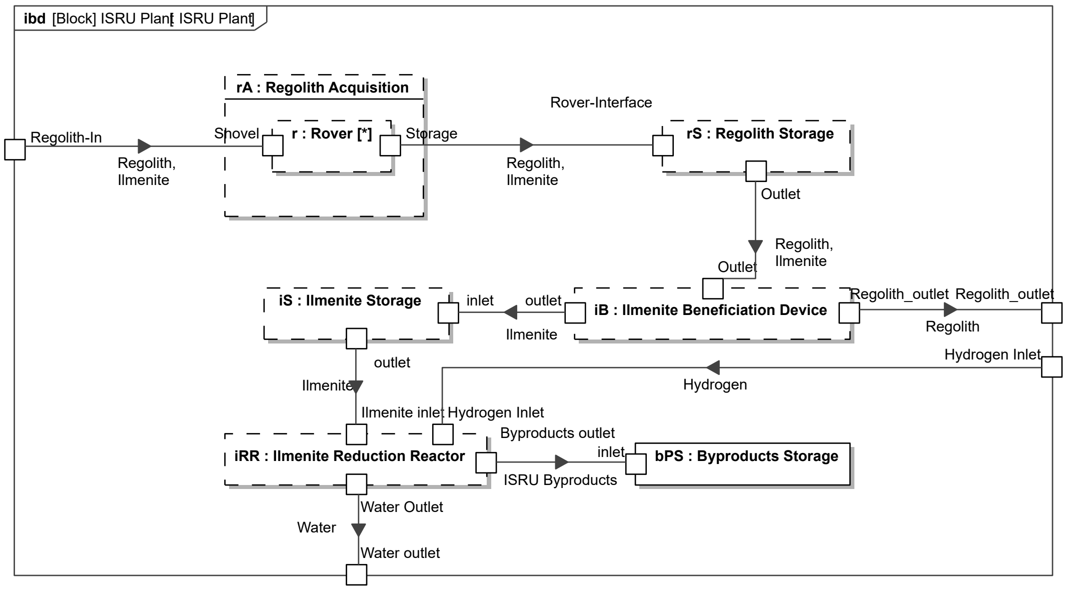


Figure 10-11.: Internal workings of the ISRU Plant

10.2.3. Graph Analyses on the Environmental Control Life Support System

The first targeted analysis in Section 10.2.1 concerning possible paths for the crew's oxygen supply can be conducted by adapting Query 9-3 which queries data paths. While the original query does not specify any beginnings or endings to the data paths, the end of the oxygen-supply path is defined as the crew of the Lunar Base. Taking the original Query 9-3 as-is would result in a list of all components processing either oxygen or any substance that contains oxygen, such as water, air or carbon dioxide, including those that never reach the crew (cf. Table A1). Additionally, the graph of the original data path query in Section 9.2.1 as demonstrated in Figure 9-12 is unwieldy and difficult to interpret.

In order to provide an easier understandable graph-response, the original query for data paths was adapted and split in two parts. In the first part, presented in Query 10-6, a transient relation is created that replaces the construct `(:BLOCK) <-[:IS_PART_OF]- (:PORT) -[:FLOWS]-> (:HYPERNODE) -[:FLOWS]-> (:PORT) -[:IS_PART_OF]-> (:BLOCK)` with a simpler `(:BLOCK) -[:OXYGENFLOWS]-> (:BLOCK)` for all connections transmitting oxygen. The second Query (Query 10-7) builds on these flows and returns a simplified graph that only shows the blocks and, the `:OXYGENFLOWS` connections which contribute to the crew's supply. To allow insights into which actual flowitem flows in an `:OXYGENFLOWS`-connection (e.g. oxygen, air, water, etc.), the name of the flowitem associated with the hypernode of the construct above is set as a property on the `OXYGENFLOWS` connector (see line 4 and 9 of Query 10-6). Figure 10-12 shows the resulting graph diagram.

The next step in the analysis of the Lunar Base model is to identify all possibilities by which carbon dioxide can be removed from the crew's air supply. In order to find the carbon dioxide removal possibilities, a first step is identifying consumers which reduce carbon dioxide. These consumers are characterized by taking in carbon dioxide while providing neither carbon dioxide nor a flowitem containing it. Once these are found, the carbon dioxide's path from the crew to the consumers has to be identified. This can be done by the shortest path algorithm. The algorithm requires a single relation type to follow. Therefore, the construct of `(:BLOCK) <-[:IS_PART_OF]- (:PORT) -[:FLOWS]-> (:HYPERNODE) -[:FLOWS]-> (:PORT) -[:IS_PART_OF]-> (:BLOCK)` needs to be reduced to a single relation type. This is accomplished by the same construct used in Query 10-7, which builds a transient relation type before ap-

```
//find all oxygen-containing flowitems
MATCH (oxygen{name:'Oxygen'})-[:IS_PART_OF*0..]->(flowItems)-[:FLOWS_IN]->(hpn)
//create flowtype property on the hypernodes
set hpn.flowtype=flowItems.name
WITH hpn, flowItems
//find the flow-paths of oxygen or flowitems containing oxygen
MATCH path =
↪  (a:BLOCK:INSTANCE)<-[rel0:FLOWS]-(:PORT)<-[rel1:FLOWS]-(hpn)<-[rel2:FLOWS]-(:PORT)<-[rel3:FLOWS]-(b:BLOCK:INSTANCE)
//create the simplified flow "OXYGENFLOWS" that speeds up digestion of the information and set the flowtype parameter
↪  to the chemical composition of the flow
MERGE (a)<-[of:OXYGENFLOWS{flowtype:hpn.flowtype}]-(b)
```

Query 10-6: Preparing simplified flows in the graph for analysis purposes

```
//second query, showing the oxygen supply of the crew with the simplified :OXYGENFLOWS relations
//retrieve sources of oxygen (blocks that yield but do not consume oxygen),
MATCH(oxsource)-[:OXYGENFLOWS{flowtype:'Oxygen'}]->() WHERE NOT EXISTS
↪  {()-[:OXYGENFLOWS{flowtype:'Oxygen'}]->(oxsource)}
with oxsource
MATCH oxpath = shortestpath((oxsource)-[oxygenFlows:OXYGENFLOWS*]->(crew:BLOCK{name:'crew'}))
UNWIND oxygenFlows AS oxFlow
//limit the results to a single listing of each relation, even if it occurs in multiple paths
WITH DISTINCT  oxFlow,crew
//show the relations
MATCH path2 = ()-[oxFlow]-()
RETURN  path2,crew
```

Query 10-7: Queries to derive the possible routes to supply the crew with oxygen in the Lunar Base

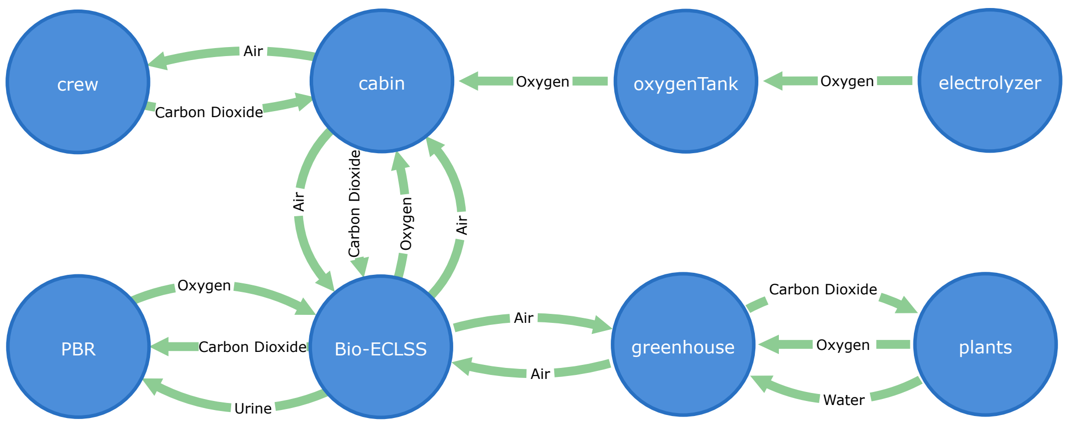


Figure 10-12.: Possible oxygen supply paths for the crew of the lunar base. Graph diagram resulting from Query 10-7. The denominators on relations show the respective flowitem's name

```
        //find consumers of carbon dioxide
        MATCH(consumer)<-[:OXYGENFLOWS{flowtype:'Carbon Dioxide'}]-() WHERE NOT EXISTS
↪   {()<-[:OXYGENFLOWS{flowtype:'Carbon Dioxide'}]-(consumer)}
        WITH consumer
        //find the shortestpaths from crew to consumers
        MATCH co2path = shortestpath((crew:BLOCK{name:'crew'})-[co2flows:OXYGENFLOWS*]->(consumer))
        UNWIND co2flows AS co2flow
        //reduce to a single representation of each relation
        WITH DISTINCT co2flow, crew
        //reduce to relevant flows
        WHERE co2flow.flowtype in ['Carbon Dioxide', 'Air']

    MATCH path = ()-[co2flow]-()
    RETURN path, crew
```

Query 10-8: Paths to remove carbon dioxide from the crew and reduce it

```
MATCH(c:BLOCK:INSTANCE)-[rel:OXYGENFLOWS]-()
WITH DISTINCT c ORDER BY c.name
RETURN collect(c.name)
```

```
["BPA", "Bio-ECLSS", "CCAA", "CCAA2", "CDRA", "PBR", "SRCA", "UPA", "WPA", "cabin", "carbonDioxideTank", "crew", "crop",
"electrolyzer", "foodSupply", "fuelCell", "greenhouse", "ilmeniteBeneficationDevice", "ilmeniteReductionReactor", "ilmeniteStorage", "is-
ruPlant", "lifeSupportSystem", "lunarBase", "nutrientSolution", "oxygenTank", "oxygenTank2", "plants", "regolithStorage", "rover",
"urineTank", "waterTank"]
```

Query 10-9: Listing all components processing either oxygen or carbon dioxide independent of their molecular form.

plying the algorithm. Query 10-8 implements this train of thought. The result is displayed in Figure 10-13. It shows how the crew distributes carbon dioxide to the cabin, which transports it as part of the air to the life support system and the Biological Environmental Control and Life Support System (BIO-ECLSS) system. After multiple processing steps, the three consumers are the Photo Bio Reactor (PBR), the plants, and the Sabatier CO2 Reprocessing Assembly (SRCA), which is consistent with the model.

Compared to the fallout scenarios for data and power queries as described in Section 9.2.2, the fallout scenarios for the lunar base need to consider redundancies in the system. Therefore instead of following all relations from a potentially broken component, the proper path is to take the component out of the analysis and check whether the necessary flows to keep the crew alive can still be provided. Ideally, such an analysis also allows conducting analyses on combinations of errors. To achieve this analysis, the same logic as presented in Query 10-6 can be applied.

