Systems Engineering for University Satellite Projects
Engineering within Constraints

N. van der Pas

Master of Science Thesis
Systems Engineering for University Satellite Projects
Engineering within Constraints

MASTER OF SCIENCE THESIS

For the degree of Master of Science in Aerospace Engineering at Delft University of Technology

N. van der Pas

August 13, 2015

Faculty of Aerospace Engineering (AE) · Delft University of Technology
Abstract

Many universities have ventured into the design of satellites as educational projects. The Delft University of Technology (TU Delft) has a rich history in this field with the Delfi-C$^3$ and Delfi-n3Xt projects. Currently the Delfi programme is working on the DelFFi satellites, the third project in line. Applying and researching systems engineering has been a major part of these projects, and the current study continues in this tradition.

Systems engineering methodologies need to be adapted to their environment. Team members, both from Delfi projects as other projects, report that currently these methodologies do not function optimal.

The aim of this thesis is to propose a new systems engineering methodology for university satellite projects. This goal will be achieved by studying previous and current projects. In addition the stakeholder use cases will be examined to improve the success of the university satellite systems engineering methodology.

University satellite projects are undertaken for educational purposes. The teams consist for a large part of students, have a high turnover and fluctuate in size. The team members use these projects to perform research and receive an education. These needs to be integrated well into the project.

It has been found that the performance of the project can be parametrised by looking at the quality of the system, the project length and the project cost. The performance of the students in the project can be measured with their grades and the time they spent on their assignments in comparison to their peers.

Based on the stakeholder use cases and taking into account the main parameters, a university satellite Systems Engineering (SE) Methodology was developed. The University satellite methodology should consider the documents that students generate as part of their education as the main project documentation. The system can best be developed from a verified baseline. The system should be broken down into configuration items, which are developed in a number of consecutive work packages. These work packages are based on the life cycle of the students involved.
# Table of Contents

List of Acronyms xv

Acknowledgements xvii

1 Thesis Outline 1
  1.1 Previous Work & Research Motivation ............................................. 1
  1.2 Research Objectives ................................................................. 2
  1.3 Research Questions ................................................................. 2
  1.4 Research Approach ................................................................. 3
  1.5 Methodology Definition ............................................................ 3
    1.5.1 Stakeholder Representation ................................................... 4
  1.6 Structure of the Report ............................................................ 5

2 Systems Engineering in University Satellite Projects 7
  2.1 Characteristics of a University Satellite Project ............................ 7
  2.2 Objectives of a University Satellite Project .................................. 8
  2.3 Team characteristics ............................................................... 9
  2.4 Design characteristics ............................................................ 9
  2.5 Conclusion ................................................................................. 10

3 Systems Engineering in the Delfi Programme 11
  3.1 Programme Objectives ............................................................... 11
  3.2 Participants .............................................................................. 12
  3.3 Systems Engineering Philosophy ............................................... 12
  3.4 Configuration Management ....................................................... 12
  3.5 Requirements Management & Control ....................................... 13
  3.6 Design Control ........................................................................ 14
3.7 Model Philosophy .................................................. 15
3.8 Development Phases & Life Cycle Model ......................... 15
3.9 Change Management ................................................ 15
3.10 Interface Management .............................................. 16
3.11 Integration, Verification and Validation ......................... 16
3.12 Risk & Reliability Management ................................. 17
3.13 Team Structure & Characteristics ............................... 17
3.14 Meetings .......................................................... 17
3.15 Documentation .................................................... 18
3.16 Systems Engineering in Practice ............................... 19
3.17 Conclusions ....................................................... 19

4 Team research ......................................................... 21
4.1 Early observations .................................................. 21
4.2 Methodology ....................................................... 22
4.3 Data Analysis ...................................................... 22
  4.3.1 Project filter .................................................... 23
  4.3.2 Filter 1 .......................................................... 23
  4.3.3 Filter 2 .......................................................... 23
4.4 Results .............................................................. 24
4.5 Conclusions ......................................................... 27

5 University Satellite Project Context .................................. 29
5.1 Stakeholders ......................................................... 29
  5.1.1 Stakeholder Roles ............................................. 29
  5.1.2 Design Stakeholder Role .................................... 30
  5.1.3 Supplier Stakeholder Role ................................... 30
  5.1.4 External Stakeholder Role ................................... 32
  5.1.5 Secondary Stakeholder Role ................................. 33
  5.1.6 Customer Stakeholder Role ................................ 35
  5.1.7 Key Stakeholder Roles ..................................... 36
5.2 Life Cycles .......................................................... 36
  5.2.1 Student Life Cycles .......................................... 37
  5.2.2 System Lifecycles ............................................. 39
5.3 Information, Knowledge and Communication ..................... 41
5.4 Use Cases .......................................................... 42
  5.4.1 Engineering Use Cases ...................................... 43
  5.4.2 Management Use Cases ...................................... 44
  5.4.3 Role Related Use Cases .................................... 46
5.5 Key Parameters ..................................................... 48
  5.5.1 Student Key Parameters ..................................... 48
  5.5.2 Project Key Parameters ..................................... 49
  5.5.3 Research Key Parameters ................................... 50
  5.5.4 Key Parameter Discussion .................................. 50
5.6 Conclusions ......................................................... 51
# Systems Engineering Philosophy & Model Trade-off

## 6.1 Trade-off Methodology

## 6.2 Systems Engineering Philosophy

### 6.2.1 Criteria

### 6.2.2 Systems Engineering Philosophy Options

### 6.2.3 Trade-Off Conclusions

## 6.3 Model Approach

### 6.3.1 Criteria

### 6.3.2 Model Approach Options

### 6.3.3 Trade-off Conclusions

## 6.4 Conclusions

# The University Satellite Systems Engineering Methodology

## 7.1 Configuration Management

### 7.1.1 Configuration Item Definition

### 7.1.2 Configuration Item Properties and Links

### 7.1.3 Configuration Items, Hardware & Software

### 7.1.4 Configuration Item Identification

### 7.1.5 Configuration Item Breakdown

## 7.2 Work Packages

### 7.2.1 Work Package Identification

### 7.2.2 Work Package Control

## 7.3 The Baseline Life Cycle

### 7.3.1 Baseline Initialisation

### 7.3.2 Baseline Iterations

### 7.3.3 Payloads and the Baseline

## 7.4 Requirements Management & Control

## 7.5 Design Control

## 7.6 Documentation Control & Information Sharing

## 7.7 Documentation Storage

## 7.8 Reviews

### 7.8.1 Work Package Review

### 7.8.2 Baseline Reviews

## 7.9 Meetings

### 7.9.1 Team Meetings

### 7.9.2 Expert Meetings

## 7.10 Example: Deployable Solar Panels

---

Master of Science Thesis  
N. van der Pas
# Table of Contents

## 8 Conclusions & Recommendations

8.1 Conclusions

8.1.1 University Satellite Projects

8.1.2 University Satellite Context

8.1.3 University Satellite Methodology

8.2 Recommendations & Discussion

## A Systems Engineering in the Delfi programme

A.1 Programme Objectives

A.2 Participants

A.3 Systems Engineering Philosophy

A.4 Configuration management

A.5 Requirements Management & Control

A.6 Design Control

A.7 Model Philosophy

A.8 Development Phases & Life Cycle Model

A.9 Change Management

A.10 Interface Management

A.11 Integration, Verification and Validation

A.12 Risk & Reliability Management

A.13 Team Structure & Characteristics

A.14 Meetings

A.15 Documentation

A.16 Systems Engineering in Practice

A.17 Conclusions

## B Delfi Team Research

B.1 Data Collected

B.2 Data analysis

B.2.1 Project filter

B.2.2 Filter 1

B.2.3 Filter 2

B.2.4 Student Levels

B.3 Results

B.3.1 Team Size

B.3.2 Team Member Characteristics

B.4 Conclusions

## C Recommendations Elstak

N. van der Pas

Master of Science Thesis
D  Trade Off Comparisons  155
   D.1  Systems Engineering Philosophy  155
      D.1.1  Criteria Comparison  155
      D.1.2  Options Comparison  156
   D.2  Model Approach  166
      D.2.1  Criteria Comparison  166
      D.2.2  Options Comparison  166
E  Digital Files  171

Index  179
List of Figures

1.1 Breakdown of a methodology .............................................. 4
1.2 Inputs and outputs of a methodology ..................................... 5
1.3 Abstract representation of the report .................................... 6
3.1 Generic configuration item breakdown ................................. 13
3.2 Diagram of a generic team in the Delfi projects ...................... 17
4.1 Flowchart project filter .................................................... 23
4.2 Filter influence ............................................................... 24
4.3 Amount of people related to Delfi projects ......................... 25
4.4 Boxplot of the estimated team size during the program and the different projects. 26
4.5 Statistics on man-months per person .................................... 26
5.1 Diagram of Stakeholder Roles ............................................. 30
5.2 Breakdowns of ‘Design’ and ‘Supplier’ Stakeholder roles .......... 31
5.3 Breakdown of the external stakeholder role ......................... 32
5.4 Breakdown of the ‘Secondary’ stakeholder role ..................... 34
5.5 Breakdown of the ‘Customer’ stakeholder role ...................... 35
5.6 Life cycle of a MSc student in theory .................................... 37
5.7 Life cycle of a MSc student in practice ................................ 38
5.8 Life cycle of an intern ....................................................... 38
5.9 Life cycle of a generic course ............................................ 39
5.10 Comparison of life cycles ................................................ 40
5.11 Information, documentation and communication context ........ 42
5.12 Use cases related to design control .................................... 43
5.13 Use cases related to information ........................................ 44
5.14 Use cases related to integration and deployment .................... 45
B.2 Distribution of data points over names ........................................ 140  
B.3 Distribution of names by amount of data points ............................. 140  
B.4 Flowchart project filter ............................................................... 141  
B.5 Filter influence .......................................................................... 142  
B.6 Amount of people related to Delfi ................................................ 144  
B.7 Amount of people related to Delfi projects .................................... 145  
B.8 Histogram of the team size during the whole programme and the different projects 145  
B.9 Boxplot of the estimated team size during the program and the different projects. 146  
B.10 Boxplot on estimated team size divided over the different years. ........ 147  
B.11 Boxplots on estimated teamsize divided over the months. ............... 147  
B.12 Statistics on man-months per person .......................................... 148  
B.13 Pie charts of the distribution of the team members over the student levels. 149  
B.14 Weighted team member distribution ......................................... 150

C.1 Project Phases as proposed by Elstak ............................................. 154
List of Tables

5.1 Seven life cycle phases as defined by ECSS and NASA ........................................ 39
6.1 Scoring table for qualitative comparisons in the trade-offs ................................. 54
6.2 Random index ....................................................................................................... 54
6.3 Weights of criteria in systems engineering philosophies trade off ......................... 57
6.4 Results for the systems engineering philosophies trade off ................................ 60
6.5 Weights of criteria in the model approach trade off ............................................. 62
6.6 The CubeSat Model Approach as proposed by Elstak ......................................... 64
6.7 Scores of the model approach trade off ............................................................... 65
6.8 Traceability between trade off criteria and the key parameters ............................ 65
7.1 Configuration items for the ‘deployable solar panel iteration’ example .................... 84
7.2 Work Packages for the ‘Deployable solar panel iteration’ example ......................... 85
7.3 To Be Determined (TBD)s for the ‘Deployable solar panel iteration’ example ....... 86
A.1 Power and mass margins in the Delfi-C³ project .................................................. 105
A.2 Power and mass margins in the Delfi-n3Xt project ............................................. 106
A.3 The CubeSat Model Approach as proposed by Elstak .......................................... 108
B.1 Distribution of Datapoints over source location .................................................... 139
B.2 Table of Student Level Categories .................................................................... 143
C.1 The CubeSat Model Approach as proposed by Elstak .......................................... 153
D.1 Systems engineering philosophy criteria comparison ............................................. 156
D.2 Systems engineering philosophy option performance on criteria SEP-C-1 ............ 157
D.3 Systems engineering philosophy option comparison on criteria SEP-C-1 .......... 157

Master of Science Thesis

N. van der Pas
D.4 Systems engineering philosophy option performance on criteria SEP-C-2 . . . . . . . 158
D.5 systems engineering philosophy option comparison on criteria SEP-C-2 . . . . . . . 158
D.6 Systems engineering philosophy option performance on criteria SEP-C-3 . . . . . . 159
D.7 systems engineering philosophy option comparison on criteria SEP-C-3 . . . . . . . 159
D.8 Systems engineering philosophy option performance on criteria SEP-C-4 . . . . . . 160
D.9 Systems engineering philosophy option comparison on criteria SEP-C-4 . . . . . . . 160
D.10 Systems engineering philosophy option performance on criteria SEP-C-5 . . . . . . 161
D.11 systems engineering philosophy option comparison on criteria SEP-C-5 . . . . . . . 161
D.12 Systems engineering philosophy option performance on criteria SEP-C-6 . . . . . . 162
D.13 Systems engineering philosophy option comparison on criteria SEP-C-6 . . . . . . . 163
D.14 Systems engineering philosophy option performance on criteria SEP-C-7 . . . . . . 163
D.15 Systems engineering philosophy option comparison on criteria SEP-C-7 . . . . . . . 164
D.16 Systems engineering philosophy option performance on criteria SEP-C-8 . . . . . . 164
D.17 Systems engineering philosophy option comparison on criteria SEP-C-8 . . . . . . . 165
D.18 Model Approach criteria comparison ................................................................. 166
D.19 Model Approach option performance on criteria MA-C-1 .................................. 167
D.20 Model Approach option comparison on criteria MA-C-1 .................................... 167
D.21 Model Approach option performance on criteria MA-C-2 .................................. 168
D.22 Model Approach option comparison on criteria MA-C-2 .................................... 168
D.23 Model Approach option performance on criteria MA-C-3 .................................. 169
D.24 Model Approach option comparison on criteria MA-C-3 .................................... 169
D.25 Model Approach option performance on criteria MA-C-4 .................................. 170
D.26 Model Approach option comparison on criteria MA-C-4 .................................... 170
List of Acronyms