Query 10-6 created the `:OXYGENFLOWS` and relations in the system that go directly from component to component along the paths of oxygen and any substance containing oxygen. Query 10-9 provides a list of all oxygen and carbon dioxide processing components.

Out of the components listed in Query 10-9, the following combinations are chosen for the analysis:

1. Common Cabin Air Assembly (CCAA)

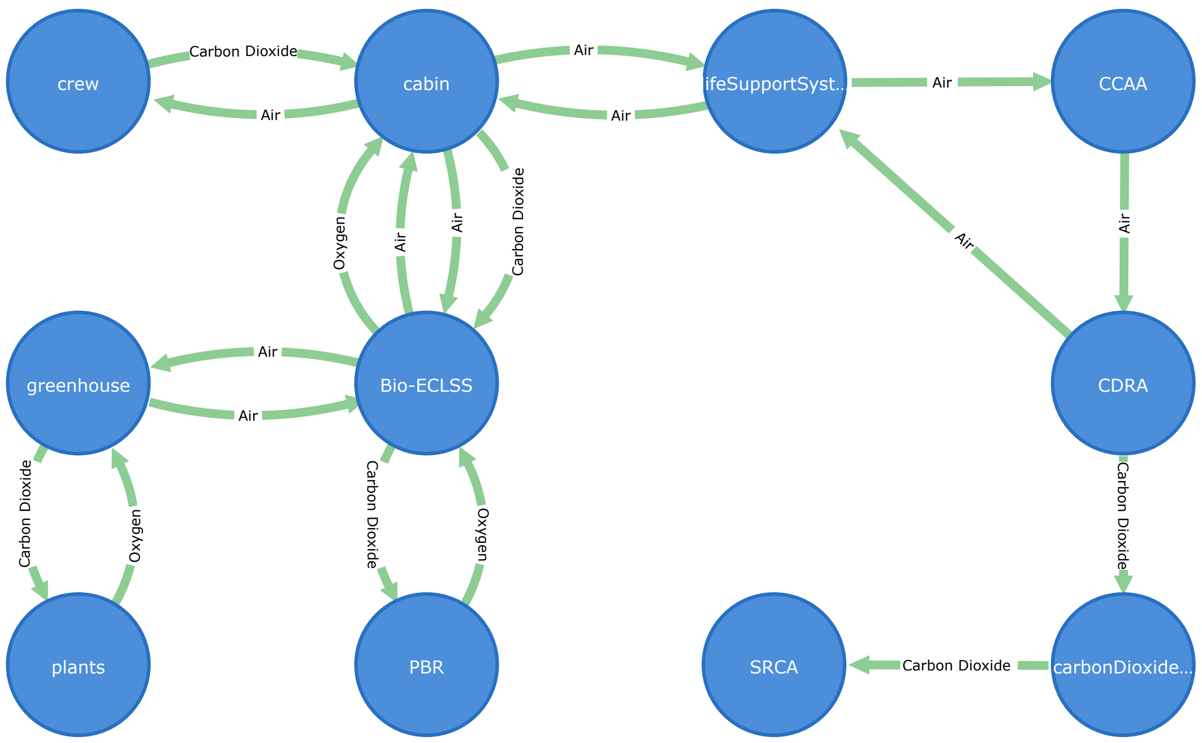


Figure 10-13.: Simplified flow graph of the possible paths to extract and reduce carbon dioxide in the Lunar Base generated by Query 10-8. Source of the carbon dioxide is the crew-node, while the nodes displayed at the bottom of the figure are the consumers of carbon dioxide.

2. greenhouse
3. CCAA and greenhouse
4. PBR, CCAA and greenhouse

Query 10-10 prepares the network of `:OXYGENFLOWS` relations for the fallout analysis and defines the consumer and source nodes which reduce carbon dioxide and produce oxygen. After the consumer and source nodes are marked by respective properties, the `:OXYGENFLOWS` relations of the components considered broken are deleted. Analyzing Figure 10-13 reveals that by disabling the CCAA, the SRCA is cut off, hence the carbon dioxide can only be processed by the PBR and the plants. Disabling the greenhouse results in the plants being cut off, thereby reducing the carbon-dioxide removal possibilities to the PBR and SRCA. By disabling the CCAA and the greenhouse, the only remaining carbon dioxide reductor is the PBR. The final analysis should show that by cutting off the PBR, CCAA and greenhouse, no carbon dioxide can be removed and reduced anymore. Removing the PBR and greenhouse also affects the oxygen supply of the crew. However, as the electrolyzer is still active, the crew still has an oxygen-supply path. Figure 10-14 shows the graphs resulting from the analysis query after cutting off the CCAA and greenhouse (upper part of the figure) and cutting off the PBR as well (lower part of the figure). Query 10-11 shows the respective code, which builds on the shortest path algorithm.

```
// set consumer and source roles for the analysis
MATCH(consumer)<-[:OXYGENFLOWS{flowtype:'Carbon Dioxide'}]-() WHERE NOT EXISTS {()<-[:OXYGENFLOWS{flowtype:'Carbon
↪  Dioxide'}]-(consumer)}
SET consumer.consumer = TRUE
WITH consumer
MATCH(oxsource)-[:OXYGENFLOWS{flowtype:'Oxygen'}]->() WHERE NOT EXISTS
↪  {()-[:OXYGENFLOWS{flowtype:'Oxygen'}]->(oxsource)}
SET oxsource.oxsource=TRUE

//creating a list of the "broken" components
:param brokenComponents=>['CCAA','greenhouse']

//cutting of the broken components from the :OXYGENFLWOS and :CO2FLOWS networks
MATCH (brokenComponent)-[rel:OXYGENFLOWS]-()
WHERE brokenComponent.name in $brokenComponents
delete rel
```

Query 10-10: Graph preparation for fallout analysis of the Lunar Base

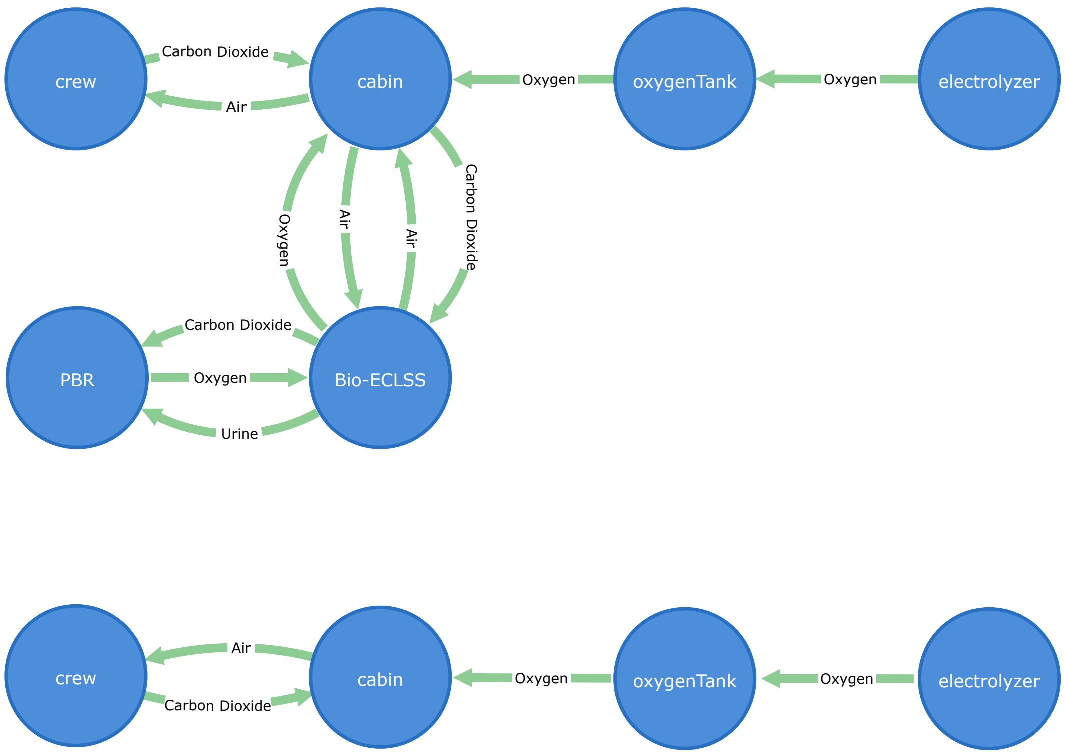


Figure 10-14.: Graph analysis of fallout scenarios. Upper part: fallout of the nodes CCAA and greenhouse, lower part: additional fallout of the node PBR. Compare Figure 10-12,10-13 for the original flow graphs translated from the SysML Model.

```
// use the shortest path algorithm to find any co2 flows between crew and consumers
CALL{
MATCH co2path = shortestpath((crew:BLOCK{name:'crew'})-[co2flows:OXYGENFLOWS*]->(consumer {consumer: TRUE}))
UNWIND co2flows AS co2flow
RETURN co2flow as combinedFlow,crew
UNION
// use the shortest path algorithm to find oxygen flows between source and crew
MATCH oxpath = shortestpath((oxsource{oxsource: TRUE})-[oxygenFlows:OXYGENFLOWS*]->(crew:BLOCK{name:'crew'}))
UNWIND oxygenFlows AS oxFlow
RETURN oxFlow as combinedFlow, crew
}

WITH combinedFlow
WHERE combinedFlow.flowtype in ['Air','Carbon Dioxide', 'Oxygen']
WITH DISTINCT combinedFlow
//show the relations
MATCH path = ()-[combinedFlow]-()
RETURN path
```

Query 10-11: Analyzing the fallout on the carbon dioxide removal of combinations of disabled components

11. Discussion

11.1. Summary

In this thesis a graph schema and modeling guideline was developed to enable in-depth analyses of SysML Models independent of their size and complexity. This was based on theoretical analysis and survey of the informational needs within a small spacecraft development project. The phases D (System Assembly Integration & Test, Launch & Checkout) and E (Operations and Sustainment) of the project were identified as critical regarding their informational needs due to the clash of intended design and implemented design in Phase D and the lack of knowledge carriers from previous phases in Phase E, combined with the high amount of heterogeneous information relevant to Phase E (cf. Section 2.3).

An approach was laid out to support the transfer of knowledge in those later phases, based on SysML Models of the space mission (cf. Chapter 3).

SysML Models were created for three Systems to enable a validation of the graph schema and modeling guideline:

- the MOVE-II space mission (including the spacecraft, its ground station and all further ground infrastructure), with a focus on data-flows
- The Extendable Modular Power System
- The Lunar Base System

The capacity of the graph schema and modeling guidelines was demonstrated on the MOVE-II System by developing queries that address the analysis goals defined in Section 6.2 and Section 7.1. While the universal applicability of the modeling guidelines and graph schema could not be proven, it was validated by application on the fundamentally different Extendable Modular Power System and Lunar Base System. Respective graphs could be generated by the software written for this thesis for each of the three systems. In the following, the capabilities demonstrated in Chapter 9 and Chapter 10 are evaluated.

11.1.1. Evaluation of Analyses on the MOVE-II Model

The structural analyses presented in Section 9.2 show that all analysis goals for the structural aspects of a SysML Model could be validated against the MOVE-II SysML Model's graph transformation. The queries make excessive use of the patterns in the graph generated by the modeling guideline.

Section 9.2.3 shows how the analysis goals regarding requirements use cases and stakeholders can be met. Except Question 35, queries were derived for all analysis goals of Section 9.2.3.

Translating information about the system into the graph schema and connecting it to the SysML Model shows good prospect, as demonstrated in Section 9.3.3, where it could be shown how among hundreds of documents, all test reports on a specific requirement or component can be retrieved by querying the graph database.

11.1.2. Evaluation of the Graph Schema's Extensibility

The two models used for further analyses in Chapter 10 also adhere to the modeling guidelines presented in Section 6.5. The first goal of the extensibility analysis was to show how queries can be reused across systems provided they follow the same modeling style.

Section 10.1 shows that slight adaptions may be necessary from model to model compensating for slight changes in the modeling style. While this means some rework on the queries is to be expected, it also shows that the necessary rework requires only minor efforts. While a direct and unambiguous application of every query across all modeling styles is desirable, the fact that the queries can be adapted with only minor effort allows compensating for these differences without having to restructure the whole SysML Model, which would be considerably more effort.

The analyses specific for the Extendable Modular Power System further shows how the graph translation of the SysML Model allows users with little prior knowledge troubleshooting a system of considerable complexity. It also shows however, that the graph can only yield information that can be derived in one way or another from the information fed to it. Therefore, the last analysis goal of Section 10.1.1 could not be accomplished.

Analyzing the Lunar Base required transferring some modeling guidelines to a system in a completely different technical field. By interpreting the rules for modeling of informational flows (cf. Section 6.5.2) to the chemical composition of the lunar base's flowitems the concepts applied on the MOVE-II system and the Extendable Modular Power System could be reused on the lunar base.

Creating further transient relations and properties which are not part of the original further allowed for simpler graphs that can be interpreted faster and enable even simpler and more concise queries. This can be seen by comparing Figure 9-12 and Figure 10-13 as well as Query 10-4 and Query 10-11. More important than the simplicity of the queries however is the potential of automatically generating such analyses. The execution time for each of these queries lies well under a second, with most finishing in a span of less than 20ms on an average office computer. Thereby the analyses presented here can be automated and applied for all components and all combinations of components within a system with minimal required manpower and independent of the system's complexity or size.

11.1.3. Further Graph Schema Evaluation

The evaluation of the graph schema and modeling guidelines provides confidence for the modeling and analysis of structural SysML aspects as well as any aspects related to requirements, use cases and stakeholders. However, no evaluation of the graph schema and modeling guidelines for state machines and activity diagrams could be conducted. The Master's Thesis "Investigation of the Applicability of an Artefact[sic] Model in SysML using the CubeSat MOVE-II" by Nikolas Faatz [176], supervised by the author of this thesis, developed and compared multiple approaches of modeling the behavior of the MOVE-II spacecraft. Faatz found a wide variety of possibilities to model a spacecraft's behavior and the right modeling style to be dependent on the target audience.

The graph schema of state machines and activity diagrams was implemented in code and verified on the examples in Section 6.4.2. However, this does not compensate for a full validation on a complete model, seeing as the thesis also lacks any queries on the content of state machines or activity diagrams.

While this is a shortcoming of the conducted research it does not invalidate the provided results and analyses.

11.2. Evaluation of Hypothesis

In Section 3.3, the following hypothesis was formulated:

The integration of graph databases with MBSE allows tasks requiring interdisciplinary knowledge and system understanding being performed efficiently and effectively during the assembly integration and testing as well as operations of a small spacecraft and effectively reusing acquired capabilities in the analysis of such systems.

Chapter 9 and Section 10.1 showed how queries on a graph database can yield information that otherwise would require cumbersome and repetitive work. The same information could also be taken from the SysML Model directly. This would either require the modeler to anticipate which combination of information exactly is required and create diagrams for each case. Alternatively, the user seeking information could evaluate all existing diagrams in the model to assemble the information manually. Both approaches are inefficient and prone to errors.

The Systems Modeling Language is able to describe hardware as well as software, both in their structure and their behavior. Thereby the information drawn from the SysML Model of the spacecraft is in itself already an interdisciplinary body of information. In Section 9.3 the capability to deal with interdisciplinary information was further emphasized by drawing in any available documentation on the MOVE-II project and integrating it in the graph database and with the MOVE-II SysML Model. Although only 38% of all terms used in the SysML Model could be mapped to any of the information inserted in the graph database, the technical feasibility and potential benefits of such an approach could be shown.

Therefore, the statement *The Integration of graph databases with MBSE allows tasks requiring interdisciplinary knowledge and system understanding being performed efficiently and effectively during the assembly integration and testing as well as operations of a small spacecraft* can be seen as validated.

Section 9.2 and Section 10.1.3 further shows, how analysis capabilities in the form of Cypher-queries can be reused. This applies for analyses within the same model as well as the transfer of queries to other systems as long as their modeling style is compatible with the modeling guideline.

Finally, the development of a SysML Model based on the research of Kaschubek, Grill and Killian [171], [172] for a Lunar Base shows that the restriction of the hypothesis to small spacecraft and the assembly integration and testing phase may not be necessary. The modeling guideline proved to be applicable to this system focussed on chemical reactors and flows of substances as well as small spacecraft.

11.3. Comparison to State of the Art

In the beginning of Chapter 2, terms such as data, information and knowledge were defined. Combining the SysML Model with the informational environment of the development showed how additional context of data can be identified and thereby creating new value. As discussed in Section 2.1.3, classical knowledge (i.e. omitting the area of artificial intelligence) in difference to information exists only in the human mind. As such, the additional context to the information allows an easier understanding and thereby faster intake, speeding up learning processes.

Compared to the state of the art in information management systems, the agnostic approach was taken in this thesis. Instead of trying to invent a new way to organize information in the first place, any digitized information management can be connected to the graph. One of the problems outlined in Section 2.2.4 is the existence of multiple, redundant communication paths, i.e. the same information could have been conveyed as a comment in a ticketing-system, an email, a chat message or a formal report etc. Connecting these information management systems with the graph provides the means to search all of these systems at the same time, as shown on the example of PDF documents, software documentation and commit messages for MOVE-II in Section 9.3.3.

Another issue, common to any information management system could be resolved as well: Judging whether a piece of information is up-to-date as could be shown by Query 9-20 in Section 9.3.3. 38% of the blocks and 36% of the `:INFO`-nodes could be matched by a simple comparison of names. The application of Natural Language Processing algorithms should be able to ramp up this number and provide more outputs

Comparing the analyses that could be accomplished to those published in the literature (cf. [134], [135]) it becomes clear, that the combination of modeling guideline and graph schema enables far more capabilities than the generic approaches presented in the literature. The complexity of the analyses presented in the literature only barely surpasses the capabilities of SysML modeling tools such as MagicDraw or Enterprise Architect. Examples of these analyses are:

- Finding which percentage of requirements lack satisfy or verify relations [135],
- Showing all elements connected to a certain element.

By comparison, the analyses presented in this thesis go far beyond such simplistic information, as can be seen by the data path and anomaly queries presented in Section 9.2 or the redundancy and fallout analysis conducted on the Lunar Base in Section 10.2.3.

12. Outlook

This work shows how with very few modeling rules and a sound graph schema intricate analyses can be conducted on a SysML model's structural aspects and requirements related aspects. While focussed on small spacecraft as an area of expertise the author is familiar with, it shows that the same concepts can be applied to fundamentally different systems such as the Lunar Base. The most obvious topic for future work is to validate the schema and modeling guidelines for activity diagrams and state machines. In addition to this, the schema should be expanded for those aspects of the SysML that are currently neglected. These encompass among others:

- Constraints
- Parameter Diagrams
- Custom Stereotypes
- and Sequence Diagrams

Another area deliberately not treated in this thesis is the question of connection behavioral and structural aspects. Two concepts should be developed to cover object-oriented modeling as well as models that separate behavior and structure, since both are common modeling schemas.

Another step in the future should be the application of the analysis tool developed in this thesis on a project in development. While two of the applications used as showcases in this thesis are real-life projects with flying space hardware, a project in development has a different characteristic from one already finished. One aspect of this should be the development of change management capabilities, allowing reviews and merge requests in the modeling process.

While Section 9.3 showed that it is possible to link digitized information management systems with the SysML model in the graph, an enhancement of the matching functions could greatly benefit the system.

The model elements representing the telemetry of a spacecraft could be linked to the telemetry database. Thereby machine learning algorithms can be applied on the telemetry, making use of the context provided by the SysML Model. This is already being done in the MBSE2DL project [177], which is conducted in cooperation with the author of this thesis and further explored in the Master's Thesis of Prokaiev [178].

Finally, a proper graphical user interface should be set up. The goal of this text is to scientifically explore the capabilities a symbiosis of Model Based Systems Engineering and graph databases yields. Since the neo4j browser interface provided sufficient capabilities for this thesis, no extra effort was put into developing a specific graphical user interface. However, the time spent analyzing the models in the graph database, could significantly be reduced by an interface that is able to translate the query results back into the familiar diagram types of SysML.

Bibliography

[1] I. Merriam-Webster. "Infodemic: An Epidemic of Information | Merriam Webster." (2021), [Online]. Available: https://www.merriam-webster.com/words-at-play/words-were-watching-infodemic-meaning (visited on 02/22/2021).

[2] N. Grinberg, K. Joseph, L. Friedland, B. Swire-Thompson, and D. Lazer, "Fake news on twitter during the 2016 u.s. presidential election," *Science*, vol. 363, no. 6425, pp. 374–378, 2019. DOI: 10.1126/science.aau2706. eprint: https://www.science.org/doi/pdf/10.1126/science.aau2706. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.aau2706.

[3] J. Allen, B. Howland, M. Mobius, D. Rothschild, and D. J. Watts, "Evaluating the fake news problem at the scale of the information ecosystem," *Science Advances*, vol. 6, no. 14, 2020. [Online]. Available: https://www.science.org/doi/abs/10.1126/sciadv.aay3539.

[4] A. Guess, J. Nagler, and J. Tucker, "Less than you think: Prevalence and predictors of fake news dissemination on facebook," *Science Advances*, vol. 5, no. 1, 2019. [Online]. Available: https://www.science.org/doi/abs/10.1126/sciadv.aau4586.