3mE  Mechanical, Maritime and Materials Engineering
AAU  Aalborg University
AE   Aerospace Engineering
AIV  Assembly, Integration and Verification
ADCS Attitude Determination and Control System
AHP  Analytical Hierarchy Process
AWSS Autonomous Wireless Sun Sensors
BSc  Bachelor of Science
BEng Bachelor of Engineering
CCB  Configuration Control Board
CDR  Critical Design Review
COTS commercial of the shelf
CalPoly California Polytechnic State University
DIMES Delft Institute of Microsystems and Nanoelectronics
DSE  Design Synthesis Exercise
EEMCS Electrical Engineering, Mathematics and Computer Sciences
ECSS European Cooperation for Space Standardization
EM   Engineering Model
EPS  Electrical Power Subsystem
ESA  European Space Agency
FM   Flight Model
FMECA Failure Mode, Effect and Criticality Analysis
I2C  Inter-Integrated Circuit
ICD  Interface Control Document
ISIS Innovative Solutions In Space BV
ISL  Innovative Space Logistics BV
KISS Keep It Simple Stupid
LEOPS Launch and Early Operations
LOS  Launch & Orbit Segment
MBSE Model Based Systems Engineering
MDR  Mid-term Design Review
MSc  Master of Science
NASA National Aeronautics and Space Administration
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLR</td>
<td>Nationaal Lucht- en Ruimtevaartlaboratorium (National Aerospace Laboratory)</td>
</tr>
<tr>
<td>OBC</td>
<td>On-Board Computer</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed Circuit Board</td>
</tr>
<tr>
<td>PDR</td>
<td>Preliminary Design Review</td>
</tr>
<tr>
<td>PERT</td>
<td>Program Evaluation and Review Technique</td>
</tr>
<tr>
<td>PhD</td>
<td>Doctor of Philosophy</td>
</tr>
<tr>
<td>PMSE</td>
<td>Project Management and Systems Engineering</td>
</tr>
<tr>
<td>RAMS</td>
<td>Reliability, Availability, Maintainability and Availability</td>
</tr>
<tr>
<td>SHOTS</td>
<td>Space Hardware of the Shelve</td>
</tr>
<tr>
<td>SPF</td>
<td>Single Point Failure</td>
</tr>
<tr>
<td>SE</td>
<td>Systems Engineering</td>
</tr>
<tr>
<td>SSE</td>
<td>Space Systems Engineering</td>
</tr>
<tr>
<td>TBD</td>
<td>To Be Determined</td>
</tr>
<tr>
<td>TFSC</td>
<td>Thin Film Solar Cells</td>
</tr>
<tr>
<td>TNO</td>
<td>Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek (Netherlands Organisation for applied Scientific Research)</td>
</tr>
<tr>
<td>TRL</td>
<td>Technology Readiness Level</td>
</tr>
<tr>
<td>TU Delft</td>
<td>Delft University of Technology</td>
</tr>
<tr>
<td>UT</td>
<td>University of Twente</td>
</tr>
<tr>
<td>VKI</td>
<td>Von Karman Institute</td>
</tr>
<tr>
<td>V&amp;V</td>
<td>Verification and Validation</td>
</tr>
</tbody>
</table>
Acknowledgements

After this long and rocky road I would like to thank my supervisor Dr. Jian Guo for his assistance during this thesis. The experience of being in the Delfi team has taught me a lot about project management. And if things did not go according to plan, I could vent to my fellow thesis students.

Special thanks go out to my family, I could not have done it without their support. Joy was there for me when I needed it most. Let me end with a shout out to Amy, for telling me to stop writing, Thijs, for the bouldering, and Frederik, for the rock ’n roll.

Delft, University of Technology
August 13, 2015

N. van der Pas
“Roeien met de riemen die je hebt”
“Rowing with the oars you have got”
— Dutch proverb
Chapter 1

Thesis Outline

This chapter will briefly present the research performed in this thesis and the structure of this report. A short overview of the predecessors of this work is given in Section 1.1. The motivation for this research will be discussed in the same section. Subsequently, section 1.2 will discuss the research objective of this study, and Section 1.3 will discuss the related research questions. The approach taken to answer these questions is given in Section 1.4. Section 1.6 gives an overview of the structure of the complete report.

1.1 Previous Work & Research Motivation

Student satellites have a long history. The first university class satellite was launched in 1970 by the University of Melbourne [1]. In the previous decade there has been a significant amount of growth of satellite of these kind [1]. It should come as no surprise that systems engineering in these projects has been researched before. The application of systems engineering in university class satellite projects differs based on the background of the universities conducting the projects [2]. Results of the implementation of engineering and management methods are reported upon [3, 4, 5], however the systems itself receive more coverage. An interesting study has been conducted in the implementation of Model Based Systems Engineering (MBSE) for a university satellite by Nottage and Corns [6], which has led to the inclusion of MBSE in the trade off study in Chapter 6.

The satellite programme of the Delft University of Technology (TU Delft) has always actively implemented systems engineering. Bonnema [7] and Rotteveel [8] described the approach implemented in the Delfi-C³ project. This approach was based on well known industry handbooks and standards. During the project Elstak conducted a study which compared the approach for the Delfi-C³ satellite with the systems engineering approach taken for AAUSAT-II of Aalborg University (AAU) [9, 2]. His main conclusions are summarised in Appendix C.

The approach taken for Delfi-C³ was evaluated at the end of the project [10, 11]. Based on the experience with Delfi-C³ the systems engineering methodology was adapted for Delfi-n3Xt [10, 12]. Additionally an approach was defined for the interface control by Lebbink based on
experience in the Delfi-C$^3$ programme [13, 14, 15]. An analysis of this approach during the final phases of Delfi-n3Xt was performed by Boerci [16].

Even with all this research, the Delfi programme still had not mastered Systems Engineering (SE). Early observations showed that students were not satisfied with the integration of their work within the project. On top of that a number of key elements of SE were not performed. It was clear that the right methodology was not found yet. Conversations with team members from other satellite projects showed that they experienced similar problems. SE methodologies need to be implemented in and adapted to their environment [17]. Since the environment for these university projects is very different from industry, a dedicated methodology is needed.

The current work partly continuous in the tradition of observing the SE in the satellite projects. The work of the preceding studies form the basis of Chapters 2 and 3. In this thesis the systems engineering methodologies and implementations in the Delfi-C$^3$, Delfi-n3Xt and DelFFi project are researched. Additionally a study is conducted into the stakeholders and use cases of such a methodology. This should result in a SE methodology that fits the users needs and thus will likely achieve a easy adoption in the engineering team. The methodology proposed in this study should be more general applicable, due to the consideration of the stakeholder use cases.

## 1.2 Research Objectives

Based on the early observation described in Section 1.1, the following research objective is formulated:

The research objective is to propose a systems engineering methodology for university satellite projects, by studying previous university satellite projects, assessing their context and assessing key stakeholder use cases.

The aim of this research is thus to propose a systems engineering methodology. The definition of a systems engineering methodology is described in Section 1.5. The methodology will be designed for university satellite projects. These projects have specific characteristics which makes them different from other satellite projects, such as the involvement of student. These characteristics will be further detailed in Chapter 2. The systems engineering methodology developed in this thesis will be referred to as the ‘university satellite systems engineering methodology’. The research objective will be achieved by answering the research questions, which can be found in Section 1.3 and following the approach presented in Section 1.4.

## 1.3 Research Questions

The research objective will reached by answering the research question presented in this section. Three broad questions are posed in order to capture the different elements of the research.
1.4 Research Approach

The approach to this research is to first study the systems engineering approach and context of university satellite projects. An in-depth literature study will be performed on document archive of the Delfi programme. This will be amended with notes from previous team members if possible. In parallel observations will be made of the ongoing DelFFi project. This will be done by participating in meetings and questioning the team members in the project. Their experience highlight current issues and their needs form the basis of the university satellite SE methodology.

Second the stakeholders and context of the project will be mapped. These are based on the analysis in the previous part of the study. The use cases and life cycles of these stakeholders will be distilled from this study to investigate the needs of the stakeholders and to discuss what constraints they bring to the project. The result will set the frame in which the methodology can be designed. Trade-offs will be performed for major elements of the methodology. The needs of the project and people will need to be satisfied by these tools and processes. These tools and processes are based on the experience in previous project, recommendations from previous research and related research.

1.5 Methodology Definition

In order to understand what is being designed in this thesis, it is important to understand how a methodology is defined in this context.

A methodology consists of processes, tools and predefined documentation, as shown in Figure 1.1 [17]. Some sources also mention methods as part of the methodology [18, 17], however these sources state that a method is a kind of process, specifically a process for a process.
For simplicity, methods are not considered as a separate element. Documentation is often not considered as a separate element, and can be viewed as a specific type of tool. Research, presented in Chapter 3, showed the prominent role documentation plays in university satellite projects, and therefore it is considered as a separate element. Documentation is also one of the outputs of a methodology as shown in Figure 1.2, which separates it from a regular tool.

The methodology interacts with the context of the project, such as the people involved. Processes and tools are defined as follows:

**Process** A process is a set of interrelated or interacting activities that transform input into outputs. [19, 18]

**Tool** A tool supports a process.

To better view the aim of the methodology, it is important to know what it should result in. In Figure 1.2 this is visualised. The methodology should result in a system and documentation about that system. The system is based on needs and will use resources, such as manpower. Stakeholders who have secondary stakeholder roles (see Section 5.1.1) will generate their own documentation by applying the methodology. For example, a thesis student will generate a thesis report while being involved in the project, and thus while applying the methodology. This information might be related or part of the system information, but since it is generated for different purposes, it is visualised separately.

### 1.5.1 Stakeholder Representation

The SE methodology has to be implemented by the satellite design team, without their cooperation the methodology is an empty shell. The stakeholders, including the design team, will be represented by stakeholder roles, as will be explained in detail in Chapter 5.1.1.
To represent the stakes of the stakeholders the following design principles, as given by de Bruijn and Herder [20], will be adapted:

- The methodology should give the stakeholders a change to realize their own interests.
- The methodology should protect the core values of each stakeholder.
- The methodology should contain incentives for the stakeholders to make progress.
- The outputs of the methodology should be able to withstand expert scrutiny.

The first three items will be satisfied by using stakeholder roles that represent the needs of the stakeholders unrelated to the project (see Section 5.1.1). The fourth item refers to the outputs of the methodology, which are shown in Figure 1.2. Review of the system and project documentation will be addressed both by elements of the university satellite methodology. The secondary documents, for examples journal papers and graduation reports, are currently already assessed by experts, such as review and graduation committees.

1.6 Structure of the Report

The report is divided up into 8 chapters, including this outline chapter and the final concluding chapter. The information flow throughout the report is shown in Figure 1.3. Chapter 2 defines what university satellites are and describes the context of university satellites projects. A case study of the university satellite projects at TU Delft is presented in Chapter 3. This chapter will describe what went well and less well in these projects. Chapter 4 conclusions from research on teams in the Delfi projects is presented, the complete research can be found in Appendix B. The stakeholder roles, life cycles and use cases that the university satellite methodology needs to take into account are described in Chapter 5. Key parameters which can describe the success of a new methodology are also described in this chapter. These four chapters together describe the history and context of university satellite projects. They form
the basis on which a new methodology can be designed and describes the context of these projects.

Onwards from Chapter 6 the new university satellite systems engineering methodology will be defined. Using the parameters from Chapter 5, a number of trade-offs on important elements of the new methodology are made in Chapter 6. Details of these trade-offs can also be found in Appendix D. These decisions, together with the conclusions from Chapters 2, 3 and 4 and the context described in Chapter 5, form the basis for the new methodology described in Chapter 7. The main conclusions and recommendations from the complete thesis will be given in Chapter 8.
This chapter will address the general challenges a university satellite project faces. Note that the systems engineering which is applied in the Delfi programme is discussed separately in Chapter 3. The university satellite is a growing method of education. In the last decade the number of university satellite projects are on the rise [1], spurred the CubeSat standard [21] and by programs from European Space Agency (ESA)[22, 23], National Aeronautics and Space Administration (NASA)[24] and Von Karman Institute (VKI)[25].

2.1 Characteristics of a University Satellite Project

There is a wide range of systems that can be defined as university satellites and satellite systems. This term can be defined in a number of ways, for example:

- A university team can develop a satellite, like for example the Delfi-C$^3$ satellite. The University takes the initiative and finds partners for the project. Education is on of the main objectives of the project.
- A university can develop a satellite for an agency or institute. An example of this is the Solar Probe Plus mission from NASA, which is produced by the Johns Hopkins University Applied Physics Laboratory [26].
- A university can be involved in the development of a space instrument. Several Universities were involved in the consortium that developed the SPIRE instrument of the ESA Herschel Space Observatory. For example, the lead of this project was Cardiff University [27].
These projects can range from student driven projects to very professional and high-end engineering. From Section 1.1, it follows that this work is interested in projects that involve students, thus this thesis will limit itself to the university-class satellites. As defined by Swartwout [28], a University-class satellite has the following three characteristics:

1. The system under design is a functional spacecraft, rather than a payload instrument or component. To fit the definition, the system must operate in space with its own independent means of communications and command. Self-contained systems that are attached to other vehicles are allowed under this definition.
2. Untrained personnel (i.e. students) perform a significant fraction of key design decisions, integration & testing activities, and flight operations.
3. The training of these people is as important as (if not more important) the nominal 'mission' of the system itself.

The main documentation of these projects are student reports [29], which only reflect the design at a given time and thus not necessarily the final system.

It is important that all phases of the project work can be academically credited, else this will cause dependence on volunteers in certain phases and imbalance in the workforce [9]. The projects typically know a short development time [9].

2.2 Objectives of a University Satellite Project

It is generally recognised that university satellite projects aim to give their students experience with the design and production of a real satellite mission [2, 28, 29, 30, 31]. The projects offer educational benefits in the form of hands on activities [28]. The projects also serve as inspiration for students [28].

There are also research objectives to achieve and technology to be demonstrated, though these are not present in some missions. Additionally a number of satellites provide communication capabilities for radio amateurs and possibly other services. [1].

An important observation to make is that the educational objectives are set by the educational institute. This means that for a university the mission can be a success even if the mission does not launch [4], however for sponsors and payload providers these objectives will not necessarily form the drivers [1, 30]. It should be noted however that the space industry likely benefits from university satellite students entering the work force with hands on experience [32].

An exiting new technology demonstration payload at mission initiation could be hampered irrelevant at mission completion due to schedule slips, such was seen in Sapphire [33] and Delfi-n3Xt \(^1\). Thus Specially for the technology demonstrator mission it is important to launch on time.

\(^1\)From conversation with Jasper Bouwmeester
2.3 Team characteristics

University satellite teams have a number of characteristics:

- University satellite teams consist of inexperienced students [7].
- The teams are often small [29].
- There is a high and regular turnover of team members [4, 6, 9, 14, 29, 34].

Projects had difficulty maintaining a consistent output and tracking information over the years [4] and ‘educational’ errors cannot be excluded [9]. This results in a need for good documentation [4, 14, 34]. The Icarus student satellite project [4] involved students early in their education and found this protected the continuity of the design. The UNISAT programme [30] avoided the continuity problems by limiting the project development time to the length of a the university courses, 2 years in this case.

Team members often volunteer themselves to the project [4, 34]. The team members you get, might not be the team members you need. The expertise of the student depends heavily the university and courses offered there [9, 14]. The ideal team configuration is likely never achieved [34]. The lines of communication within the teams can be short [14]. The schedule of the students involved can conflict with the schedule of the project [4]. In general there is limited availability of manpower [9, 14, 31].

The level of the students involved in the project differs, some involve mainly undergraduates [4], some mainly graduate students [2](see also Appendix B) and others PhD and graduate students [30]. Again others, like ESTCube, involved students throughout their educational career\(^2\). The staff members involved can have different roles as well, in some project they participate in the design, while in others they only have a advisory and management role [9]. The way students are involved in the projects also differ. In the AAUSAT-II project students were involved as part of group assignments, while for Delfi-C\(^3\) students were involved for thesis or internship assignments [9]. The former gave a stable workforce during the design phase [9], however, it should be noted that the integration phase was on a voluntary bases.

2.4 Design characteristics

The design of a university satellite bus remains conservative [29] and often use existing or commercial of the shelve (COTS) components[1, 29, 30] in order to achieve the mission objectives in the short term. However the satellites can carry innovative payloads [29, 32] and universities aim for innovation [14]. Delfi-C\(^3\) for example combined an acquired the on board computer, while also carrying the first linear transponder for nano-satellites [11]. The satellite are lightweight and small [1, 29], and in recent years a majority is built to the CubeSat standard [21, 32, 35].

The system is designed with limited financial resources [14, 30, 31]. Design decisions can also be influenced by political and social factors, though this is not always documented [31]. For

\(^2\)Conversation with ESTCube member
example, a certain payload can be chosen to strengthen the relations between the university and a company.

Even though a lot of university class satellites carry commercial non-space qualified components, the major failures occur at system level [36]. Component level failures cannot be entirely neglected, specially when the resources to address these are available or for missions with a long lifetime [36].

Project Management and Systems Engineering (PMSE) is applied differently in different teams. Depending on the level of PMSE education given to the students in the usual curriculum, PMSE is applied to the projects [2]. Within the Delft University of Technology (TU Delft) PMSE is taught extensively, and as a result PMSE is applied explicitly in projects [9].

For the PMSE of student satellites Elstak [9] advised the define the project organisation in the early phases of the project. Elstak notices that it is very desirable that all system elements mature at an equal rate. [9]

2.5 Conclusion

University class satellites offers hands on education in satellite engineering. This is one of the main motivators to undertake such a project. Other objectives vary per mission and can include performing science, demonstrating technologies and providing services.

The teams are small, consist of students and have a high turnover. The level of the students is project dependent.

Due to the limited resources the satellite often use COTS components. Nowadays many of the satellites follow the CubeSat standard. PMSE is applied differently in different projects depending on the context.
Chapter 3

Systems Engineering in the Delfi Programme

This chapter will present research done on the systems engineering processes as performed in the Delfi programme. A complete comparison can be found in Appendix 3.

The Delfi programme is used as a study case to research the implementation of systems engineering in a university context and to document how this is implemented in practice. The Delfi programme has been chosen because it is the satellite programme of the Delft University of Technology (TU Delft) and extensive internal documentation is available on these projects. This chapter will focus on several topics and describe how these were handled in the different Delfi projects. Note that the Delfi-C$^3$ and Delfi-n3Xt projects have been completed, albeit the latter with a redesign. The DelFFi project is in progress during this research, and thus not all aspects of systems engineering on this project can be described.

3.1 Programme Objectives

The objectives of the Delfi satellites are divided in three categories; educational, technical and programme objectives. These objective are set by the Delfi programme [37].

The technical objectives are project dependent, and aim to test new technologies. An example of a technological objective is the testing of the wireless sun sensor on the Delfi-C$^3$. The educational objective aims to give students hands on experience in teamwork and with satellite development. These objectives thus represents the student needs in these projects. In practice the projects facilitate thesis work and internships [12, 37, 38]. The programme objectives aim to mature the nano-satellite platform, and are thus related to engineering research. In the Delfi-n3Xt project this was for example the development of an attitude control system. Delfi-C$^3$ was also supposed to be an inspiration for students [11]. The technical objectives got more attention, and the programme objectives less. For DelFFi also a scientific objective is set, which could indicate a maturing platform, but could also be motivated by the partnership with science project QB50.
3.2 Participants

This section will describe the most important participants in the projects and their role. Naturally, all projects were conducted at the TU Delft, with students of the TU Delft.

The Delfi satellites were engineered at the TU Delft, with the chair of Space Systems Engineering (SSE) involved as the project lead. Other faculties were involved to provide specific expertise, specially strong cooperation with Electrical Engineering, Mathematics and Computer Sciences (EEMCS) was seen as mandatory [37] and inherent [7] to the project.

All Delfi satellites carried payloads for other partners. Some testing was performed at external institutes. Solar cells and the launch were brokered through external partners for all missions [12, 16]. With every new satellite, more subsystems were procured. The DelFFi mission is an exceptional case in that it has one partner (QB50/Von Karman Institute (VKI)) which provides payloads, launch and requirements [38, 39].

What is notably missing, is the involvement of staff as researchers. The satellites did not carry a research payload directly related to research of university staff or Doctor of Philosophy (PhD) students nor was their research directly connected to the satellites. The propulsion system on DelFFi was research pursued by the University, however the development was done completely by students.

3.3 Systems Engineering Philosophy

While an systems engineering philosophy could give direction to project, really only for Delfi-C3 the approach to the engineering of the satellite was explicitly stated [7]. The mission would be low cost, use commercial of the shelve (COTS) components, according to the Keep It Simple Stupid (KISS) philosophy and would not depend on new technology.

Also in this project a "flexible, on-the-fly (systems) engineering" approach was chosen, which meant that standard systems engineering processes would be tailored, while the project was progressing when issues occurred [7]. There has been no analysis of this approach. From a systems engineering point of view this looks like bad practice, since problems would occur before the processes to mitigate the problems were established.

After evaluation of Delfi-C3, it was concluded that the KISS approach could not successfully be applied to all subsystems; a design decisions that seemed simple in an early phase could lead to an increase in complexity in later phases [11].

3.4 Configuration Management

All three projects used configuration items to make a breakdown of the project. Configuration items can be defined as dependant, disciplinary homogeneous items which all-together encompass the complete system [40].

The breakdown was similar in all projects, and is shown in Figure 3.1 for the top levels. Though given the definition of configuration items (see A.4 and literature [41, 42] ), some of the items could be identified in more proper way. For example, the ground segment should be
split in a verification segment and a ground station segment since both part are independent and serve different purposes.

In Delfi-C$^3$ the configuration items were used to manage the engineering and control the status of the design [7]. Formalised processes were used for this control. Besides the configuration item breakdown a hardware breakdown was used [7, 10]. While the former was used to relate to requirements, the latter was used more by team members, which caused problems during the design [10].

In all projects the configuration items were related to the requirements [10, 40]. In Delfi-n3Xt the configuration items were assigned to a single responsible person [40]. In DelFFi it was attempted to related configuration items to work packages as well[39].

The implementation of configuration items seems to have a positive impact on the engineering process, but the exact form and impact they should have is debatable. It is clear that the configuration items give a breakdown of the structure which can be related to work to be done and can be used to connect requirements, hardware and work packages.

### 3.5 Requirements Management & Control

Requirements are central in the design, they need to be defined well and can change throughout the engineering process.

Both for Delfi-C$^3$ as Delfi-n3Xt an ambitious approach was taken to requirements [7, 43], only to be replaced by a simpler approach later[44]. In both projects the requirements were collected in a single file and categorised. In Delfi-C$^3$ a traceability matrix was set up, but rarely used later on in the project.

Initially in the Delfi-n3Xt project this traceability was captured within the requirements list [43]. For this particular approach forms needed to be submitted to the responsible engineer when new requirements needed to be added or changed. It is suspected that for Delfi-n3Xt this approach failed because of the cumbersome requirement change process, which put the requirement control strictly under responsibility of a single person and added additional tasks to the other team members. The requirements list was replaced with a simpler version, which linked the requirements with their configuration items and ordered the requirements hierarchically[44]. This seemed to have been successful.
The final Delfi-n3Xt approach to requirements was also used in DelFFi. For DelFFi the requirement management was also strictly the responsibility of a single position. However the requirements of DelFFi were incomplete and not updated for long periods of time when this position was not filled, which led to frustration with the team members. Conversations with the team members showed that they were often struggling to understand the context of their design, due to the lack of proper requirements.

The requirement process is important and central in systems engineering. Requirements are desired by the team members, however, the requirement control process should not pose too many additional tasks on the team members. Dependence of the requirements on a single engineer should be avoided.

An important observation is also that none of the projects related the requirements to a verification approach. Relating requirements to verification within the requirement definition process can lead to better requirements. In this way the verification processes will also take shape during the early phases of the design, which could make the planning more robust.

### 3.6 Design Control

In all projects budgets were used to control the design. Mass, power, link and thermal budgets have been used, the latter two will not be further discussed because they are subsystem specific. It has been noted that the budget are central in the design process.

The mass budget related to a hardware components, and tabulates the following characteristics:

- amount
- unit mass
- best estimate mass (of all components)
- maturity
- contingency (related to maturity)
- allocated mass (best estimate mass including contingency)

It should be noted that amount, unit mass and maturity are input values, while all others are calculated based on these values.

The power budget is approached differently in the different project:

- Delfi-C\(^3\) takes the most detailed approach, calculating the power consumption from the current that the components draw.
- In Delfi-n3Xt the power consumptions are given per part of the subsystem.
- And in DelFFi the subsystem power consumption was given as a percentage of the nominal power consumption.

In both Delfi-C\(^3\) and Delfi-n3Xt the power has, similar to the mass budget, contingencies to take into account uncertainties. What all budgets share is that the power consumption is related to the different subsystem and system modes. Similarly, the power budgets also include how much power the satellite can generate or store.

---

1Conversation with J. Bouwmeester
3.7 Model Philosophy

Very different model philosophies have been applied to the different projects, all faced problems. In Delfi-C$^3$ a proto-flight approach was taken for all but the high risk parts of the system[7]. However many of the subsystems needed redesigns and could not be used in the final satellite. It was concluded that the team was too inexperienced for a proto-flight approach [7, 9, 36, 45].

To counter this a double prototype approach was chosen for Delfi-n3Xt. The first prototype would be a functional model, the second prototype would be a complete engineering model. Production of flight subsystems would commence when open issues would not affect that subsystem [10, 37]. This approach proved very sensitive to delays in the project [10, 37].

For DelFFi a novel evolutionary approach was proposed [38]. This model philosophy relied on a baseline model, which would be improved upon. Every improvement would introduce a new baseline model, which could be used as a flight model. This is further detailed in Sections 6.1 and 7.3. This never got a proper start because a baseline model was not fully established and not robust enough to the difference in mission. It can be concluded that the model philosophy has to take into account the resources available, the expertise of the team, the sensitivity to delays and its dependence on heritage.

3.8 Development Phases & Life Cycle Model

The life cycle model describe the consecutive phases of the design. It looks at what is to be done in what order, and implements the model philosophy.

Several life cycle models were attempted in the different projects:

- Delfi-C$^3$ implemented a Vee-diagram approach, with parallel phases and special session for non-parallel design [7, 8].
- For Delfi-n3Xt the focus was on two iterative detailed design phases [10, 37].
- In the DelFFi project a radical new iterative baseline approach was attempted [36, 38].

It is clear that parallel development and maturity of all subsystem is not reached in any of the projects. Methods to address this issue have so far been unsuccessful. Planning remained optimistic, specially the later phases were neglected in the schedule. The planning, instead of the design maturity, drove the design phases at times.

3.9 Change Management

During the development, the design of the satellite can change. This includes small adjustments, such as changing the thermal tapes, but also large changes such as the removal of complete subsystems or a change in the solar configuration. These changes are needed to solve problems, or deal with development issues. Any change can have an impact on the development and system.
Change management seems naturally to occur in meetings between experts[8, 10]. An example of this occurred during DelFFi when it was found that part of the teams were working with different (sub-)system modes. The team members involved organised a meeting to resolve the differences. Minor changes have little impact on the rest of the system, while major changes do. Minor changes can be made without major interference, while major changes need more rigour. The changes should be communicated, either in meetings or directly, and documented well. The systems engineer and project manager can be included in the discussions if needed. The rest of the team should be made aware of the change.

3.10 Interface Management

Interface management is needed to ensure all elements can be combined to form a successful system.

It is important that it is clear what the partners are designing and what their requirements are, to avoid double work. For example in Delfi-n3Xt a number of margins were taken into account by different parties, and thus were counted double [15]. In all projects, the interfaces have been documented in Interface Control Document (ICD)s [9, 15], which was successful. The processes of setting and changes these should not become bloated [8, 15]. Meetings should be held regularly to ensure that all partners are aware of the status of the satellite and payloads [15].

3.11 Integration, Verification and Validation

The integration, Verification and Validation (V&V) are closely related activities which should result in a system that satisfies the needs of the project.

The integration starts with getting all the parts together. All projects have seen delays in the delivery of components and subsystems [16]. There are various reasons for this, such as production delays with partners [16] and administrative delays. These delays caused further delays in the project and gave limited possibility to completely verify and characterise the designs [45, 16]. Enough time should be reserved to complete all subsystems.

It is clear that the V&V should be organised according to a verification plan. Both Delfi-C3 and Delfi-n3Xt encountered delays and a decreasing team size in this phase [16]. At moments of time pressure careful documentation and planning was abandoned, which resulted in a lower quality of the work and errors being made [16, 45]. The time available for testing has been identified as a major contributor to the system reliability [36]. The design work was not necessarily done when verification started [9]. Verification activities can be categorized based on what is being verified as was shown in both Delfi-C3 and Delfi-n3Xt, for example the activities could be divided in mechanical, electrical, functional and environmental testing with their own separate priorities and constraints.
3.12 Risk & Reliability Management

Risk should be identified and managed within a project to ensure that failure will not propagate.

The reliability approach taken in the projects was mostly implemented towards technical risks. Technical risks often got priority over programmatic risks. Single Point Failure (SPF) was to be avoided in all projects [7, 12, 38]. For Delfi-n3Xt and DelFFi the same reliability approach was taken. It was noted that commercial grade components failure rates were a minor issue compared to overall system reliability and reliability issues due to planning and resource issues [36].

3.13 Team Structure & Characteristics

Students form the core part of the team, supervised by staff [34]. The project is managed by a staff member [34]. A generic view of the team structure in the Delfi projects is shown in Figure 3.2. Some staff members are part of the project team, while others are not. The project manager is a staff member, who manages and supervises the other team members, which are mostly students. External institutes both advise and cooperate with the project team. The depth of the cooperation depends on their involvement in the project.

Aerospace Engineering (AE) students are best represented, and there is a shortage of team members from EEMCS [11]. The team membership varies during the project, with less people involved near the end of the projects[16]. Alumni and staff from external partners have been hired in the later stages of the project to finish the project either because their expertise was needed or because they were already familiar with the project [11, 16].

Student have little experience, staff is needed to supervise the students [11, 16, 45]. The project hierarchy in the team is essentially flat [45], though staff members can have the upper hand in decisions. Students are motivated to a degree depending on team spirit and the reason they are involved in the project [11, 16].
3.14 Meetings

All projects have used regularly meetings to keep team members up to date with the progress of the project [8]. General team meetings are mostly informative [8, 16], but sometimes lack interaction [46, 47]. Decisions are often made in dedicated meetings between subsystem engineers. Besides those there review meetings are organised to inspect the complete system. Meetings between experts or on specific aspects of the system occur at the initiative of the members when the need arises.