[5] M. Brunnermeier, "Nist planning report 99h1," *NIST Planning Report 99H1*, 2004.

[6] L. Nancy, "Role of software in spacecraft accidents," *JOURNAL OF SPACECRAFT AND ROCKETS*, 2004.

[7] ——, "Software challenges in achieving space safety," *Journal of the British Interplanetary Society*, 2009.

[8] A. Berquand, A. Riccardi, F. Murdaca, *et al.*, "Artificial intelligence for the early design phases of space missions," 2019.

[9] *ESA's Technology Strategy v1.1*. European Space Agency, 2019.

[10] M. P. Robillard, A. Marcus, C. Treude, *et al.*, "On-demand developer documentation," in *2017 IEEE International Conference on Software Maintenance and Evolution*, 2017, pp. 479–483.

[11] A. J. Ko, R. DeLine, and G. Venolia, "Information needs in collocated software development teams," in *Proceedings of the 29th International Conference on Software Engineering*, ser. ICSE '07, IEEE Computer Society, 2007, pp. 344–353. [Online]. Available: https://doi.org/10.1109/ICSE.2007.45.

[12] A. Ko, B. Myers, M. Coblenz, and H. Aung, "An exploratory study of how developers seek, relate, and collect relevant information during software maintenance tasks," *Software Engineering, IEEE Transactions on*, vol. 32, pp. 971–987, Jan. 2007. DOI: 10.1109/TSE.2006.116.

[13] U. Lindemann, *Methodische Entwicklung technischer Produkte: Methoden flexibel und situationsgerecht anwenden*, ser. VDI-Buch. Springer Berlin Heidelberg, 2009, ISBN: 9783642014222. [Online]. Available: https://books.google.de/books?id=WL1evwEACAAJ.

[14] S. S. Library and N. Administration, *NASA Systems Engineering Handbook - NASA SP-2016-6105 Rev2: Design Test Integrate Fly*. CreateSpace Independent Publishing Platform, 2017, ISBN: 9781977821966. [Online]. Available: https://books.google.de/books?id=VqabtAEACAAJ.

[15] J. Martin, *Systems Engineering Guidebook: A Process for Developing Systems and Products*. CRC Press, 2020, ISBN: 9780429606892. [Online]. Available: https://books.google.de/books?id=TdHgDwAAQBAJ.

[16] U. Lindemann, M. Maurer, and T. Braun, *Structural Complexity Management: An Approach for the Field of Product Design*. Springer Berlin Heidelberg, 2010, ISBN: 9783642099670. [Online]. Available: https://books.google.de/books?id=pDQDYgEACAAJ.

[17] S. Friedenthal, A. Moore, and R. Steiner, *A Practical Guide to SysML: The Systems Modeling Language*, ser. The MK/OMG Press. Elsevier Science, 2011, ISBN: 9780123852069. [Online]. Available: https://books.google.de/books?id=8KY2YZIiXv0C.

[18] Thomas McDermott, Nicole Hutchison et al., "Benchmarking the benefits and current maturity of model-based systems engineering across the enterprise," Stevens Institute of Technology, Tech. Rep., 2020.

[19] J. B. M. Chami, "A survey on mbse adoption challenges," *INCOSE EMEA Sector Systems Engineering Conference*, 2018.

[20] B. M. et al., "Issues in conceptual design and mbse successes: Insights from the model-based conceptual design surveys," 2016.

[21] R. J. Pawlikowski Gerald Holladay Jon, "Systems engineering and model-based systems engineering stakeholder state of the discipline - independent assessment of perception from external/non-nasa systems engineering (se) sources," National Aeronautics and Space Administration, Tech. Rep., 2019.

[22] Schummer Florian, Hyba Maximilian, "A methodology for system analysis with mbse and graph data engineering," *Data-Centric Engineering*, 2021.

[23] International Council on Systems Engineering, "Systems engineering vision 2020," 2007.

[24] European Space Agency, "Mb4se user needs," Tech. Rep., 2020.

[25] LiquidPlanner Inc. "2017 State of Project Management in Manufacturing." (2019), [Online]. Available: https://www.liquidplanner.com/blog/stats-2017-project-management-manufacturing-report/ (visited on 07/24/2019).

[26] I. R. McChesney and S. Gallagher, "Communication and co-ordination practices in software engineering projects," *Inf. Softw. Technol.*, vol. 46, pp. 473–489, 2004.

[27] K. North and G. Kumta, *Knowledge Work(ers) in the Digital Age*. Springer International Publishing AG, 2018, pp. 109–156.

[28] Douglas Harper. "data | origin and meaning of data by Online Etymology Dictionary." (), [Online]. Available: https://www.etymonline.com/word/data (visited on 10/14/2021).

[29] "Data," in *The Concise Encyclopedia of Statistics*. Springer New York, 2008.

[30] V. Ferretti, "Dictionary german — english," in *Woerterbuch der Datentechnik / Dictionary of Computing: Englisch-Deutsch / Deutsch-Englisch*. Springer Berlin Heidelberg, 1996.

[31] B. Witt, *Datenschutz kompakt und verständlich: Eine praxisorientierte Einführung*, ser. Edition kes. Vieweg+Teubner Verlag, 2010, ISBN: 9783834896537. [Online]. Available: https://books.google.de/books?id=%5C_NoYxjz61F8C.

[32] WebFinanceInc. "Business Dictionary Website." (), [Online]. Available: http://www.businessdictionary.com/definition/information.html (visited on 02/23/2020).

[33] O. U. Press. "Oxford Advanced American Dictionary." (), [Online]. Available: https://www.oxfordlearnersdicti
com/definition/american_english/metadata (visited on 07/24/2019).

[34] E. Bolisani and C. Bratianu, "The elusive definition of knowledge," in *Emergent Knowledge Strategies: Strategic Thinking in Knowledge Management*. Cham: Springer International Publishing, 2018.

[35] Plato. Penguin Books, 375 BC.

[36] Aristotle. Oxford World Classics, 335-322 BC.

[37] V. Inozemtsev, M. Ivleva, and V. Ivlev, "Artificial intelligence and the problem of computer representation of knowledge," in *Proceedings of the*, vol. 2, 2017, p. 1151.

[38] A. Mohammadpour, E. Karan, and S. Asadi, "Artificial intelligence techniques to support design and construction," in *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, IAARC Publications, vol. 36, 2019, pp. 1282–1289.

[39] D. Langenberg, "Sucht ihr noch oder wisst ihr schon?," 2013.

[40] M. Corp. "Sharepoint, Team collaboration software tools." (), [Online]. Available: `https://products.office.com/en-us/sharepoint/collaboration` (visited on 07/24/2019).

[41] A. Inc. "Confluence Team COllaboration Software." (), [Online]. Available: `https://www.atlassian.com/software/confluence` (visited on 07/24/2019).

[42] T. Gruber, "Ontology," in *Encyclopedia of Database Systems*, L. Liu and M. T. Oezsu, Eds. New York, NY: Springer New York, 2016, pp. 1–3, ISBN: 978-1-4899-7993-3. DOI: `10.1007/978-1-4899-7993-3_1318-2`. [Online]. Available: `https://doi.org/10.1007/978-1-4899-7993-3_1318-2`.

[43] G. Yue, J. Liu, and Y. Hou, "Design rationale knowledge management: A survey," in *International Conference on Cooperative Design, Visualization and Engineering*, Springer, 2018, pp. 245–253.

[44] D. Noble and H. W. Rittel, "Issue-based information systems for design," 1988.

[45] W. Kunz and H. W. Rittel, *Issues as elements of information systems*. Citeseer, 1970, vol. 131.

[46] J. Lee and K.-Y. Lai, "What's in design rationale?" *Human-computer interaction*, vol. 6, no. 3, pp. 251–280, 1991.

[47] W. C. Regli, X. Hu, M. Atwood, and W. Sun, "A survey of design rationale systems: Approaches, representation, capture and retrieval," *Engineering with computers*, vol. 16, no. 3-4, pp. 209–235, 2000.

[48] F. M. Shipman and R. J. McCall, "Integrating different perspectives on design rationale: Supporting the emergence of design rationale from design communication," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 11, no. 2, pp. 141–154, 1997. DOI: `10.1017/S089006040000192X`.

[49] M. Klein, "Capturing design rationale in concurrent engineering teams," *Computer*, vol. 26, no. 1, pp. 39–47, 1993. DOI: `10.1109/2.179154`.

[50] A. Tang, M. A. Babar, I. Gorton, and J. Han, "A survey of architecture design rationale," *Journal of systems and software*, vol. 79, no. 12, pp. 1792–1804, 2006.

[51] K. R. LLC. "Slack Future of Work Study." (), [Online]. Available: `https://slackhq.com/trust-tools-and-teamwork-what-workers-want` (visited on 07/24/2019).

[52] "The KCS Academy » Knowledge-Centered Service." (), [Online]. Available: `https://www.serviceinnovation` `org/current-members/` (visited on 01/03/2019).

[53] "Knowledge centered services v6 adoption guide," Tech. Rep., 2019.

[54] "The KCS Academy » Knowledge-Centered Service." (), [Online]. Available: `https://www.thekcsacademy.net/kcs/` (visited on 01/03/2019).

[55] R. Alkadhi, J. O. Johanssen, E. Guzman, and B. Bruegge, "React: An approach for capturing rationale in chat messages," Nov. 2017. DOI: 10.1109/ESEM.2017.26.

[56] R. Alkadhi, M. Nonnenmacher, E. Guzman, and B. Bruegge, "How do developers discuss rationale?," Mar. 2018. DOI: 10.1109/SANER.2018.8330223.

[57] Wikimedia Foundation. "Wikipedia Website." (2021), [Online]. Available: https://en.wikipedia.org/ (visited on 07/17/2021).

[58] ——, "Wikipedia Article on Wikipedia." (2021), [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia (visited on 10/16/2021).

[59] Matt Chase. "Wikipedia is 20, and its reputation has never been higher." (2021), [Online]. Available: https://www.economist.com/international/2021/01/09/wikipedia-is-20-and-its-reputation-has-never-been-higher (visited on 10/16/2021).

[60] Dan Fletcher. "A Brief History of Wikipedia." (2009), [Online]. Available: http://content.time.com/time/business/article/0,8599,1917002,00.html (visited on 10/16/2021).

[61] C. Wagner, "Wiki: A technology for conversational knowledge management and group collaboration," *Commun. Assoc. Inf. Syst.*, vol. 13, p. 19, 2004.

[62] A. Majchrzak, C. Wagner, and D. Yates, "Corporate wiki users: Results of a survey," in *Proceedings of the 2006 international symposium on Wikis*, 2006, pp. 99–104.