3.15 Documentation

Overall the Delfi projects have taken a document based systems engineering approach, thus most design information was stored in (digital) documents. Documentation quality was identified as the most important factor in information transfer [36]. This section will look at the documentation structure and structure of the documents.

In Delfi-C$^3$ an extensive document control structure was envisioned [7, 8], however this resulted in the document control rarely being used [8, 34, 9]. The documentation was kept at different location, which made it difficult for new members to find what they needed [7, 9, 45]. It also led to duplicate documentation [34]. Team members within Delfi-C$^3$ started new documentation for their own work, leaving previous documents not updated.

To introduce new members a special set of documents was made in Delfi-C$^3$ [7], but soon grew with budget and engineering documents [8]. Ultimately they were abandoned since they needed continues maintenance$^2$.

In the Delfi-n3Xt project these issues were addressed, documentation was kept at a single documentation disk with a rigid structure and documentation control sheets were simplified [34, 48]. Documents were also transferred to successors [34]. This had success. [9, 16].

The DelFFi project attempted to use the same approach [49], however due to time pressure, the document manager did not find time to update the documentation. This lead to team members using only preliminary documentation or abandoning the documentation structure completely.

Time pressure forms a danger to documentation, which is often the first activity to be abandoned [8, 16, 45]. However, experience in all projects shows that this has a negative impact on the quality of the design [8, 45]. It can be seen that without documentation mistakes are made [8, 45], new members will need to do more work to familiarize themselves with the project [47, 50], work cannot be reviewed, and designs cannot be replicated. It is shown that team members need a proper introduction into the documentation and enough time needs to be reserved to produce documentation [16].

The experiences within the DelFFi team show that clear documentation does not automatically arise when a team cooperates together. The structure and documentation quality needs to be controlled.

$^2$From conversations with Jasper Bouwmeester
3.16 Systems Engineering in Practice

This section collects the remaining observations made on the projects. Issues that could not be captured within the previous sections will be described, to get a complete view of the projects.

In all projects the transfer of a task did not go smoothly. In half of the cases, there was no overlap between a team member and a successor [36]. As a result, the knowledge transfer takes up a significant amount of the students' time [16].

All project experience difficulties combining theses with the project [11, 46, 50]. Tasks performed for the projects were not always of the required rigour and quality for a thesis. As mentioned in Section 3.13, this partly explains a lack of electronics and programming students in the project. It was recommended to have a project related supervisor and a thesis supervisor who is not involved in the project [50]; this never seems to have been implemented. It is suspected that having two active thesis supervisors is not realistic within the University context. A proper approach for this has to be found.

Planning is another aspect often mentioned as an issue. Both for Delfi-n3Xt and DelFFi, the planning was considered unrealistic [15]. This had a negative impact on the design work. New systems often had delays, adding delays to an already tight schedule [45]. In DelFFi, still in its early phases, this led to the planning being mostly discarded by team members. In all projects trying to meet deadlines resulted in incomplete or missing documentation (see also Section 3.15), resulting in more delays later on in the project and design errors [8, 45]. Setting short terms tasks seem to have a positive impact on this issue.

Both in Delfi-C3 and DelFFi it was observed that communication towards the student team members can be improved [45, 47]. The design effort did advance slow when students were kept in the dark on decisions, and afterwards the students had to scramble to update their designs. This relates to the question whether the project is a student or university project. For a student project, the students need to be involved in all aspects of the design, for a university project the students require clear and unambiguous assignments.

3.17 Conclusions

A university satellite project requires the involvement of several fields of expertise, the needs of the people and organisations that provide these will need to be fulfilled. Similar, to involve more staff members and institutes in the design, the mission will need to satisfy their needs. Currently this seems to have been neglected in favour of education and the prime objectives of developing the satellite hardware. The engineering methodology should also take into account the capabilities and resources team members bring to the project.

In order to divide the design up in less complex part, all projects used configuration items successfully. It is important that all configuration items are related to tangible parts of the design, and to have a clear view of their maturity. Closely related to the configuration items are the requirements. Proper requirements and their control are very important in giving the team a shared mental model of the system they are working on.

Processes should be kept simple. For a number of processes, such as document control, requirement control and interface control, an ambitious approach was set up. Often team
members considered these approaches as cumbersome, which left the projects in a worse state. Simplifying these processes to their core was the solution. Another issue was that processes were under the responsibility of a single person, which let to a bottle neck if this person could not find enough time to perform these processes. This learns that it is important not to let important processes depend on a single team member.

The model philosophy and development life cycle are very important in controlling the projects, yet so far none of the approaches has been successful. It is important that an approach can handle delays and design flaws. The planning of the project should implement the life cycle model realistically, else it is destined to run into problems. All projects have seen delays, or even relied on them, yet have not been cancelled due to this issue. This is an indicator that runtime of the project has not been one of the main needs.

Avoiding SPFs during the design has resulted in robust satellites. During V&V it is very important to set up verification plans and carefully document the processes.

Meetings are important to keep the team up to date with the project status. Experts organise meetings themselves to discuss subsystem progress, make decisions and manage change. In contrast regular team meetings could lack interaction, yet for a successful project it is important that the team members are transparent towards each other.

Documents have been the prime method of storing information and are very important in information transfer. The documentation should be stored in a clear structure which need to be controlled, but as for other processes the documentation control should not create a bottle neck under the responsibility of single team member. Documentation should be clear, but not create a burden of work on the team members.

Despite these problems Delfi-C³ and Delfi-n3Xt successfully completed their missions. Hard work from team members, inventive solutions and rigour of the designs ensured that the issues encountered could be overcome. That does not mean that the issues identified here should not be addressed, the output of these projects can go up and the development time down with a trustworthy systems engineering methodology to facilitate the team members in their work.
Chapter 4

Team research

Often the fast changing team or lack of manpower are pointed out as major contributors to delays in a university satellite project. This is often not substantiated with research into the team, nor is it concluded whether the manpower was initially overestimated or whether the work to be done was overestimated. In order to get more insight into this issue a study was conducted. In Section 4.1 previous observations are recorded. In Section 4.2 the methodology of the study is described. In Section 4.5 the conclusions of the research are summarised. In Appendices B a complete report on the team study can be found.

4.1 Early observations

Early on in the development of Delfi-C$^3$ it was recognised that students did not form the same workforce as one would expect in a professional organisation [7]. The students were recognised to have little experience, which was expected to result in more man-hours than normal for a given task. The flat structure of the student team was expected to threaten project control. Discontinuities, and related risk to the consistency, in the project were also expected. All of these issues have been encountered, as can be found in Chapter 3.

One of the problems encountered in a student team is that there is no authority of one student over another. This is particularly a problem for the systems engineering and project management tasks [7]. It is also difficult for staff to put sanctions on a team member, given the environment in which tasks are performed. This does improve the quality of discussion, since team members have to actively convince each other [7].

At the start of the Delfi program is was recommended finding ways to include bachelor students into the project [7]. It was recognised that the curriculum at that time did not allow for this.
4.2 Methodology

The objective of this part of the research was to identify the size and characteristics of the complete team working on the Delfi projects. This should give a realistic view on the available work force and how it varies in time.

In order to do this all available documentation was scanned for names of people involved in the development. If a name occurred together with other data, such as a date, a function or a project, it was noted down. Names often occur as the author and reviewer of a document, as a participant in meeting minutes or as recipient and sender in an email conversation. Some lists with participants were found as well.

The data points were collected in a data sheet and subsequently filtered in order to get an overview of who was involved when, and what their function and positions were. The goal is to get a temporal view of the amount of team members involved in the project at a given time, and what was the distribution of stakeholder groups in time. It should be noted that future dates are in general not taken into account. It is assumed a temporal resolution of a month is sufficient. Since the notation of the names can be messy (only initials or first names) the data was cleaned up during collection at discretion of the researcher with the help of other documentation, such as lists of members annotated with their respective acronym. It should be noted that this approach hinges on the assumption that everyone involved leaves a paper trail.

After the data collection was completed the data set was cleaned up. For the researcher there is a drive to make the data set as clean and consistent as possible, in order to curb this drive every edit done to the data set has been documented in a detailed data log. This log can be found in the data file in Appendix E. The main edits consist of finding typing inconsistencies done by both the researcher and by the original writer of the document. Other common edits are relating a first name to a complete name, making the naming of institutes consistent and removing trailing spaces.

Subsequently the data was analysed for trends and other statistical data and reported upon. The complete result can be found in Appendix B.

4.3 Data Analysis

Before the analysis of the data could be started, a number of filters were applied to the data to make the data ‘cleaner’. A project filter was applied to relate as many of the datapoints to a project as possible. Two filters were applied to fill gaps in the data and remove incidental occurrences of persons (filter 1 and 2). The filters are described in detail in Sections 4.3.1, 4.3.2 and 4.3.3.

The effect that the filters have on the data is shown in Figure 4.2. Filter 1 only adds months, as expected. A total of 252 months is added by filter 1 compared to the raw data. After filter 1 and filter 2 are both applied, the difference is distributed normally with a mean of –0.61 with a corrected sample standard deviation of 1.69. A total of 259 months is removed when compared to the original data.
The two persons who gain the most months due to the filter are persons who have been involved in the projects as manager. The three that lose the most months have been involved as staff member and supervisor.

4.3.1 Project filter

The aim of the project filter is to related data points which are not related to one of the three projects to one of these projects. The project filter looked at all the projects related to a single name. If there where data points for which the project was unknown and the other data points were only related to a single other project, it was assumed that the person had only been part of one project. Subsequently the data points with an empty project entry were filled in with the relevant project. The diagram shown in Figure 4.1 illustrates the decision structure.

4.3.2 Filter 1

Filter 1 aims to fill gaps in the collected data. This filter works on the assumption that if a persons is found to be involved in a month, is not found in a second subsequent month, but is found again in a third subsequent month, that person was likely part of the team in all three months. For example if 'John' was found to be part of the team in March and May, but no evidence was found of him in April, it is still assumed he was part of the team in April. This assumption is made (and this filter is applied) because it seems likely that a person might not leave a paper trail every month. For example if a person does research for 5 weeks for a technical document in a period when there were no documented meeting, this person would not leave a paper trail. The people who have only been associated with the first month (November 2004) or last month (August 2014) are not affected by this filter, since there is no earlier or later month to compare these dates with.

4.3.3 Filter 2

This filter implements the assumption that someone who was only present in a single month, without being found in the previous and next month was only involved in the team incidentally. For example, if ‘Jane’ is found in April, but not in March and May, it is assumed she was not a significant part of the team. Combined with filter 1 this means that occurrences of a person that have a gap of two months or more in between occurrences are filtered out.
This filter will filter out all people who have only a single date associated with them. The people who have only been associated with the first month (November 2004) or last month (August 2014) are not filtered out, since there is no earlier or later month to compare them with. An important group that is filtered out are the people associated with launch and early operations. These people are very active in the period of launch, but not in the months before and after. It is important to realise that after filter 2 is applied, the operations team is omitted. This happens since the operations team is mostly active in the month of the launch and does not leave much subsequent documents.

### 4.4 Results

As noted the complete result of the team research can be found in Appendix B, this section summarizes the main conclusions.

A total of 11,398 data points were collected from 646 sources from the project disk, the mail server and a number of other sources. The data contained 428 unique names, of which 33 had no date associated with them. The median of data point per unique name is five, a select few have hundreds of data points associated with them. The data points are not evenly distributed over time.

The team size found during the different months is shown in 4.3, including launch months and month in which the satellites were delivered. For Delfi-C\(^3\) the delivery was in the same month as the launch, so this is not annotated. Comparing the middle and bottom graph it can be seen how the operation team is partly filtered out after filter 2 is applied, specially for Delfi-C\(^3\). All graphs show less people involved in the months prior to the launch and delivery. Delfi-C\(^3\) shows a low team size from April 2007 until launch. Delfi-n3Xt shows little activity...
4.4 Results

Figure 4.3: Approximate People related to the separate Delfi projects over time, color coded for every project. The project filter was applied to all three graphs. Note that someone who was found in multiple projects in a single month is imaged for both projects in this graph.

from July 2009 to September 2010. Also initially a project will have a small team size. Apart from these there does not seem to be a clear trend in the team size over the project life cycle. The team size seems rather characterised by short periods of (more or less) constant team size interchanged sudden changes in the team size.

The statistics of the team size per month based on the data obtained with filter one and two applied is shown in Figure 4.4. The median team size throughout the programme is estimated to have been 19, higher than the median of the separate projects, due to overlap. For Delfi-C3 the median estimated team size is 12.5. For Delfi-n3Xt the estimated team size is 16.5, including weeks between delivery and launch, or 18, without these months. For DelFFi the team size is estimated to be 10.

In Figure 4.5 it is shown how many months were spent on the project by the persons involved. Looking at the filtered boxplot of the complete program, half of the persons involved spent between 4 and 13 months within the project. The upper quartile spreads towards 26 months. The spread is largest for Delfi-n3Xt. The small spread and lower median of DelFFi can be explained by the fact that the project only recently started and has not seen many people graduating within that project. The median for the complete programme, Delfi-C3, and Delfi-n3Xt is 7 months after filtering.
Figure 4.4: Boxplot of the estimated team size during the program and the different projects.

Figure 4.5: Statistics of the amount of months persons were involved in the project. Outliers who were involved for over 30 months have been excluded from this graph for clarity. Note that the figure compares the statistics found both before (boxplots 1 to 5) and after the filters (boxplots 6 to 9). From boxplots 2 to 9, persons who were not found to be involved in the project have been excluded.
4.5 Conclusions

Throughout the project the median team size has been 19, separated over the projects. In the months leading up to a launch or delivery, the team size seems smaller. In the initial stages of a project the team size is also smaller. There are indicators that the team size varies periodically during the year, with a minimum in July, and a maximum in January. Except for these observation, there is no clear trend visible in estimated team size. Team size can change suddenly. Half of the team members spend between 4 and 13 months on the project, with a median of 7 months.

These conclusions confirm that the team size of a university satellite project change and the project needs to take into account a high and sometimes sudden turnover of team members.
Chapter 5

University Satellite Project Context

The university satellite project context describes the environment, stakeholders and criteria the methodology has to interact with. Part of the context has been set in previous Chapters 2 and 3, this chapter will elaborate and generalise the context. This chapter will also make a selection which context will be taken into account.

5.1 Stakeholders

The stakeholders of the methodology are the persons, institutes and other entities that have a stake or are affected by the methodology, and thus in extend by the system [51]. For specific projects the stakeholders can be represented by referencing to the actual person or institute, but in order to make a more robust methodology design, and to represent the core values of each stakeholder, the stakeholders are said to own stakeholder roles which represent their needs and interests. The stakeholder roles will be elaborated in Section 5.1.1.

5.1.1 Stakeholder Roles

In order to identify the needs of the entities involved in a project, stakeholders roles where identified. Stakeholders roles are held by stakeholders. A stakeholder can have multiple roles, for example a student can have the 'student' role and the 'engineer' role. Counter intuitively, if a student volunteers as an operator in the project, this student will have the roles 'operator' and 'volunteer', but not 'student', since this person is not involved in the project as part of an educational program.

The top level breakdown of the stakeholder roles is given in Figure 5.1. This division of stakeholder roles is derived from the stakeholder roles from Holt and Perry [52]. Note that the Supplier stakeholder role Holt and Perry is equivalent to the Design stakeholder role in this division. The Supplier stakeholder role has been added to account for external stakeholder that produces part of the system, and thus form part of the context of the project.
The secondary stakeholders roles are added to reflect the prime interest of the stakeholders. Ignoring these roles would ignore the motivation and interest of the stakeholders and thus misinterpret the role a stakeholder is willing to play in the project.

In Sections 5.1.2 to 5.1.6 the stakeholder roles will be defined and further broken down. In Section 5.1.7 the key stakeholder roles will be discussed.

5.1.2 Design Stakeholder Role

The 'Design' Stakeholder Role refers to all the roles that are associated with providing a service or product that relates to the development of the system.

Stakeholders with the 'Design' stakeholder role are directly involved in the design of the system. They form the core development team of the satellite. The 'Design' stakeholder role can be broken down into two other roles, as is shown in Figure 5.2a. One role is identified for the technical work on the system, while another is identified for the management of the project. The definitions of these roles are:

Engineer The 'Engineer' Design Stakeholder Role refers to all technical roles.
Manager The 'Manager' Design Stakeholder Role refers to all (design) roles that are management related.

5.1.3 Supplier Stakeholder Role

The 'Supplier' Stakeholder Role refers to all the roles that deliver services or products to the system, and can be negotiated with.

The stakeholders with the 'Supplier' role interact with the project by delivering parts or services to the system. They are not part of the core team but can be very involved in the project. This role can be broken down in four other roles, as is shown in Figure 5.2b.
Stock Supplier. The ‘Stock Supplier’ Supplier Stakeholder Role refers to all roles that supply parts to the system, these parts come from the stock of the supplier or are built as a trivial product to the specifications of the customer. Examples of stakeholders with these roles are PCB manufacturers, suppliers of bulk materials like thermal tape or suppliers of components like screws or electrical components. The main need of this stakeholder is to be reimbursed for the delivered components.

Launch Provider. The ‘Launch Provider’ Supplier Stakeholder Role refers to all roles that facilitate the launch of the system. This role is held both by the launch service provider, as space transportation companies. These stakeholders put constrains on the system, as they have requirements for the system to be launched. In some projects the final orbit can be negotiated, in others the orbit will be dependent on the main payload launch by the launch provider.

System Supplier. The ‘System Supplier’ Supplier Stakeholder Role refers to all roles that provide complete customised sub-systems to the system. The supplied items are customised for the system. This supplier role is characterised by a lot of contact between this supplier and the design team. An example of a stakeholder with this role was SystematIC BV in the Delfi-C3 project, this company delivered an electrical power subsystem specifically designed for the Delfi-C3 satellite. Cooperating institutes who make a scientific or technological payload also have this stakeholder role.

Test Facility. A ‘Test Facility’ Supplier Stakeholder Role refers to all roles that provide facilities to research the system or parts thereof. A university might not have all test equipment in-house, a solution is to find an institute that can provide these. These test facilities will put requirements on the items to be tested, and thus possibly the final system. An example of a
stakeholder with this role is Thales, which provided a vibration facility to test the Delfi-n3Xt satellite.

5.1.4 External Stakeholder Role

The 'External' Stakeholder Role refers to all the roles that have an interest in the system or development of the system, but that cannot be argued with.

The external stakeholder roles are fulfilled by stakeholders that have influence on the system or development thereof, however they have no stake in the project itself. These stakeholders bring constraints to the project. A breakdown of this role is shown in Figure 5.3.

**Academic Publisher**  The 'Academic Publisher' external stakeholder role refers to all roles that publish peer-reviewed documentation. It is named here because the research done during the design of a satellite should be of sufficient quality to warrant publication in peer-reviewed journals.

**Educational Institute**  The 'Educational Institute' External Role represents all roles that put educational constraints and demands on the development of the system. This could be a University, a technical school or an governmental institute that guarantees the quality of education.

**Legal**  The 'Legal' External Stakeholder Role refers to all roles that put legal constrains on the system or the development of the system. An example of this could be rules for workplace safety or rules on the export of information and technologies.
5.1 Stakeholders

Management  The ‘Management’ External Stakeholder Role refers to all roles that influence the system and the development of the system on management issues. For example, within a university satellite project this role can be held by the board of the university.

Media  The ‘Media’ External Stakeholder Role refers to all roles that transfer information about the system that cannot be argued with. National news outlets might report on the launch of the satellite for example.

Standard  The ‘Standard’ External Stakeholder Role refers to all standards that are applicable to the system and the development of the system. The standards need to be applied during the system design and in the system. Many CubeSats apply the Inter-Integrated Circuit (I²C) standard to communicate with the different systems in the satellite.

Support  The ‘Support’ Stakeholder Role refers to all roles which deliver advise or information during the development of the system. For example in the Delfi-C³ project radio amateurs gave advise on the implementation of the communication platform of the satellite.

5.1.5 Secondary Stakeholder Role

The ‘Secondary’ Stakeholder Role refers to all the roles that have an influence on the development process of the system.

The ‘Secondary’ stakeholder roles are among the most important to take into account for the design of a methodology. These roles are held by the the core team members and describe what these stakeholders want to get out of the project for personal reasons. In a university environment this is often either education or research. A breakdown of the ‘Secondary’ stakeholder role is given in Figure 5.4.

Researcher  The ‘Researcher’ Secondary Stakeholder Role represents all roles that use the system development or parts thereof as part of their research. An example of this is the in-house development of a micro propulsion system for testing on the DelFFi satellites. The researcher needs to collect data on the systems and make sure that the research project can be finalised with success.

Staff  The ‘Staff’ Secondary Stakeholder Roles refers to all the roles that are hired to participate in the development of the system. These stakeholders bring relatively few constraints to the project.

Student  The ‘Student’ Secondary Stakeholder Role represents all roles that participate in the development as part of their education. As a part of their education the student will need to perform the task set within the educational program of the university or educational institute. This limits the task these stakeholders can do and the time they can allocate for the project.
Supervisor  The ‘Supervisor’ Secondary Stakeholder Role represents all roles that guide and advise participants in the development of the system. The supervisor checks the work of students and ideally advises the students on their educational progress.

Volunteer  The ‘Volunteer’ Secondary Stakeholder Role represents all roles that participate in the development of the system on a voluntary basis. The volunteer has no other stake then participating in the project, but has limited time available.

The ‘Student’ role has been defined more specifically. It might be tempting to order the students by their level of education, however, ordering the students by how they take part in the project will give better insight what the students want to get out of the project. In other words the use case of different levels of students is the same, while the use cases of a ‘Course Student’ and an ‘Intern’ are different. The level of the student can better be captured as a property of the student. It should be noted that a role could be made for the minor and Design Synthesis Exercise (DSE) student, but these are very specific for The Netherlands and Delft University of Technology (TU Delft) and are therefore not considered in this diagram.

Course Student  The ‘Course Student’ Student refers to all roles that participate in the development of the system in order to fulfill the requirements to pass a course. This was for example the case in the AAUSAT-II of Aalborg University (AAU), the student participated in the project through assignments [9].

Intern  The ‘Intern’ Student refers to all roles that participate in the development of the system as part of an internship. In the Delfi programme students from applied universities perform work for the project as an internship.
5.1 Stakeholders

Thesis Student  The ‘Thesis Student’ Student refers to all roles that participate in the development of the system as part of the work to complete a degree. These students need to perform research to complete their education. This puts constraints on the type of work these students can do.

5.1.6 Customer Stakeholder Role

The ‘Customer’ Stakeholder Role refers to all the roles that receive the Service or Product associated with the System.

Stakeholders with the ‘Customer’ stakeholder role are the final users of the system. The breakdown of the customer roles is shown in Figure 5.5.

Operator  The ‘Operator’ Customer Stakeholder Role refers to all roles that operate, control and handle the system. These are the people that execute the mission and manually control the satellite.

Sponsor  The ‘Sponsor’ Customer Stakeholder Role refers to all roles who invest resources into the development of the system and the development process. This investment can be made with monetary resources, an example of this is the university which invests money in the project. The investment can also be made in terms of manpower or discounts on products.
**User**  The 'User' Customer Stakeholder Role refers to all roles that represent the end-users of the system. This is a wide role and thus it is further broken down into the 'Public Relations' role, the 'Scientist' role and the 'Service User' role, which are described in the following paragraphs.

**Public Relations**  The 'Public Relations' User role refers to all roles that use the system for public relation purposes. An example of this is a company using the satellite as a platform to show off their technology. Another example is the university using the satellite to attract new students.

**Scientist**  The 'Scientist' User Role refers to all roles that use data generated by the system. This role is related to the experiments performed on the satellite or other data gathered by the satellite. Delfi-C\(^3\) carried, for example, a thin film solar cell experiment for Dutch Space [36], which means Dutch Space had the stakeholder role of 'Scientist'.

**Service User**  The 'Service User' User role refers to all roles that use functional services provided by the system. Radio amateurs could use Delfi-C\(^3\) as a communication relay, thus they used a service provided by the system.

### 5.1.7 Key Stakeholder Roles

Not all stakeholder roles are of equal importance for the methodology. This section will list the key stakeholders.

The core team that will implement the methodology are the stakeholders with the 'Design' stakeholder role. This makes these key stakeholder roles.

University class satellites are mainly an educational, as was described in Section 2.1. The 'Student' stakeholder roles represent the educational needs of the project, and thus these are key stakeholder roles. Since these projects are also performed University context, staff has likely research objective, thus the 'Researcher' stakeholder role is also a key stakeholder role.

### 5.2 Life Cycles

Life cycles come in many forms and shapes. People are often familiar with the life cycle projects and systems adhere to, however there are also life cycles related to the stakeholders involved, the program, subsystems and components [52].

A life cycle can be described in terms of phases and gates. A phase is related to particular activities, and gates assess the phases. A gate is often a decision moment, it can be decided to continue with the current phase, to continue with the next phase, or discontinue the life cycle or project.

The life cycles discussed in this section will be those of the student stakeholder roles and the generic systems life cycles, as these are the life cycles related to the key stakeholder roles. The life cycle of the students have an important influence on the project life cycle, since the
5.2 Life Cycles

students form the basis of the team and their research and work form the basis of the project. These will be discussed in Section 5.2.1. The system life cycles are considered in Section 5.2.2 in order to understand the general aspects of systems development.

5.2.1 Student Life Cycles

In this section several life cycles related to students will be discussed. In order to illustrate the context better the life cycles encountered by TU Delft students have been taken as default here. Other universities and educational institutes might well have different life cycles, however it is expected that these do not differentiate significantly.

Thesis Student Life Cycle

The thesis or graduation should be the crowning work on the education of the student. A significant portion of the students in the Delft projects did their work in the context of their thesis. The intended life cycle of a master student at Space Systems Engineering (SSE) of TU Delft is shown in Figure 5.6. The phases which are important for the thesis are the literature study and the thesis itself.

In the literature study the student performs a study on the subject of the thesis. The student researches the basic concepts and latest research in the field of the graduation. Within the program of the SSE of the TU Delft, this is an explicit part of the curriculum separate from the thesis. In other educational institutes this can also be found as part of the thesis itself. This study is intended to be done in the first year, in practice it is often done right before the start of the thesis, as is shown in Figure 5.7.

There are a number of other differences between the intended and real life cycles of graduate students. The planning phase often starts already within the literature study and the writing phase often starts before the greenlight review. Note also that both figures are simplifications.

---

1The thesis life cycle is based on information from the course 'Research Methodologies'
students still follow courses during the second MSc year, and documentation activities are also done in the other phases of the thesis.

**Internship Life Cycle**

The life cycle of an intern is shown in Figure 5.8. This life cycle is based on the life cycle of the internship at the TU Delft. The internship search phase is the period in which the student is looking for a position. The project team can influence this search by actively advertising and searching for intern. This phase is ended by an interview or other agreement that the student can perform the internship on the project. The student often can not start immediately, so there is a preparation phase in which the student does not work on the project, but is expected to join. In the internship itself the student works on the project according to an established work package. This work package is established by the project and the educational institute of the student. The intern is evaluated at points in the internship. After the final evaluation the student does not work on the project, though often the student will need to prepare an internship report for the educational institute of the student.
5.2 Life Cycles

Table 5.1: Seven life cycle phases as defined by ECSS-M-ST-10C [53] and the NASA System Engineering Handbook [42]

<table>
<thead>
<tr>
<th>Phase 0</th>
<th>mission analysis and needs identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase A</td>
<td>feasibility</td>
</tr>
<tr>
<td>Phase B</td>
<td>preliminary definition</td>
</tr>
<tr>
<td>Phase C</td>
<td>detailed definition</td>
</tr>
<tr>
<td>Phase D</td>
<td>qualification and production</td>
</tr>
<tr>
<td>Phase E</td>
<td>utilization</td>
</tr>
<tr>
<td>Phase F</td>
<td>disposal</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ECSS-M-ST-10C</th>
<th>NASA SE Handbook</th>
</tr>
</thead>
<tbody>
<tr>
<td>concept studies</td>
<td>concept and technology development</td>
</tr>
<tr>
<td>preliminary design and technology completion</td>
<td>final design and fabrication</td>
</tr>
<tr>
<td>system assembly, integration and test, launch</td>
<td>operations and sustainment</td>
</tr>
<tr>
<td>closeout</td>
<td></td>
</tr>
</tbody>
</table>

Course Life Cycle

For completeness this section shows the life cycle of a generic course. The life cycle is shown in Figure 5.9. A course can have more gates in the form of deliverables. The final gate has been named examination, this can be a report, a written exam or another form of examination.

5.2.2 System Lifecycles

The systems life cycle describes the phases which a system will go through. A number of the life cycles are compared in Figure 5.10 Both the European Cooperation for Space Standardization (ECSS) guideline ECSS-M-ST-10C [53] and the NASA Systems Engineering Handbook [42] divide the development life cycle up into seven phases, as defined in Table 5.1.

Development phases can also be more generally defined. Hartman [54] identifies 3 phases of a mission life cycle: preliminary concept development, mission development, operations & data exploitation phase. He notes that the workforce of these phases consist of different teams, even though there might be some overlap. According to Hartman, the progress within the phases is dependent on 2 main activities: figuring out what has been done, and figuring out what needs to be done next. Chhaniyara et al. [55] define three general phases of the system engineering process, being the concept development phase, the engineering development phase and the post development phase. After these the system will be operational.