[63] C. Standing and S. Kiniti, "How can organizations use wikis for innovation?" *Technovation*, vol. 31, no. 7, pp. 287–295, 2011.

[64] C. C. Pfaff and H. M. Hasan, "Overcoming organisational resistance to using wiki technology for knowledge management," 2006.

[65] J. Grudin and E. S. Poole, "Wikis at work: Success factors and challenges for sustainability of enterprise wikis," in *Proceedings of the 6th international symposium on Wikis and open collaboration*, 2010, pp. 1–8.

[66] O. Arazy and I. R. Gellatly, "Corporate wikis: The effects of owners' motivation and behavior on group members' engagement," *Journal of Management Information Systems*, vol. 29, pp. 87–116, 2012.

[67] J.-P. Lang. "Redmine website." (), [Online]. Available: https://www.redmine.org/ (visited on 10/27/2021).

[68] G. Inc. "Gitlab website." (), [Online]. Available: https://about.gitlab.com/ (visited on 10/27/2021).

[69] A. P. Ltd. "Jira website." (), [Online]. Available: https://www.atlassian.com/software/jira (visited on 10/27/2021).

[70] J.-P. Lang. "Redmine features." (), [Online]. Available: https://www.redmine.org/projects/redmine/wiki/Features (visited on 10/27/2021).

[71] G. Inc. "Gitlab features." (), [Online]. Available: https://about.gitlab.com/features/ (visited on 10/27/2021).

[72] A. P. Ltd. "Jira features." (), [Online]. Available: https://www.atlassian.com/de/software/jira/features (visited on 10/27/2021).

[73] Y. B. Leau, W. K. Loo, W. Y. Tham, and S. F. Tan, "Software development life cycle agile vs traditional approaches," in *International Conference on Information and Network Technology*, vol. 37, 2012, pp. 162–167.

[74] M. Stoica, M. Mircea, and B. Ghilic-Micu, "Software development: Agile vs. traditional.," *Informatica Economica*, vol. 17, no. 4, 2013.

[75] Merriam-Webster, Inc. “Intranet | Merriam Webster.” (2021), [Online]. Available: https://www.merriam-webster.com/dictionary/intranet (visited on 10/27/2021).

[76] R. Fagin, R. Kumar, K. Mccurley, *et al.*, “Searching the workplace web,” May 2003. DOI: 10.1145/775152.775204.

[77] Y. Li, Z. Liu, and H. Zhu, “Enterprise search in the big data era: Recent developments and open challenges,” *Proc. VLDB Endow.*, vol. 7, pp. 1717–1718, 2014. [Online]. Available: https://doi.org/10.14778/2733004.2733071.

[78] D. E. Jones, N. Chanchevrier, C. McMahon, B. Hicks, *et al.*, “A strategy for artefact-based information navigation in large engineering organisations,” in *DS 80-10 Proceedings of the 20th International Conference on Engineering Design (ICED 15) Vol 10: Design Information and Knowledge Management Milan, Italy, 27-30.07. 15*, 2015, pp. 143–154.

[79] D. E. Jones, Y. Xie, C. McMahon, M. Dotter, N. Chanchevrier, and B. Hicks, “Improving enterprise wide search in large engineering multinationals: A linguistic comparison of the structures of internet-search and enterprise-search queries,” in *Product Lifecycle Management in the Era of Internet of Things*, A. Bouras, B. Eynard, S. Foufou, and K.-D. Thoben, Eds., Cham: Springer International Publishing, 2016, pp. 216–226.

[80] Apache Software Foundation. “Apache solr website.” (2021), [Online]. Available: https://solr.apache.org/ (visited on 02/25/2022).

[81] Iphos IT Solutions GmbH. “Enterprise search software - unternehmensinterne suchmaschine searchit.” (2021), [Online]. Available: https://www.searchit-enterprise-search.com/ (visited on 01/25/2022).

[82] R. Dumitrescu et al., *Advanced Systems Engineering Wertschoepfung im Wandel*. Fraunhofer IEM Paderborn, 2021.

[83] G. Dubos, S. Schreiner, D. A. Wagner, G. Jones, A. A. Kerzhner, and J. Kaderka, “Architecture modeling on the europa project,” in *AIAA SPACE 2016*, 2016, p. 5310.

[84] C. B. Phillips and R. T. Pappalardo, “Europa clipper mission concept: Exploring jupiter’s ocean moon,” *Eos, Transactions American Geophysical Union*, vol. 95, no. 20, pp. 165–167, 2014.

[85] T. Bayer, J. Day, E. Dodd, *et al.*, “Europa clipper: Mbse proving ground,” in *2021 IEEE Aerospace Conference*, 2021.

[86] R. Karban, M. Zamparelli, B. Bauvir, B. Koehler, L. Noethe, A. Balestra, “Exploring model based engineering for large telescopes: getting started with descriptive models,” in *Modeling, Systems Engineering, and Project Management for Astronomy III*, International Society for Optics and Photonics, vol. 7017, SPIE, 2008.

[87] K. R. Andolfato Luigi, “Workstation software framework,” *Proceedings of SPIE - The International Society for Optical Engineering*, 2008.

[88] Z. M. Karban Robert Andolfato Luigi, “Towards model re-usability for the development of telescope control systems,” 2009.

[89] W. T. Karban Robert Hauber Rudolf, “Mbse in telescope modeling,” *INSIGHT*, vol. 12, 2009.

[90] D. D. Karban Robert Kornweibel Nicholas, “A,” *International Nuclear Information System*, 2011.

[91] C. Gibson, R. Karban, L. Andolfato, and J. Day, “Formal validation of fault management design solutions,” *ACM SIGSOFT Software Engineering Notes*, vol. 39, no. 1, pp. 1–5, 2014.

[92] ——, “Abstractions for executable and checkable fault management models,” *Procedia Computer Science*, vol. 28, pp. 146–154, 2014, 2014 Conference on Systems Engineering Research. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050914000829.

[93] R. Malone, B. Friedland, J. Herrold, and D. Fogarty, "Insights from large scale model based systems engineering at boeing," in *INCOSE International Symposium*, Wiley Online Library, vol. 26, 2016, pp. 542–555.

[94] M. Brandstaetter and K. Roder, "Ein artefaktmodell zur verbesserung der prozessmodellierung," pp. 1–4, 2017.

[95] M. Brandstaetter and C. Buehler, "Using the artifact model to model cae artifacts in the mbse process,"

[96] M. Brandstaetter and K. Roder, "Ein artefaktmodell zur verbesserung der prozessmodellierung systementwicklungsprozess von der idee zum produkt," 2017.

[97] J. J. Miller, "Graph database applications and concepts with neo4j," in *Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA*, vol. 2324, 2013.

[98] N. Mark. "Link Prediction with Neo4j Part 2: Predicting co-authors using scikit-learn." (), [Online]. Available: https://towardsdatascience.com/link-prediction-with-neo4j-part-2-predicting-co-authors-using-scikit-learn-78b42356b44c (visited on 03/31/2020).

[99] S. B. Merkl. "The 5-Minute Interview: David Meza, Chief Knowledge Architect, NASA." (2016), [Online]. Available: https://neo4j.com/blog/david-meza-chief-knowledge-architect-nasa/ (visited on 03/29/2020).

[100] M. Goman, M. Rath, and P. Mader. "Lessons learned from analyzing requirements traceability using a graph database." (2017).

[101] S. Nurdiati and C. Hoede, "25 years development of knowledge graph theory: The results and the challenge," *Memorandum*, vol. 1876, no. 2, pp. 1–10, 2008.

[102] L. Ehrlinger and W. Woess, "Towards a definition of knowledge graphs.," *SEMANTiCS (Posters, Demos, SuCCESS)*, vol. 48, pp. 1–4, 2016.

[103] R. Clancy, I. F. Ilyas, and J. Lin, "Scalable knowledge graph construction from text collections," in *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 39–46. DOI: 10.18653/v1/D19-6607. [Online]. Available: https://www.aclweb.org/anthology/D19-6607.

[104] A. Singhal. "Introducing the knowledge graph: Things, not strings." (May 2012), [Online]. Available: https://blog.google/products/search/introducing-knowledge-graph-things-not/ (visited on 02/25/2022).

[105] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic web*, vol. 8, no. 3, pp. 489–508, 2016.

[106] T. Berners-Lee *et al.* "Semantic web road map." (1998), [Online]. Available: https://www.w3.org/DesignIssues/Semantic.html (visited on 02/25/2022).

[107] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific american*, vol. 284, no. 5, pp. 34–43, 2001.

[108] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data: The story so far," in *Semantic services, interoperability and web applications: emerging concepts*, IGI global, 2009, pp. 205–227.

[109] Dan Brickley, Ramanathan Guha. "RDF 1.1 Schema." (2014), [Online]. Available: https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/ (visited on 06/24/2021).

[110] World Wide Web Consortium. "RDF 1.1 Concepts and Abstract Syntax." (2014), [Online]. Available: https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/ (visited on 06/24/2021).

[111] F. Manola, E. Miller, B. McBride, *et al.*, "Rdf primer," *W3C recommendation*, vol. 10, no. 1-107, p. 6, 2004.

[112] N. Chah, "Ok google, what is your ontology? or: Exploring freebase classification to understand google's knowledge graph," *arXiv preprint arXiv:1805.03885*, 2018.

[113] N. Guarino, D. Oberle, and S. Staab, "What is an ontology?" In *Handbook on Ontologies*, S. Staab and R. Studer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 1–17.

[114] European Space Agency. "OSMoSE - Home." (), [Online]. Available: `https://mb4se.esa.int/OSMOSE_Main.html` (visited on 06/27/2021).

[115] Serge Valera, Quirien Wijnands. "OSMoSE - Governance." ESA MBSE2021 Workshop. ().

[116] Quirien Wijnands, Serge Valera. "The Space System Ontology." ESA MBSE2021 Workshop. ().

[117] National Aeronautics and Space Administration. "openCaesar." (), [Online]. Available: `http://www.opencaesar.io/` (visited on 11/02/2021).

[118] M. D. Watson and P. A. Farrington, "Nasa systems engineering research consortium: Defining the path to elegance in systems," in *Conference on Systems Engineering Research*, 2016.

[119] Sean MCGoey. "Panama papers revenue recovery reaches 1.36 billion as investigations continue." (2021), [Online]. Available: `https://www.icij.org/investigations/panama-papers/panama-papers-revenue-recovery-reaches-1-36-billion-as-investigations-continue/` (visited on 02/25/2022).

[120] M. Cabra, E. Kissane. "Wrangling 2.6tb of data: The people and the technology behind the panama papers." (2016), [Online]. Available: `https://www.icij.org/investigations/panama-papers/data-tech-team-icij/` (visited on 02/25/2022).

[121] M. Hunger, W. Lyon. "Analyzing the panama papers with neo4j: Data models, queries and more." (2016).