It might seem at this point that the design cycle is a linear process where one task follows another, however in reality it is an intrinsic iterative process [19, 42, 51].
**Figure 5.10:** Comparison of life cycles, extended from [19] with information from [53] and [42]
For the purpose of this research five different phases are identified. From figure 5.10 it can be seen that most life cycles follow five similar phases for development:

- **Early development phase**
  In the early development phase stakeholder needs are researched, concept are generated, trades are performed, budgets are allocated and requirements are settled upon. The aim of this phase is to develop a high level feasible design. Early development is characterised with small development teams and rapid changes in design. This phase is equivalent to phase 0, A and B in the ECSS standard and the concept stage in the INCOSE handbook [19].

- **Detailed design phase**
  In this phase the detailed design is performed. The aim of this phase is to complete the design in all its details and make it ready for production. Detailed design is characterised with expert teams working in parallel on their own subsystem. This phase is equivalent to phase C in the ECSS standard and the development stage in the INCOSE handbook [19].

- **Integration and Test phase**
  In this phase components are produced, and integrated with each other. The complete system is tested as well. Deployment activities can be seen as part of this phase. This phase is characterised with work on hardware where changes to the design are unwanted. However in this phase many problems overlooked during design are discovered as well. This phase is equivalent to phase D in the ECSS standard and the production stage in the INCOSE handbook [19].

- **Operations phase**
  The operations phase occurs when the system is in use. It is characterised with a small operations team that needs to now the ins and out of a system. No design changes occur, however failures will. This phase is equivalent to phase E in the ECSS standard and the utilization and support stages in the INCOSE handbook [19].

- **Close out phase**
  The mission is ended and the results of the development and mission are documented. For satellite systems this means the system is turned off or de-orbited. The mission results can be analysed and lessons learned should be documented. This phase is equivalent to phase F in the ECSS standard and the retirement stage in the INCOSE handbook [19].

**5.3 Information, Knowledge and Communication**

Information and knowledge are essential within any projects. Universities aim to create new knowledge and companies often develop strategies to retain it. There is a high turnover of team members in a student project, which makes it important to understand how information is handled. The view as given in Figure 5.11 shows the information, documentation and communication context and how these are handled by stakeholders. The definitions of the elements of this view is given next.
Communication 'Communication' refers to all acts where information is transferred between two or more stakeholders. This can be, amongst other, conversations, emails and presentations. A conversation does not generate documentation, but emails and presentations do, since emails are stored in an email inbox and presentations can be documented by the presentation slides.

Documentation 'Documentation' refers to all instances that store information. This can be reports, emails, and interface documents but also models, tables, technical drawings and other images. Documentation is generated by the stakeholders, and can be generated as a by-product of communications.

Information 'Information' refers to data, statements or a collection of statements. Information is contained within the knowledge of the stakeholders and the documentation. It can be seen as that what is described or remembered.

Knowledge 'Knowledge' refers to the understanding of things. The understanding can be implicit or explicit, theoretical or practical. Things refer to entities like experiences, facts, systems, methods and skills. Note that the understanding of knowledge is out of the scope of this study, but it is important to understand that part of the information and skills are private to the stakeholders.

5.4 Use Cases

In this section the use cases are discussed. The use cases represent the way in which the stakeholder roles use the methodology, and which functionality the methodology should fulfil.
5.4 Use Cases

5.4.1 Engineering Use Cases

The use cases related to the 'Engineer' stakeholder role have been divided up over several diagrams for simplicity.

In Figure 5.12 a diagram with the use cases of the engineer in relation to general design is shown. One of the most important use cases in this diagram is 'Implement methodology'. The engineers will be the prime users of the methodology and will bring it in practice, and thus it is important to make sure they are able and convinced to do this. The engineer also uses tools and manages technical parameters, such as mass and power. The engineer need to comply the design with interfaces and standards, such as the CubeSat standard and test equipment manuals.

The diagram in Figure 5.13 shows the engineer use cases in relation to the information. The engineer will need to use information, both internal to the project and external. An internal progress report is an example of the former, while a peer reviewed paper or a fact sheet of a manufacturer are examples of the latter. The external information comes from stakeholders with the 'Customer', 'External' or 'Supplier' roles.

The engineer will document his or her design. This documentation will need to be stored and controlled. This is mainly a task of the 'Management' role.

Figure 5.14 shows the use cases in relation the the integration and deployment of the system. The system needs to be integrated by the engineers, this use case refers to all cases where hardware elements are combined, where hardware elements are integrated into a higher tier in the hierarchy, to all cases were software elements are combined and to all cases were software is loaded onto hardware. Stock and system suppliers are involved in this case, since they supply the components and systems to be integrated.
In addition to this, the system needs to be deployed. The Operator performs the tasks to make this happen, while the system is launched by the launch provider.

In Figure 5.15 the use case diagram of the engineer use cases in relation to the system needs and requirements. The use cases represent the definition and discovery of stakeholder needs, requirements, and system functions. Note also the use case ‘Control requirements’, which refers to all cases where the requirements are adapted, updated, iterated and broken down into new sub requirements. This use case should not be omitted, because in practice requirements are rarely fully defined at the start of the design. Additionally the use case ‘Standard compliance’ refers to the case were explicitly a standard is implemented within the project or system.

The diagram in Figure 5.16 shows the engineer use cases in relation to the Verification and Validation (V&V) of the system. The verification of the system includes the four standard ways of verification [56]; analysis, inspection, review of design and testing.

5.4.2 Management Use Cases

In this section the use cases directly related to the 'Manager' stakeholder role. The use cases are shown in Figure 5.17. Stakeholders with the 'Manager' role are responsible for the finances within the project, thus within the methodology they will need to make and control the budget. In order to have money to spend the managers will also need to acquire funds from sponsors.

A second task associate with the 'Manager' stakeholder role is control and production of the
Figure 5.14: Use cases related to integration and deployment

Figure 5.15: Use cases related to needs and requirements
5.4.3 Role Related Use Cases

In Figure 5.18a the use cases most directly related to the customer stakeholder are shown. Customers have needs that need to be identified and satisfied within the methodology. The engineer performs the tasks to successfully identify and satisfy these needs.

In Figure 5.18b the use cases directly related to the researcher stakeholder role are shown. The researcher performs technology research as part of the project, as such the methodology needs to incorporate this in the project. The manager will need to the research work inot account within the project planning.

The use cases related to the 'Supplier' stakeholder role are shown in Figure 5.19. The suppliers will need to interface with the system. For system suppliers the control of the interfaces will happen more active than for the other suppliers. The launch provider has a big impact on the orbit of the satellite, since most university satellites use piggy back launches. The manager and management of the institute the system is build are responsible for acquisitions.
5.4 Use Cases

Figure 5.17: Use cases related to the ‘Manager’ stakeholder role

(a) ‘Customer’ use cases

(b) ‘Researcher’ use cases

Figure 5.18: Use cases related to the ‘Customer’ and ‘Researcher’ stakeholder roles
The student want to perform their education as part of the project and methodology, as is shown in Figure 5.20. Their work needs to be integrated in the project as work packages, which the stakeholders with the manager roles should control. The work will need to be reviewed. Naturally this happens during grading, though the team itself might want to review the work to check whether the result aligns with the needs of the project.

### 5.5 Key Parameters

This section will list a number of key parameters which can gauge the success of the methodology. Section 5.5.1 discusses these parameters from the point of view of the student, and Section 5.5.2 from the point of view of the project. The parameters reflect what is referred to as the "iron triangle", the three key variables (time, cost and quality) of project management [57, 58].

#### 5.5.1 Student Key Parameters

To gauge the quality of a thesis one wants to delve into the thesis plan and report to understand the academic content and value its work. However this process would be hard to quantify. Luckily there is no need to do that, since there are already processes in place to review and judge the work of students. The grades a student receives can be seen as an indicator of the quality of the work that the student delivered. Specially for the theses and internships this can be used as a good measurement, since there will also be students who will perform these outside of the project and can be used as a control group. This will be referred to as the student quality parameter. This parameter is linked to the secondary documentation output, as discussed in Section 1.5, since part of the grade is the quality of the
documentation the student makes about his or her work. For courses the grades are less of an indicator, and the quality of the course will need to be judged by the quality control of the educational institute.

Another important factor is whether the internship or thesis is completed in the set time. A delay in completion indicates that there were delays, which could indicate work on the project outside the thesis, work that had to be redone, bad project management, or a bad thesis assignment, with either too much work or too broadly defined. This parameter should also be assessed with a control group, since the time a student takes about the thesis or internship is also dependent on factors external to the project, such as capabilities of the student or personal issues that interfere with the work of the student. This parameter will be referred to as the student time parameter.

Combined these two factors form good indicators of how well the work done by the student fit within their internship or thesis assignment. An issue is that these figures can only be collected when the methodology is implemented in practice, leading to a verification only after implementation. Cost is not implemented, since the investment of the student is either directly linked to time, such as the educational fee, or is outside of the control of the project, such as travel expenses.

### 5.5.2 Project Key Parameters

Similar to students the time to completion is an important parameter to measure the success of a methodology. A good methodology will result in a faster time to completion. In contrast to the students, there is no control group for the project. For a real project the time compared
to the schedule can be compared, though this is also a function of the quality of the planning. This parameter can be assessed in a model environment, where similar modelled projects can be compared with each other. Another approach is to compare university class satellite projects that have similar complexity and resources with each others. This will be referred to as the project time parameter.

The time does partly indicate the success of the project, but a failing system after a fast project is not desirable either. The quality should be measured as well. The quality of the system can be measured by investigating whether the objectives were completed, like done for Delfi-C3 [11]. Another approach to this is to count non conformance to requirements and related faults in the design. In a modelling environment the latter will be more easily implementable, since mission objective definition is part of the systems engineering in a project. This parameter will be referred to as the project quality parameter.

An important factor in any project, also in university satellite projects is cost. To measure the success of a project the final costs can be compared to the estimated costs, but this implicitly also judges the cost estimation. Another issue is that it is not always clear or open what costs are actually made in the project; the staff members within a university project are often hired for more than just the project and income is not necessarily disclosed. Modelling this parameter is hard since the cost are also dependent on external factors and what is available for use by the team. For making a trade off it can be taken into account. This will be referred to as the project cost parameter.

5.5.3 Research Key Parameters

Research forms an important task of universities. While research is not an explicit part of the definition of a university class satellite (see page 8), integrating research into the project will likely have a positive impact on the result of the project and increase the gains of the project for the university. An in-depth discussion about how to measure the output of research is out of the scope of this study, but it could be done by comparing the h-index of research within the project with the h-index of research done external to the project [59]. This would capture the research output of the project in quality over time.

5.5.4 Key Parameter Discussion

Both for students and the project time to completion is an important parameter. For students the quality of the work is indicated by the grade of the students, in comparison to the grade of their peers who work outside of the project. For a system, the quality can be measured by looking at the success of mission requirements and objectives. Cost form an important factor in the system, while this parameter is influenced by the systems engineering methodology, it is also influenced by a number of external factors. The research can be judged by established research parameters, indicated by papers and their citations in comparison to the same metric for research done external to the project.
5.6 Conclusions

It has been established that there are many roles for the stakeholders. The key stakeholder roles are those of the engineer, the manager, the students and the researcher. The engineer and manager execute and interact most with the methodology. A university class satellite is primary concerned with education, thus the needs of the students need to be satisfied in the methodology. The university is a research institute, and thus this should be reflected and integrated in the project.

Related with these roles are life cycles. The life cycles of the student roles have their own timing and gates. These will need to be integrated in the methodology. The system life cycle of a university class satellite can be identified by 5 phases. In the early development phase, this is the phase where the mission and concept are defined. The detailed design phase should result in a complete system design. This design can be made and integrated in the integration and test phase. At the end of this phase the satellite is launched and deployed. After a successful Launch and Early Operations (LEOPS), the operations phase can start, where the system will perform its mission. The final phase is the close-out phase. The satellite will be retired, and the results of the mission documented.

The use cases describe how the stakeholders will interact with the methodology. The engineers control the design, document it and verify the system. The manager makes planning and budget. The researcher want to implement a research programme into the project. Students want to perform their educational assignments within the project.

The performance of the methodology can be measured by looking at the grades of the students and the time they spent on the project, all in comparison with their peers external to the project. From the project perspective, the project can be gauged by the time, cost and quality. The research output of the project can be gauged by established research parameters. An issue is that these parameters can only be measured during the project, and conclusions can only be made after the projects have come to an end.
In this chapter the important trade-offs will be discussed. Every section will discuss the criteria, design options and the trade-off result. The trade-offs are made with the Analytical Hierarchy Process (AHP). The comparison between the criteria and options is documented in Appendix D. The criteria are selected and weighted using information from Chapters 2, 3 and 5. Table 6.8 shows the traceability between the criteria and key parameters found in Section 5.5.

6.1 Trade-off Methodology

The trade-offs were made with AHP[60]. In order to find the weighting of the criteria, they are compared pairwise on a scale of one to nine. This scale is shown in Table 6.1. After the weights are calculated a consistency check is performed by calculating the consistency ratio on the results. They are accepted if the consistency ratio is below 10%. The resulting weights are reflected upon.

Likewise the criteria are compared pairwise. When the data was available or when a reliable estimate could be made this was done with the absolute values. For the majority of the options this was not the case. In these cases a qualitative comparison is made. In this qualitative comparison, first the performance of the option on the criteria are described. These performances are compared pairwise, with the values as given in Table 6.1. The results of these qualitative comparisons are checked for consistency by calculating the consistency ratio. These are accepted if the consistency ratio is below 10%. The results of the trade-off is reflected upon.

The consistency ratio, CR, is calculated with Equation 6.1. In this formula $\lambda_{max}$ is the principal eigenvalue of the comparison matrix and $n$ are the number of options that are compared. The random index $RI$ is dependent on the number of options compared as these are listed in Table.

Master of Science Thesis

N. van der Pas
### Table 6.1: Scoring table for qualitative comparisons in the trade-offs. Values 2, 4, 6 and 8 are intermediate values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The options/criteria are of equal importance or perform equally.</td>
</tr>
<tr>
<td>3</td>
<td>The winning option/criteria is slightly more important or performs slightly better than the losing option/criteria.</td>
</tr>
<tr>
<td>5</td>
<td>The winning option/criteria is strongly more important or performs strongly better than the losing option/criteria.</td>
</tr>
<tr>
<td>7</td>
<td>The winning option/criteria is very strongly more important or performs much better than the losing option/criteria.</td>
</tr>
<tr>
<td>9</td>
<td>The winning option/criteria is absolutely more important and performs absolutely better than the losing option/criteria.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>n</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>RI</td>
<td>0</td>
<td>0</td>
<td>0.52</td>
<td>0.89</td>
<td>1.11</td>
<td>1.25</td>
<td>1.35</td>
<td>1.40</td>
<td>1.45</td>
</tr>
</tbody>
</table>

### Table 6.2: Random index [60]

\[ CR = \frac{1}{RI} \cdot \frac{\lambda_{\text{max}} - n}{n - 1} \]  

#### 6.2 Systems Engineering Philosophy

The first step in the design is to determine on what philosophy the methodology will be established. The philosophy describes the main characteristics of the methodology. Three options and a number of variations will be considered and are described in 6.2.2. These philosophies are mainly separated based on the information management approach they take. The main identifier of this trade off is SEP, for Systems Engineering Philosophy.

#### 6.2.1 Criteria

The different Systems Engineering (SE) philosophies will be judged on the following criteria:

- SEP-C-1 Adoption period or Learning curve
- SEP-C-2 Accessibility of information
- SEP-C-3 Speed of information updates
- SEP-C-4 Error sensitivity
- SEP-C-5 Workload
- SEP-C-6 Sensitivity to workforce changes
- SEP-C-7 Cost
- SEP-C-8 Secondary stakeholder role needs
The criteria are defined such that they are independent from each other when indirect effect are ignored during the trade off. For example, more errors will increase the workload, however for the trade off this effect is ignored. The criteria can be traced back to the key parameters, as shown in Table 6.8. No criteria link back to the research quality parameter, since the effect of the systems engineering philosophy on that parameter is uncertain and likely represented in other criteria, such as workload and error sensitivity.

**SEP-C-1 Adoption period or Learning curve**  This criteria judges how fast the methodology can be adapted by new team members. It is assumed a new team member has the knowledge of a master student with some information on standard engineering practices. The learning curve is important because team members in a university project often only have a limited period to work on the project, time that is spent learning the methodology cannot be spend on project work. This criteria can be quantified by collecting data on how much time new team members spent on training related activities. In the current trade-off this criteria is judged qualitatively, since the data is not available to make a quantitative decision.

**SEP-C-2 Accessibility of information**  This criteria judges how easy it is to obtain information within the methodology. This is a complex interaction between the accessibility of the documentation and the links between these documents. Likely no quantitative judgement can be made, thus this criteria is judged qualitatively. When judging this criteria it is assumed that team members have full knowledge of the methodology. Accessible information makes the design effort more efficient and makes it easier to review project information and the status of the design.

**SEP-C-3 Speed of information updates**  This criteria judges how fast information becomes available to the team to work with. This is important because if the information because available slower, the team will continue to work with outdated information. This criteria can be quantified by measuring the time it takes between the moment a document is finished and the moment it is made officially available. Since this data is not available, an estimation in number of weeks has been made for the different options.

**SEP-C-4 Error sensitivity**  This criteria judges how well the methodology handles the occurrence of errors. This judges for example how well inconsistencies can be detected and how well design errors are prevented. The only error that will not be judged by this criteria is errors due to out of date information, since this is covered by criteria SEP-C-3. This criteria can be quantified the number of errors found in documentation during a review. The errors have to be predifined and categorised. One could think of calculation errors, citation errors, and non-conformance errors. The latter was for example performed by Lebbink [15]. In the current trade-off this criteria is judged qualitatively, since the data is not available to make a quantitative decision.

**SEP-C-5 Workload**  This criteria rates the added workload of the methodology on team members. This excludes the initial workload due to the adoption period since this is covered in criteria SEP-C-1. Team members prefer to spent most time designing the system, and not
on, for example, documentation. A methodology with a light workload will likely find an easy adoption. This criteria can be quantified by having team members document how much hours they spent on methodology specific activities. However, since that data is not currently available, the criteria is judged quantitatively.

**SEP-C-6 Sensitivity to workforce changes** Teams of university satellite project undergo constant change, often there is also too little manpower available at a given moment. This criteria judges how well the methodology handles these changes. It will look at how well the hand over is handled and how well the methodology holds up when not all positions within the team are filled. In review this criteria can be measured by comparing projects using the same methodology with differences in the available workforce, this data is however not available. Thus the comparison was done qualitatively.

**SEP-C-7 Cost** This criteria judges the investment needed to implement the methodology. For some of the methodologies advanced tools and specific environments are needed. It is assumed that workplaces and computers with an office software suite are available. This criteria can be quantified by counting the costs needed for the implementation of the methodology. However, since the current decision is made on the general approaches, and not the specific implementation, the costs are hard to estimate in precise numbers. Time constraints also limit the possibilities to make this estimation. Therefore the decision will be made qualitatively.

**SEP-C-8 Secondary stakeholder role needs** This criteria judges how well the methodology aligns with the secondary needs of the main stakeholder roles. For students this is how well the methodology aligns with their own deliverables. For staff this looks how well the methodology can be combined with research work. This criteria can be judged qualitatively by having the team members judge their situation through interviews or questionnaires. Since this can only be done after implementation, for now the performance of this criteria is judged qualitatively.

**Criteria weights**

The criteria were compared pairwise, which resulted in the weights given in Table 6.3. The comparison is summarised in Table D.1.

The criteria that takes into account the other needs of the stakeholders has been given the highest weight. University class satellite projects are done for education, thus proper integration into the education is important. This is both given in the university class satellite definition (Section 2.1) and was observed in Section 2.3 and 3.16 In table 6.8 it can be observed that criteria SEP-C-8 is also the only criteria related to the student parameter, which explains the relative high weight as well. The Adoption period, Workload and costs have been given the smallest weight. These criteria have only a in-direct impact on the success of the methodology. Moreover these criteria are related to the time parameter of the project, which was seen not to have a high priority in Chapter 3. It is also considered that the impact of costs will likely not be very large in comparison with satellite project costs.
6.2 Systems Engineering Philosophy

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-C-1</td>
<td>4.87%</td>
</tr>
<tr>
<td>SEP-C-2</td>
<td>11.89%</td>
</tr>
<tr>
<td>SEP-C-3</td>
<td>11.89%</td>
</tr>
<tr>
<td>SEP-C-4</td>
<td>13.55%</td>
</tr>
<tr>
<td>SEP-C-5</td>
<td>4.87%</td>
</tr>
<tr>
<td>SEP-C-6</td>
<td>13.55%</td>
</tr>
<tr>
<td>SEP-C-7</td>
<td>6.80%</td>
</tr>
<tr>
<td>SEP-C-8</td>
<td>32.59%</td>
</tr>
</tbody>
</table>

Table 6.3: Weights of criteria in systems engineering philosophies trade off

6.2.2 Systems Engineering Philosophy Options

The systems engineering philosophies that will be considered are:

- SEP-O-1 Document based systems engineering, with variants:
  - SEP-O-1.1 Reviewed document based systems engineering
  - SEP-O-1.2 non-reviewed document based systems engineering
  - SEP-O-1.3 Assignment report based systems engineering

- SEP-O-2 Model based systems engineering, with variants:
  - SEP-O-2.1 All members modelling
  - SEP-O-2.2 Single modeller

- SEP-O-3 Concurrent engineering

It can be noted that these design options are not mutually exclusive. For example, a model based systems engineering with a single modeller can still use documents to store part of the information. Specially concurrent engineering is a design option that stands out in this list. Concurrent engineering can be applied in a model based or document based fashion and also identifies the setting in which the engineering is done and life cycle characteristics. However it was decided to include this design option, because concurrent engineering is also characterised by a different communication of information during the design.

SEP-O-1 Document based systems engineering

Document based systems engineering is the classical approach in which the design information is mainly stored in technical reports. Reports are accessible since every one involved can read them, however the amount of reports increases easily. To keep track of these a proper archiving approach and version control need to be implemented. The documents also need to be checked on errors and inconsistencies between them, proper referencing is necessary for this. It should also be noted that the construction and review of reports can be a time consuming practise. For the evaluation it is assumed that a proper clear accessible documentation structure is available.

Master of Science Thesis

N. van der Pas
SEP-O-1.1 Reviewed document based systems engineering
This is the standard variant of document based systems engineering. The reports are first reviewed, before they are officially released. Using this method documents will likely contain little errors, though their release might be delayed due to the reviewing process. This is the approach taken to the Delfi program so far.

SEP-O-1.2 Non-reviewed document based systems engineering
In this variant the documents are not reviewed and are released immediately to the team. It relies on the assumption that the team members produce quality documents and review these properly themselves. The advantage is that documents are available immediately, but there is a higher change of errors and non-standard documents.

SEP-O-1.3 Assignment report based systems engineering
This approach can be specifically applied to university projects. The team consist of mainly student, who produce reports in order to finalise their course, internship or thesis. These documents are automatically reviewed during grading, though errors can remain in the documents since there will only be a resubmission if the student fails the assignment. This also means documents will only be available at the end of their assignment.

SEP-O-2 Model based systems engineering
Model based systems engineering is a new systems engineering approach which takes a system model as the core of the design effort. The use of a system model can reduce the inconsistency within the system, since all information is linked within the model. However team members will need to understand the modelling approach, language and tool [61, 62, 63, 64, 65]. It should be noted that a computer will offer the engineers a number of automated analysis and reporting tools, experience with these showed that these still need more development to be usable. The model does not exclude errors either [64, 62] and thus will need to be reviewed as well, though this effort should be easier than the review of documents.

SEP-O-2.1 All members modelling
In this variant the modelling will be done by all team members. Design solutions and options will be modelled, which will make the information available to the team and immediately show the consistency and need of the design. This does mean that everyone needs to be able to work with the model.

SEP-O-2.2 Single modeller
In this variant there is only a single system modeller. This system modeller has the specific task to model the system, and thus check the design. Team members can subsequently check automatically generated reports or the system model directly. Less effort is needed to only read the model, and a single modeller can more easily make a consistent model. However, if this means a single team member is responsible for the main documentation and systems engineering effort. If this team member delivers bad quality or drops out of the project for any reason, the project will be negatively impacted. Handover to a new person could also cause discontinuities in the modelling.
6.2 Systems Engineering Philosophy

Concurrent engineering is a methodology which applies all activities for a phase in parallel. It is characterised by team members using an integrated software suit and short communication lines. Documentation is primarily done at the end of the study to document the results. To perform studies often a separate facility is build such as the Concurrent Design Facility at European Space Agency (ESA).

6.2.3 Trade-Off Conclusions

The options were compared pairwise for every criteria, which can be viewed in Appendix D.1.2. The results are tabulated in Table 6.4 and also shown in Figure 6.1. In the last column of Table 6.4 it can be observed that all consistency ratios are below 10%, thus the scores can be considered consistent.

It can be seen that the methodology which relies on education report documentation is the winner of the trade off. On most criteria this option scores average or high. This approach has a minimum impact on the normal life cycle of the students, and delivers reviewed documents. A disadvantage of this approach is that updates to the documentation do take a long time, which results in this option receiving the lowest score on criteria SEP-C-3. This is an issue that should be addressed in implementation of this methodology.

The runners up are the other document based methodologies and the model based methodology where all team members model. The latter scores very low on the fulfilment of the secondary stakeholder needs, all documentation work for this should be done besides the methodology and project. Option SEP-O-2.1 also score very low on the adoption period. It should be noted that when model based engineering is part of the curriculum of the students and when better automatic document generation becomes available this trade-off will have to be redone. If these are available it is likely that Model Based Systems Engineering (MBSE) will be the preferred approach.
<table>
<thead>
<tr>
<th></th>
<th>SEP-O-1.1 Reviewed documents</th>
<th>SEP-O-1.2 Non-reviewed</th>
<th>SEP-O-1.3 Assignment report</th>
<th>SEP-O-2.1 MBSE (all)</th>
<th>SEP-O-2.2 MBSE (single)</th>
<th>SEP-O-3 Concurrent</th>
<th>Consistency Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-C-1</td>
<td>21.02%</td>
<td>21.02%</td>
<td>42.79%</td>
<td>2.24%</td>
<td>5.49%</td>
<td>7.43%</td>
<td>5.75%</td>
</tr>
<tr>
<td>SEP-C-2</td>
<td>8.07%</td>
<td>8.07%</td>
<td>30.32%</td>
<td>42.75%</td>
<td>2.73%</td>
<td>5.20%</td>
<td></td>
</tr>
<tr>
<td>SEP-C-3</td>
<td>4.31%</td>
<td>8.62%</td>
<td>0.86%</td>
<td>34.48%</td>
<td>17.24%</td>
<td>34.48%</td>
<td>0.00%</td>
</tr>
<tr>
<td>SEP-C-4</td>
<td>31.77%</td>
<td>3.08%</td>
<td>31.77%</td>
<td>14.20%</td>
<td>10.73%</td>
<td>8.46%</td>
<td>6.95%</td>
</tr>
<tr>
<td>SEP-C-5</td>
<td>2.82%</td>
<td>6.36%</td>
<td>16.37%</td>
<td>29.83%</td>
<td>14.80%</td>
<td>29.83%</td>
<td>5.03%</td>
</tr>
<tr>
<td>SEP-C-6</td>
<td>4.68%</td>
<td>25.24%</td>
<td>25.24%</td>
<td>25.24%</td>
<td>4.68%</td>
<td>14.94%</td>
<td>0.59%</td>
</tr>
<tr>
<td>SEP-C-7</td>
<td>28.24%</td>
<td>28.24%</td>
<td>28.24%</td>
<td>3.70%</td>
<td>5.57%</td>
<td>6.02%</td>
<td>6.36%</td>
</tr>
<tr>
<td>SEP-C-8</td>
<td>21.64%</td>
<td>21.64%</td>
<td>42.30%</td>
<td>5.75%</td>
<td>5.75%</td>
<td>2.91%</td>
<td>4.03%</td>
</tr>
<tr>
<td>Unweighted total</td>
<td>15.32%</td>
<td>15.28%</td>
<td>24.46%</td>
<td>18.22%</td>
<td>13.38%</td>
<td>13.35%</td>
<td></td>
</tr>
<tr>
<td>Weighted SEP-C-1</td>
<td>1.02%</td>
<td>1.02%</td>
<td>2.08%</td>
<td>0.11%</td>
<td>0.27%</td>
<td>0.36%</td>
<td></td>
</tr>
<tr>
<td>Weighted SEP-C-2</td>
<td>0.96%</td>
<td>0.96%</td>
<td>0.96%</td>
<td>3.61%</td>
<td>5.08%</td>
<td>0.32%</td>
<td></td>
</tr>
<tr>
<td>Weighted SEP-C-3</td>
<td>0.51%</td>
<td>1.03%</td>
<td>0.10%</td>
<td>4.10%</td>
<td>2.05%</td>
<td>4.10%</td>
<td></td>
</tr>
<tr>
<td>Weighted SEP-C-4</td>
<td>4.30%</td>
<td>0.42%</td>
<td>4.30%</td>
<td>1.92%</td>
<td>1.45%</td>
<td>1.15%</td>
<td></td>
</tr>
<tr>
<td>Weighted SEP-C-5</td>
<td>0.14%</td>
<td>0.31%</td>
<td>0.80%</td>
<td>1.45%</td>
<td>0.72%</td>
<td>1.45%</td>
<td></td>
</tr>
<tr>
<td>Weighted SEP-C-6</td>
<td>0.63%</td>
<td>3.42%</td>
<td>3.42%</td>
<td>3.42%</td>
<td>0.63%</td>
<td>2.02%</td>
<td></td>
</tr>
<tr>
<td>Weighted SEP-C-7</td>
<td>1.92%</td>
<td>1.92%</td>
<td>1.92%</td>
<td>0.25%</td>
<td>0.38%</td>
<td>0.41%</td>
<td></td>
</tr>
<tr>
<td>Weighted SEP-C-8</td>
<td>7.05%</td>
<td>7.05%</td>
<td>13.79%</td>
<td>1.87%</td>
<td>1.87%</td>
<td>0.95%</td>
<td></td>
</tr>
<tr>
<td>Weighted total</td>
<td>16.54%</td>
<td>16.13%</td>
<td>27.37%</td>
<td>16.74%</td>
<td>12.46%</td>
<td>10.77%</td>
<td></td>
</tr>
</tbody>
</table>

**Table 6.4:** Results for the systems engineering philosophies trade off.
6.3 Model Approach

This trade off will assess which model approach is best for a university satellite project. The conclusion will advise on a main model approach to be used. For specific elements another approach could be chosen due to specific circumstances. For example, Delfi-C\textsuperscript{3} opted for a proto-flight approach, but also made prototypes for high risk items, as can be read in Section 3.7. The main identifier of this trade off is MA, for Model Approach

6.3.1 Criteria

The different model approaches will be judged on the following criteria:

- MA-C-1 Time
- MA-C-2 Cost
- MA-C-3 Reliability
- MA-C-4 Robustness

The criteria can be traced back to the key parameters from Section 5.5, as shown in Table 6.8. For the model approach trade off, no criteria linked to the student parameters have been used, since it is not expected that the model approach has a big influence on the needs for the student stakeholder. There is no criteria that links back to the research quality parameter, since the different model approaches do not seem to have a different effect on the quality of research.