[122] P. Krill. "IBM hails ALM standards participation." (), [Online]. Available: `https://web.archive.org/web/20121014215047/http://www.networkworld.com/news/2009/082509-ibm-hails-alm-standards.html` (visited on 11/24/2021).

[123] A. G. Ryman, A. Le Hors, and S. Speicher, "Oslc resource shape: A language for defining constraints on linked data," *LDOW*, vol. 996, 2013.

[124] C. Kemper, *Beginning Neo4j*. Springer, 2015.

[125] R. Pittmann. "Neo4j Powers Intelligent Commerce for eBay App on Google Assistant." (2019), [Online]. Available: `https://neo4j.com/users/ebay/` (visited on 03/29/2020).

[126] L. C. Newman Doug. "Connecting Data with Data Usage: A Graph Approach." (2020), [Online]. Available: `https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20200000256.pdf` (visited on 03/29/2020).

[127] D. B. West, *Introduction to Graph Theory*, 2nd ed. Prentice Hall, Sep. 2000, ISBN: 0130144002.

[128] I. Robinson et al., *Graph Databases: New Opportunities for Connected Data*, 2nd. O'Reilly Media, Inc., 2015, ISBN: 1491930896.

[129] T. Neumann, G. Weikum, "Rdf-stores und rdf-query-engines," *Datenbank-Spektrum*, vol. 11, no. 1, pp. 63–66, 2011.

[130] A. Hodler and M. Needham, *Graph Algorithms: Practical Examples in Apache Spark and Neo4j*. O'Reilly Media Incorporated, 2019.

[131] R. Angles and C. Gutierrez, "An introduction to graph data management," in *Graph Data Management: Fundamental Issues and Recent Developments*, G. Fletcher, J. Hidders, and J. L. Larriba-Pey, Eds. Springer International Publishing, 2018, pp. 1–32.

[132] M. Bajaj, D. Zwemer, R. Peak, A. Phung, A. G. Scott, and M. Wilson, "Slim: Collaborative model-based systems engineering workspace for next-generation complex systems," in *2011 Aerospace Conference*, IEEE, 2011, pp. 1–15.

[133] M. Bajaj, D. Zwemer, R. Yntema, *et al.*, "Mbse++—foundations for extended model-based systems engineering across system lifecycle," in *INCOSE International Symposium*, Wiley Online Library, vol. 26, 2016, pp. 2429–2445.

[134] M. Bajaj, J. Backhaus, T. Walden, *et al.*, "Graph-based digital blueprint for model based engineering of complex systems," in *INCOSE International Symposium*, Wiley Online Library, vol. 27, 2017, pp. 151–169.

[135] L. Petnga, "Graph-based assessment and analysis of system architecture models," *Proceedings of the 29th Annual INCOSE International Symposium*, 2019.

[136] N. R. Quigley, P. E. Tesluk, E. A. Locke, and K. M. Bartol, "A multilevel investigation of the motivational mechanisms underlying knowledge sharing and performance," *Organization science*, vol. 18, pp. 71–88, 2007.

[137] R. Karban, M. Zamparelli, B. Bauvir, and G. Chiozzi, "Three years of mbse for a large scientific programme: Report from the trenches of telescope modeling," *INCOSE*, 2012.

[138] R. Karban, L. Andolfato, P. Bristow, *et al.*, "Model based systems engineering for astronomical projects," vol. 9150, Aug. 2014, p. 91500L. DOI: 10.1117/12.2055540.

[139] ESA Requirements and Standards Division, *Space engineering System engineering general requirements*. ECSS-E-ST-10C, 2017.

[140] National Aeronautics and Space Administration, *NASA Systems Engineering Handbook Rev 2*. National Aeronautics and Space Administration, 2016.

[141] ESA Requirements and Standards Division, *Space Management - Organization and Conduct of Reviews*. ECSS-E-ST-1001C, 2017.

[142] , *NASA Space Flight Program and Project Management Handbook*. National Aeronautics and Space Administration, 2010, NPR 7120.5.

[143] Jesse Leitner, *GUIDELINE FOR FORMING AND OPERATING FAILURE REVIEW BOARDS AND ANOMALY REVIEW BOARDS*. NASA Goddard Spaceflight Center, 2019.

[144] , *Procedural Handbook for NASA Program and Project Management of Problems, Nonconformances, and Anomalies*. National Aeronautics and Space Administration, 2008, NASA-HDBK-8739.18.

[145] *Fault Management Handbook*. National Aeronautics and Space Administration, 2012, Draft.

[146] —, *Procedural Handbook for NASA program and Project Management of Problems Nonconformances and Anomalies*. National Aeronautics and Space Administration, 2008, NASA_HDKB-8739.18.

[147] Roland Duphily, Harold Harder, Rodney Morehead, Joe Harman, Helen Gjerde, Susanne Dubios, Thomas Stout, David Ward, Thomas Reinsel, Jim Loman, *Root Cause Investigation Best Practicse Guide*. National Reconnaissance Office, 2014.

[148] H. Shyldkrot, E. Shmidt, D. Geron, *et al.*, "The first commercial lunar lander mission: Beresheet," in *Proceedings of the AAS/AIAA Astrodynamics Specialist Conference, Portland, ME*, 2019, pp. 11–15.

[149] J. Clements, T. Murphy, L. Jasper, and C. Jacka, "Tailored systems engineering processes for low cost high risk missions," 2019.

[150] E. Kulu. "Nanosats Database | Constellations, companies, technologies and more." (), [Online]. Available: https://www.nanosats.eu/#figures (visited on 11/17/2021).

[151] *Start-up space update on investment in commercial space ventures*, Presentation, 2021.

[152] M. Langer and J. Bouwmeester, "Reliability of cubesats-statistical data, developers' beliefs and the way forward," 2016.

[153] E. Kulu, "In-space economy in 2021-statistical overview and classification of commercial entities," 2021.

[154] G. D. Israel, "Determining sample size," 1992.

[155] D. Bryce and J. Cappaert, "Smallsat manufacturing; the spire constant npi model," 2019.

[156] L. Berthoud, M. Swartwout, J. Cutler, D. Klumpar, J. A. Larsen, and J. D. Nielsen, "University cubesat project management for success," 2019.

[157] Intercax LLC. "Syndeia - Intercax | Software For Integrated MBSE." (2021), [Online]. Available: `https://intercax.com/products/syndeia/` (visited on 07/17/2021).

[158] B. M. Selvy, A. Roberts, M. Reuter, *et al.*, "V&v planning and execution in an integrated model-based engineering environment using magicdraw, syndeia, and jira," in *Modeling, Systems Engineering, and Project Management for Astronomy VIII*, International Society for Optics and Photonics, vol. 10705, 2018, 107050U.

[159] A. Fisher, M. Nolan, S. Friedenthal, *et al.*, "3.1. 1 model lifecycle management for mbse," in *INCOSE International Symposium*, Wiley Online Library, vol. 24, 2014, pp. 207–229.

[160] R. Van Bruggen, *Learning Neo4j*. Packt Publishing Ltd, 2014.

[161] Object Modelling Group, *OMG Systems Modeling Language (OMG SysML) Version 1.6*. Object Modelling Group, 2019.

[162] P. Resnick. "Request for comment 5322." (), [Online]. Available: `https://datatracker.ietf.org/doc/rfc5322/`.

[163] W. Hills, W. Daniel, M. Y. Lu, O. Schaer, and S. Adams, "Modeling client churn for small business-to-business firms," in *2020 Systems and Information Engineering Design Symposium (SIEDS)*, 2020. DOI: `10.1109/SIEDS49339.2020.9106673`.

[164] Software Freedom Conservancy. "Git website." (2022), [Online]. Available: `https://git-scm.com/` (visited on 01/28/2022).

[165] ExplosionAI GmbH. "Spacy - industrial strength natural language processing in python." (2022), [Online]. Available: `https://spacy.io/` (visited on 01/28/2022).

[166] Albert Opoku. "Named entity recognition with spacy python package automated information extraction from text - natural language processing." (2019), [Online]. Available: `https://opokualbert.com/data%20science/nlp/named-entity-spacy/` (visited on 01/28/2022).

[167] -. "Neo4j CSV Import." (), [Online]. Available: `https://neo4j.com/docs/cypher-manual/4.4/clauses/load-csv/` (visited on 03/24/2022).

[168] MOVE-II Development Team, "Move-ii system documentation," Technical University of Munich, Tech. Rep., 2018.

[169] ——, "Move-ii critical design review," Technical University of Munich, Tech. Rep., 2016.

[170] "The neo4j cypher manual v4.4." (2022), [Online]. Available: `https://neo4j.com/docs/cypher-manual/current/` (visited on 03/08/2022).

[171] D. Kaschubek, M. Killian, and L. Grill, "System analysis of a moon base at the south pole: Considering landing sites, ECLSS and ISRU," *Acta Astronautica*, vol. 186, pp. 33–49, Sep. 2021.

[172] D. Kaschubek, "Hybrid life support systems," Ph.D. dissertation, 2022.

[173] F. Mauracher, “Scalable and modular architecture for self-organizing power systems on small spacecrafts,” Ph.D. dissertation, 2019.

[174] R. Amann, “Controllable battery charge regulator for cubesat applications,” Ph.D. dissertation, 2019.

[175] F. Schoettl, “Idp documentation software development for a modular power supply for cubesats,” Tech. Rep., 2019.

[176] Faatz Niklas, “Investigation of the applicability of an artifact model in sysml using the cubesat move-ii,” 2019.

[177] F. Schummer, A. Müller, M. Hyba, B. Beyreuther, and A. Lindner, “Mbse in ait and operations of move-ii and its impact on the mbse2dl activity and the move-iii mission,” Sep. 2021.

[178] Mykyta Prokaiev, “Todo,” 2022.

A. ECLSS Diagrams

The following diagrams complete the information on the ECLSS Model.

Table A1.: Result to querying all elements processing oxygen or an oxygen-holding material on the Lunar Base System (cf. Query 9-3, Section 10.2.3)

processedElement	source	target
Oxygen	BIO-ECLSS	cabin
Oxygen	PBR	BIO-ECLSS
Oxygen	plants	greenhouse
Oxygen	greenhouse	crop
Oxygen	electrolyzer	oxygenTank2
Oxygen	oxygenTank	cabin
Oxygen	oxygenTank	fuelCell
Oxygen	electrolyzer	oxygenTank
Air	BIO-ECLSS	cabin
Air	CCAA	greenhouse
Air	greenhouse	CCAA
Air	BIO-ECLSS	greenhouse
Air	cabin	crew
Air	lifeSupportSystem	cabin
Air	cabin	lifeSupportSystem
Air	Carbon Dioxide Removal Assembly (CDRA)	lifeSupportSystem
Air	lifeSupportSystem	CCAA
Air	CCAA	CDRA
Carbon Dioxide	crew	cabin
Carbon Dioxide	cabin	BIO-ECLSS
Carbon Dioxide	BIO-ECLSS	PBR
Carbon Dioxide	greenhouse	plants
Carbon Dioxide	crop	greenhouse
Carbon Dioxide	CDRA	SRCA
Water	foodSupply	BIO-ECLSS
Water	PBR	nutrientSolution
Water	nutrientSolution	plants
Water	plants	greenhouse

Continued on next page

Table A1 – continued from previous page

processedElement	source	target
Water	CCAA	nutrientSolution
Water	waterTank	electrolyzer
Water	waterTank	crew
Water	SRCA	lifeSupportSystem
Water	lifeSupportSystem	Water Processing Assembly (WPA)
Water	WPA	lifeSupportSystem
Water	lifeSupportSystem	waterTank
Water	isruPlant	lifeSupportSystem
Water	fuelCell	waterTank
Water	ilmeniteReductionReactor	isruPlant
Grey Water	BIO-ECLSS	crop
Grey Water	crop	nutrientSolution
Grey Water	Brine Processing Assembly (BPA)	WPA
Grey Water	Urine Processing Assembly (UPA)	WPA
Human Waste	urineTank	lifeSupportSystem
Human Waste	crew	urineTank
Human Waste	UPA	BPA
Human Waste	lifeSupportSystem	UPA
Urine	urineTank	BIO-ECLSS
Urine	BIO-ECLSS	PBR
Ilmenite	regolithStorage	ilmeniteBeneficationDevice
Ilmenite	rover	regolithStorage
Ilmenite	isruPlant	rover
Ilmenite	ilmeniteStorage	ilmeniteReductionReactor
Ilmenite	ilmeniteBeneficationDevice	ilmeniteStorage
Regolith	lunarBase	isruPlant
Regolith	ilmeniteBeneficationDevice	isruPlant
Regolith	isruPlant	rover
Regolith	rover	regolithStorage
Regolith	regolithStorage	ilmeniteBeneficationDevice

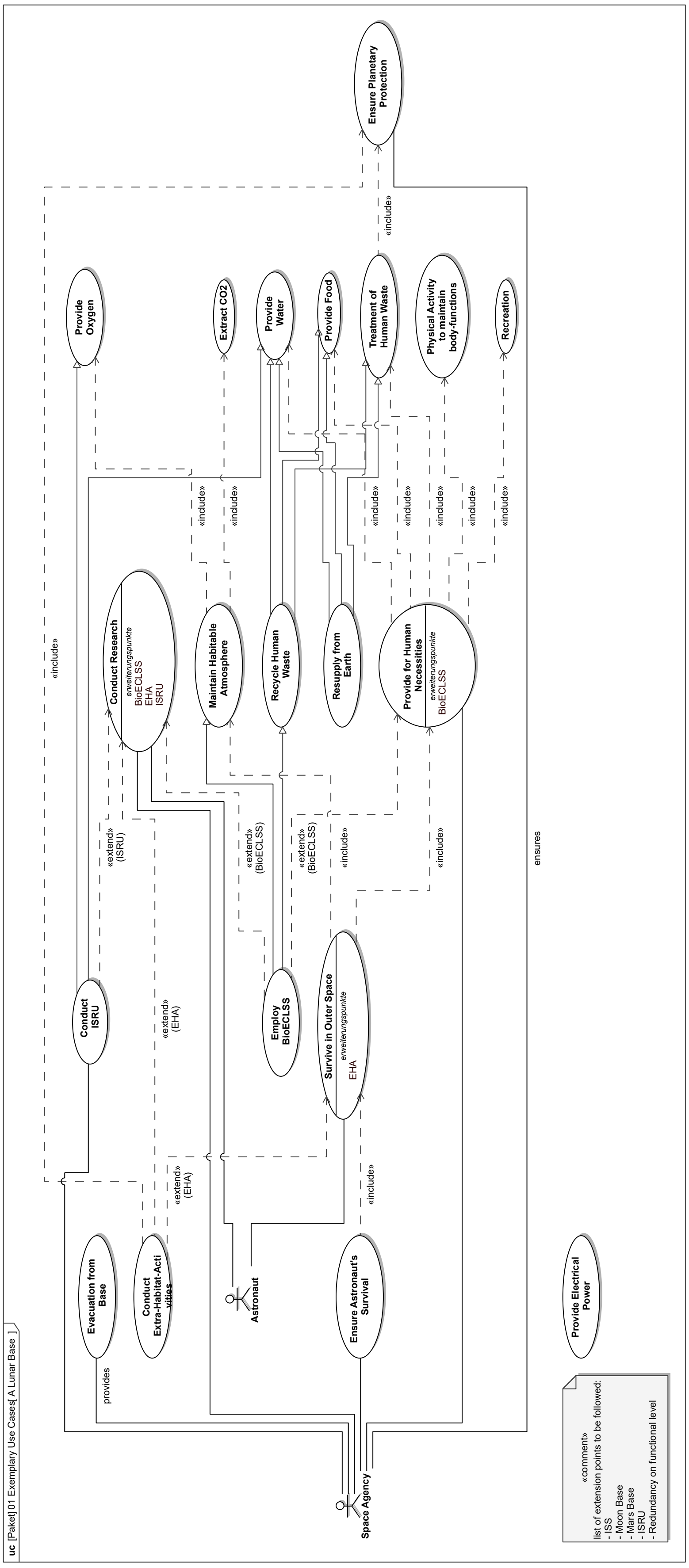


Figure A.1.: Use cases of the Lunar Base Model

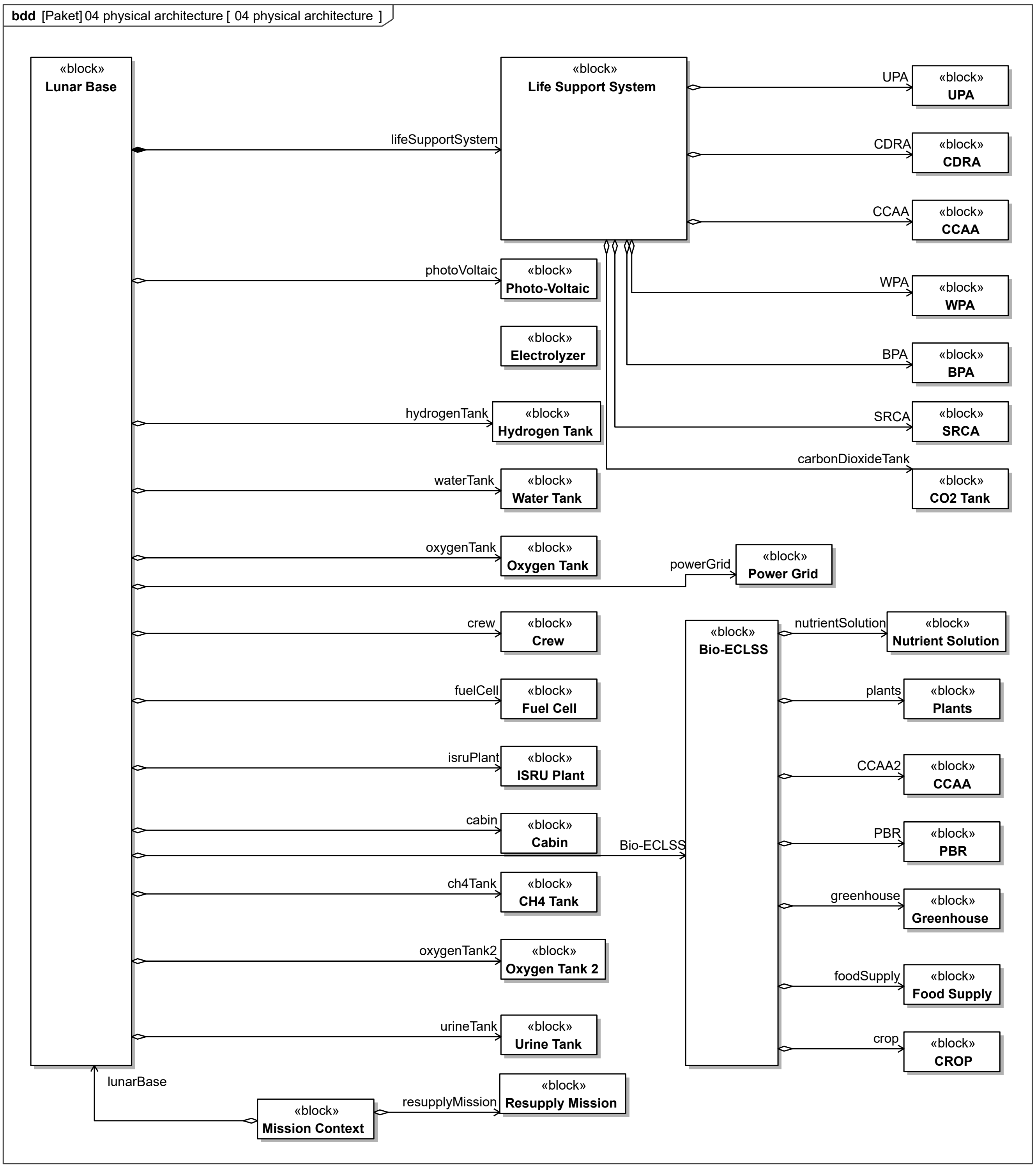


Figure A.2.: Hierarchy of the Lunar Base Physical Architecture

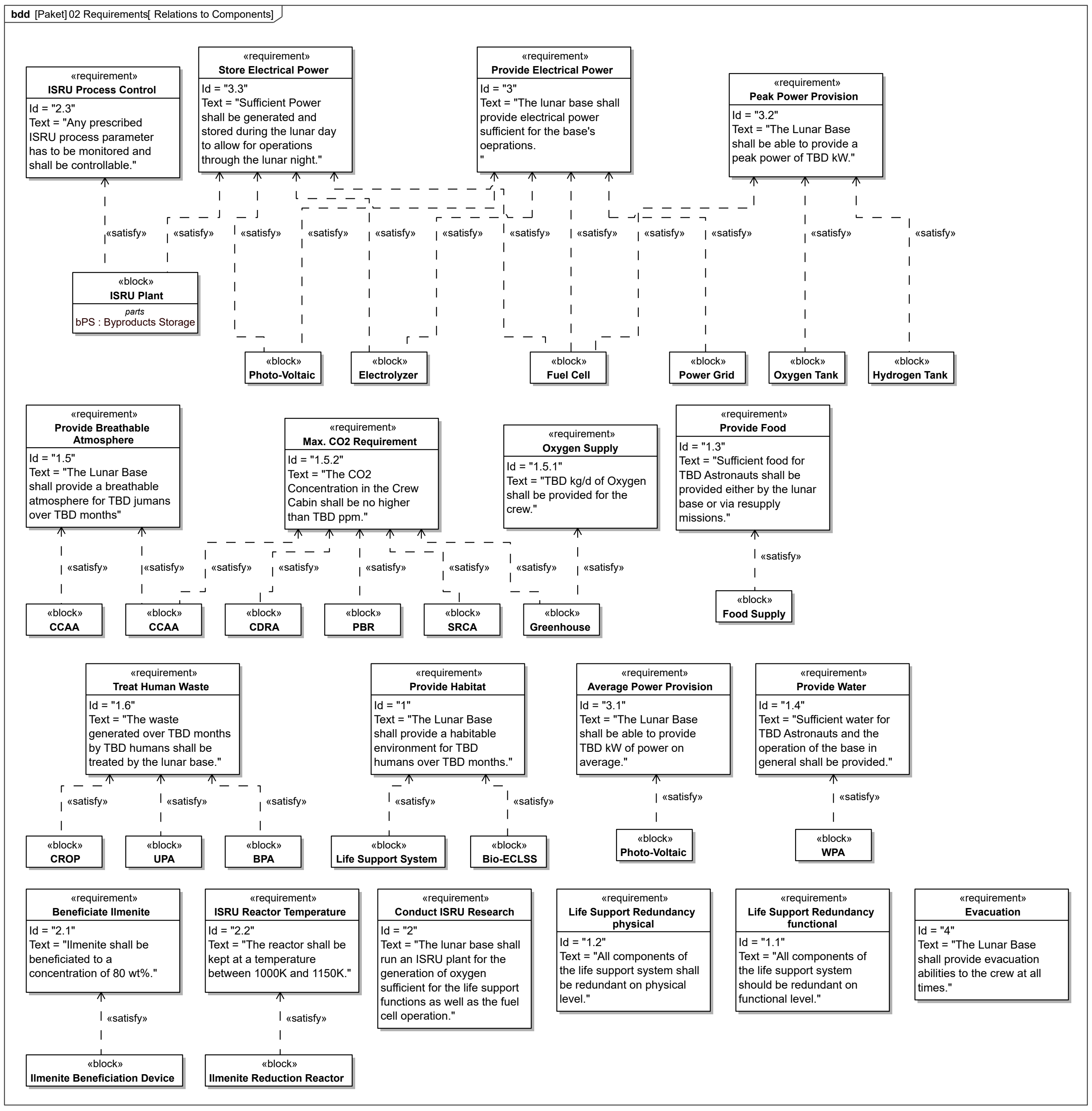

bdd [Paket] 02 Requirements[Relations to Components]
«requirement» ISRU Process Control
Id = "2.3"
Text = "Any prescribed ISRU process parameter has to be monitored and shall be controllable."
«requirement» Store Electrical Power
Id = "3.3"
Text = "Sufficient Power shall be generated and stored during the lunar day to allow for operations through the lunar night."
«requirement» Provide Electrical Power
Id = "3"
Text = "The lunar base shall provide electrical power sufficient for the base's oeprations. "
«requirement» Peak Power Provision
Id = "3.2"
Text = "The Lunar Base shall be able to provide a peak power of TBD kW."
«satisfy»
«block» ISRU Plant
parts
bPS : Byproducts Storage
«block» Photo-Voltaic
«block» Electrolyzer
«block» Fuel Cell
«block» Power Grid
«block» Oxygen Tank
«block» Hydrogen Tank
«requirement» Provide Breathable Atmosphere
Id = "1.5"
Text = "The Lunar Base shall provide a breathable atmosphere for TBD jumans over TBD months"
«requirement» Max. CO2 Requirement
Id = "1.5.2"
Text = "The CO2 Concentration in the Crew Cabin shall be no higher than TBD ppm."
«requirement» Oxygen Supply
Id = "1.5.1"
Text = "TBD kg/d of Oxygen shall be provided for the crew."
«requirement» Provide Food
Id = "1.3"
Text = "Sufficient food for TBD Astronauts shall be provided either by the lunar base or via resupply missions."
«block» CCAA
«block» CCAA
«block» CDRA
«block» PBR
«block» SRCA
«block» Greenhouse
«block» Food Supply
«requirement» Treat Human Waste
Id = "1.6"
Text = "The waste generated over TBD months by TBD humans shall be treated by the lunar base."
«requirement» Provide Habitat
Id = "1"
Text = "The Lunar Base shall provide a habitable environment for TBD humans over TBD months."
«requirement» Average Power Provision
Id = "3.1"
Text = "The Lunar Base shall be able to provide TBD kW of power on average."
«requirement» Provide Water
Id = "1.4"
Text = "Sufficient water for TBD Astronauts and the operation of the base in general shall be provided."
«block» CROP
«block» UPA
«block» BPA
«block» Life Support System
«block» Bio-ECLSS
«block» Photo-Voltaic
«block» WPA
«requirement» Beneficiate Ilmenite
Id = "2.1"
Text = "Ilmenite shall be beneficiated to a concentration of 80 wt%."
«requirement» ISRU Reactor Temperature
Id = "2.2"
Text = "The reactor shall be kept at a temperature between 1000K and 1150K."
«requirement» Conduct ISRU Research
Id = "2"
Text = "The lunar base shall run an ISRU plant for the generation of oxygen sufficient for the life support functions as well as the fuel cell operation."
«requirement» Life Support Redundancy physical
Id = "1.2"
Text = "All components of the life support system shall be redundant on physical level."
«requirement» Life Support Redundancy functional
Id = "1.1"
Text = "All components of the life support system should be redundant on functional level."
«requirement» Evacuation
Id = "4"
Text = "The Lunar Base shall provide evacuation abilities to the crew at all times."
«block» Ilmenite Beneficiation Device
«block» Ilmenite Reduction Reactor

Figure A.3.: Relation between requirements and components of the Lunar Base Model

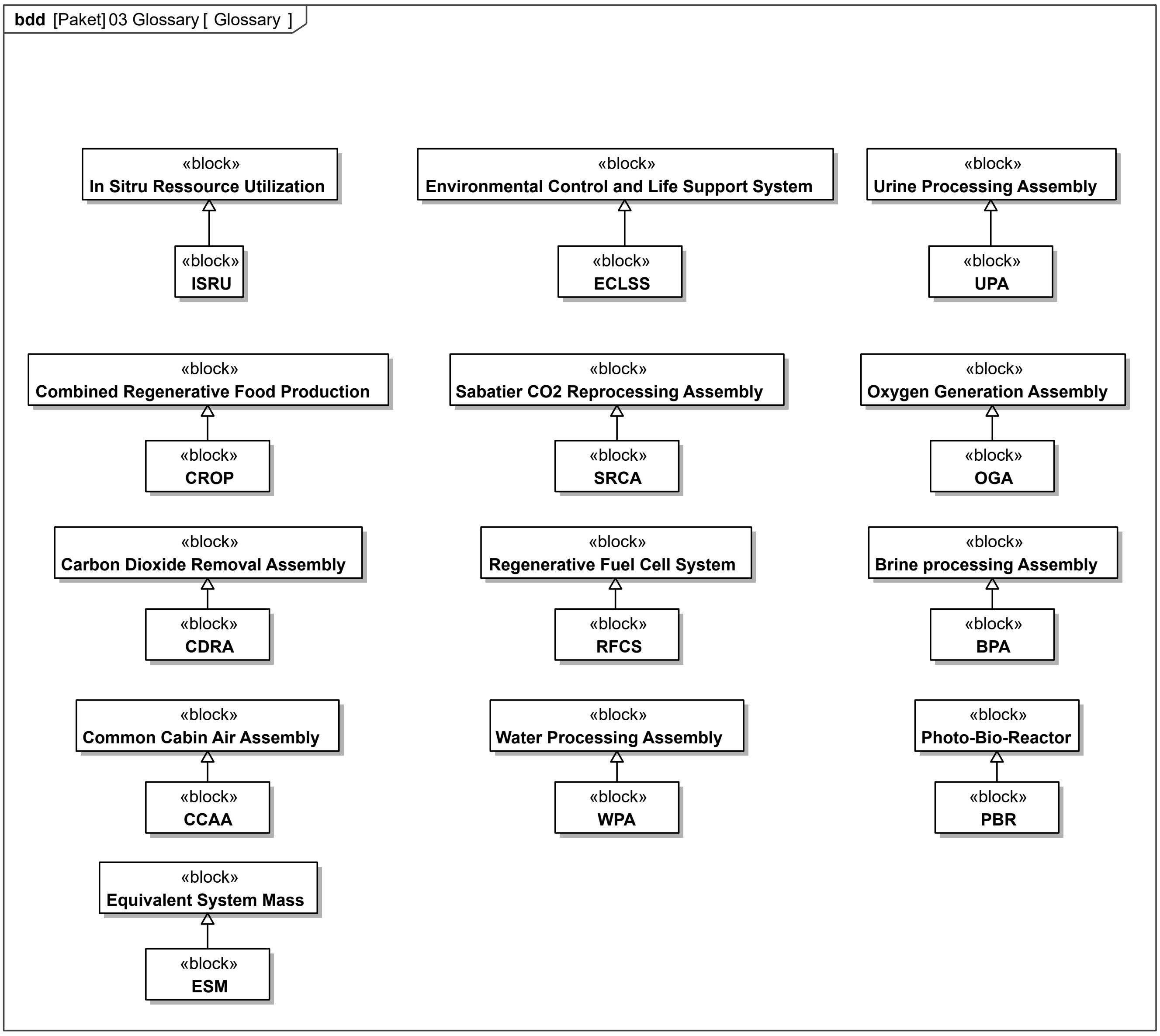

bdd [Paket] 03 Glossary [Glossary]
«block»
In Sitru Ressource Utilization
«block»
ISRU
«block»
Environmental Control and Life Support System
«block»
ECLSS
«block»
Urine Processing Assembly
«block»
UPA
«block»
Combined Regenerative Food Production
«block»
CROP
«block»
Sabatier CO2 Reprocessing Assembly
«block»
SRCA
«block»
Oxygen Generation Assembly
«block»
OGA
«block»
Carbon Dioxide Removal Assembly
«block»
CDRA
«block»
Regenerative Fuel Cell System
«block»
RFCS
«block»
Brine processing Assembly
«block»
BPA
«block»
Common Cabin Air Assembly
«block»
CCAA
«block»
Water Processing Assembly
«block»
WPA
«block»
Photo-Bio-Reactor
«block»
PBR
«block»
Equivalent System Mass
«block»
ESM

Figure A.4.: Glossary of the Lunar Base Model

B. Digital Appendix

The code written for this thesis is published under `https://gitlab.lrz.de/lrt/sysml_graph_analysis_tool`, including all magicdraw models utilized in the analyses.