**MA-C-1 Time**  This criteria judges how much time the model approach costs. This criteria could be quantified by measuring the time it takes to complete projects using the different model approaches. However since this data is not available the criteria will be scored qualitatively. The time is judged in number of iterations needed for the model approach, taking into account how long the iterations cost.

**MA-C-2 Cost**  This criteria judges the cost of the model approach in terms of the amount of models that need to be produced. Only the quantity of the models is considered here, since the model approach has mainly an impact on the cost generated by the different amount of models. The cost for a single model can be expressed in terms of hardware cost, software cost and personnel cost. The different models within one approach have different costs, however in order to simplify the comparison, this is ignored. The exact cost differs per mission and data is not available. For this trade off the number of models required are estimated to quantify the costs. It is assumed that cost of spare parts are directly dependent on the number of main models.

**MA-C-3 Reliability**  This criteria rates how well the model approach gives the opportunity to discover and correct errors. For example a prototype can be tested for errors, after which they can be addressed in a subsequent model. This criteria can be quantified by counting the amount of errors or required fixes in the final flight model. A limit on what would be
considered an error or fix will have to be determined for this, however this is not part of this research. This data is not available and collection of this data is also out of the scope of this research, therefore this criteria will be judged qualitatively. For this comparison it is assumed the amount of errors is dependent on the opportunities to find and correct errors.

**MA-C-4 Robustness** This criteria rates how well the model approach can cope with changes in the team and differences in maturity of the different elements of the system. This criteria can be measured by qualitatively judging case studies. Since no comparison has been done at this point, the options are mainly compared based on the amount of deadlines and on how strict these are.

**Criteria Weights**

The criteria were compared pairwise, which resulted in the weights given in Table 6.5. The consistency ratio of the criteria weights is 2%, and thus consistent.

The cost criteria has the highest weight. Costs need to be kept low. This was one of the motivators to choose a proto-flight approach in Delfi-C³, as could be read in Chapter 3. The limited financial resources, see Section 2.4, is also reflected in the high weight. The time has received the lowest weight, according to conclusions of Chapter 3. The runner up is the reliability criteria, since university satellites rely on inexperienced students (see Section 2.3), it is important that there is ample time to discover flaws in the system.

The time to completion is important, specially for technological demonstration missions, as noted in Section 2.2. However delays in many of the university satellite projects, such as Delfi-C³ and Delfi-n3Xt, show that this criteria is not that strict, and the projects could continue, even though the project time increased.

### 6.3.2 Model Approach Options

The model approach options considered in this trade off are:

- MA-O-1 Prototype approach
- MA-O-2 Double prototype approach
- MA-O-3 Proto-flight approach
- MA-O-4 Elstak CubeSat Model Approach
- MA-O-5 Baseline Evolution
6.3 Model Approach

MA-O-1 Prototype approach

The prototype approach is a classical approach were an Engineering Model (EM) is made for the complete system. This prototype will undergo functional and qualification testing. Afterwards a complete new flight model will be created for the Acceptance testing, possibly with a number of refurbishments.

This approach requires that a complete system is created before the flight model can be created, which will be thoroughly tested for mistakes.

MA-O-2 Double prototype approach

The double prototype approach was proposed for Delfi-n3Xt [10]. The first prototype should prove functionality and interfaces, though a number of issues could be remaining. The second prototype will be an EM without any issues remaining. This model will undergo qualification testing, and flight hardware will only be produced when no issues are remaining. In this approach 3 complete models will need to be made.

This approach is time intensive, but will likely have ample opportunities to tackle issues.

MA-O-3 Proto-flight approach

The proto-flight approach was proposed for Delfi-C³ [479x479]. A single proto-flight model will be produced for the system, after testing this model will be refurbished into the flight model. This approach requires a proto-flight model that can successful pass most tests, the refurbishment should not result in a completely new system. Breadboarding and prototyping is only done at subsystem level if necessary. The latter was an issue in the Delfi-C³ project, as can be read in Section 3.7. This model approach is thus relatively fast and cheap, but takes considerable risks.

MA-O-4 Elstak CubeSat model approach

The Elstak CubeSat model was proposed by Elstak [2] after studying the Delfi-C³ project and the AAUSAT-II project. The approach consist of two models, a basic model and a flight model, as shown in Table 6.6. Prototypes are made for subsystem testing. The basic model is comparable to the EM, and is build to verify the functional architecture. Initial environmental tests can be performed on the basic model as well. With the result of these functional tests, the design is updated. Based upon the new design a new flight model is made.

This approach is less extensive than a prototype approach, but more so than the proto-flight approach. In comparison with the prototype approach it will save some cost, in comparison with the proto-flight approach it gives more options to find and solve errors.

MA-O-5 Baseline evolution

The baseline approach was proposed for the DelFFi [36], as noted in Section 3.7. The main model in this approach is the baseline. The baseline is updated during project, as shown in
<table>
<thead>
<tr>
<th>Prototypes</th>
<th>Basic Model</th>
<th>Protoflight Model</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Purpose</strong></td>
<td>Proof of subsystem functions</td>
<td>Achieve an integrated and functional architecture</td>
</tr>
<tr>
<td><strong>Functionality</strong></td>
<td>Subsystem subfunctions</td>
<td>Comparable to EM boards</td>
</tr>
<tr>
<td></td>
<td>Subsystem functionality</td>
<td>EPS architecture functional</td>
</tr>
<tr>
<td></td>
<td>Proof of concept</td>
<td>CHDS architecture functional</td>
</tr>
<tr>
<td><strong>Verification</strong></td>
<td>Verify functions</td>
<td>Qualification testing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Acceptance testing</td>
</tr>
</tbody>
</table>

Table 6.6: The CubeSat Model Approach as proposed by Elstak [9]

Figure 6.2: The baseline evolution approach [36]

Figure 6.2. The system improves by design iterations of subsystems and components. Every improvement would undergo complete verification, making it possible for the baseline to be used as flight model.

This model approach deviates from the other approaches in that a complete flight model is available at any moment. It will thus be harder to compare on the time criteria, since any iteration, in principle, could be used for the mission. The project time is thus only dependent on the specific payloads. On cost it scores also well, since there is no need to have multiple complete models. It performs best when a baseline is available at the start of the project.

6.3.3 Trade-off Conclusions

The design options were compared pairwise for all criteria, details about this can be found in Appendix D.2.2. The scores are summarised in Table 6.7 and is shown in Figure 6.3a and Figure 6.3b. The baseline evolution model approach wins the trade off for both the unweighed and weighed score, it scores good on invested time, reliability and robustness. Due to the approach that a flight ready baseline is always present, there will be ample opportunities to
verify the system. New subsystems need to be verified before integration into the baseline, making every new integration a new testing opportunity, with minimal new hardware. This model approach is not dependent on all packages completing before a model can be completed, if a package is not completed, it will not become part of the baseline.

Table 6.7: Scores of the model approach trade off

<table>
<thead>
<tr>
<th></th>
<th>MA-O-1 Prototype</th>
<th>MA-O-2 Double Prototype</th>
<th>MA-O-3 Proto-flight</th>
<th>MA-O-4 Elstak</th>
<th>MA-O-5 Baseline</th>
<th>Consistency Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unweighed \ MA-C-1</td>
<td>12.11%</td>
<td>3.58%</td>
<td>33.37%</td>
<td>12.11%</td>
<td>38.83%</td>
<td>5.30%</td>
</tr>
<tr>
<td></td>
<td>20.00%</td>
<td>13.33%</td>
<td>26.67%</td>
<td>20.00%</td>
<td>20.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td></td>
<td>10.74%</td>
<td>27.83%</td>
<td>2.51%</td>
<td>16.70%</td>
<td>42.21%</td>
<td>7.08%</td>
</tr>
<tr>
<td></td>
<td>3.54%</td>
<td>7.82%</td>
<td>13.64%</td>
<td>24.34%</td>
<td>50.67%</td>
<td>7.84%</td>
</tr>
<tr>
<td>Total</td>
<td>46.39%</td>
<td>52.56%</td>
<td>76.19%</td>
<td>73.15%</td>
<td>151.71%</td>
<td></td>
</tr>
<tr>
<td>Weighed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.77%</td>
<td>0.23%</td>
<td>2.13%</td>
<td>0.77%</td>
<td>2.48%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7.97%</td>
<td>5.31%</td>
<td>10.63%</td>
<td>7.97%</td>
<td>7.97%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.80%</td>
<td>9.84%</td>
<td>0.89%</td>
<td>5.90%</td>
<td>14.92%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.65%</td>
<td>1.44%</td>
<td>2.51%</td>
<td>4.48%</td>
<td>9.32%</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>13.19%</td>
<td>16.82%</td>
<td>16.16%</td>
<td>19.13%</td>
<td>34.70%</td>
<td></td>
</tr>
</tbody>
</table>

The runner up is the proto-flight approach, though it’s place is mostly due to a good score on the cost criteria, only a single model with some replacements needs to be produced. However this model approach scores very bad on the reliability criteria, as was experienced during the Delfi-C3 project, see Section 3.7 for more information on this. Taking into account criteria weighting, this approach does not score much better than the other approaches.

Table 6.8: Traceability between trade off criteria from this chapter and the key parameters presented in Section 5.5.
6.4 Conclusions

From the trade-offs made in this chapter it can be concluded that currently the preferred systems engineering philosophy is a document based methodology. A methodology based on the reports of students is seen as most desirable. In this approach, it can take a long time for information to be officially documented, there should be processes in place that encourage information sharing between active team members. If the students and other team members have a good knowledge model based systems engineering, the current trade off is no longer valid and a new look at the problem is needed.

The model approach that has been selected is an approach that improves a verified baseline incrementally. This approach gives confidence in the design and can accommodate a varying team. This model approach has a big impact on the life cycle. The establishment of a baseline also requires attention.
This chapter presents the university satellite Systems Engineering (SE) methodology. This methodology is based on the research and trade-offs made in the previous chapters. The chapter starts by covering the most important elements of the methodology; Configuration management in Section 7.1, Work Packages in Section 7.2 and the baseline life cycle in Section 7.3. The methodology will be illustrated by an example given in Section 7.10.

7.1 Configuration Management

Configuration items are used to break the system and design problem in to pieces for easier design. It has been applied in all Delfi projects, as can be read in Section 3.4. The configuration items have been shown to successfully link the requirements to the design. Though there were also issues, in Delfi-C\textsuperscript{3} the configuration item breakdown was not reflected in the hardware breakdown, and a number of configuration items were ill defined. It was seen in both Delfi-C\textsuperscript{3} and DelFFi that team members were linked to the configuration items, as can be seen in Appendix A.4. This does make clear who is responsible for which item, but does not match with the actual work package, since a configuration item consist of multiple work packages.
7.1.1 Configuration Item Definition

Before procedures for the identification and breakdown of configuration items can be established, it is important to define what a configuration item is. Both the ECSS standard [66] and NASA [42] define a configuration item as an aggregation of hardware, software, functionality or a combination thereof that is used for configuration management. Based on these definition and previous work done by Genbrugge [43], the following definition of a configuration item is established:

Configuration item:
A configuration item is an aggregation of hardware, software, functionality or a combination thereof which are worthwhile to control separately and display disciplinary homogeneity.

In contrast to the ECSS and NASA definition, disciplinary homogeneity was added to the definition. This is important for a university satellite, because a configuration item can be linked to a field of study, and thus to a university department, course or thesis topic.

The ECSS standard [41] makes a difference between developed and non-developed configuration items. The developed configuration items are developed fully or partially for the mission, while the non-developed configuration items are not. For university satellites this is an important distinction to make as well, since they often incorporate commercial off-the-shelf (COTS) components and systems (see Section 2.4).

The configuration items form the central element in the systems engineering and project management. They indicate both what has been done and what needs to be done.

The configuration items should be documented in a central document. In previous Delfi projects the configuration items were contained in a simple list, however this does not provide the information that is needed to use the configuration items as central design elements. The configuration items should be documented with a description of all its properties. An overview of the configuration items can be contained in a single list, which is sourced to this main file. In Appendix E, a template can be found of such a list.

7.1.2 Configuration Item Properties and Links

The implementation of configuration items within the methodology is shown in Figure 7.1. As can be read in Section 3.4, configuration items within the Delfi projects were already linked with the requirements. This is still the case. As follows from the definition, the configuration item is an aggregate of software and hardware items.

The configuration items are a convenient way to control the work in the project. This was attempted previously in DelFFi, though this did not work out. The work that will need to be done to complete the configuration item should be described as a development life cycle. This life cycle is divided up into separate work packages. The work packages will be elaborated upon in Section 7.2. To aid the overview every configuration item has a maturity that should be recorded, in this case it is recorded by Technology Readiness Level (TRL).
The domain of the configuration item is the discipline which is needed to perform work on the configuration item. For example, for the design of the electronics of an On-Board Computer (OBC) expertise in electronical design is needed. Thus the domain is “electronics”. Other examples of domains are “systems engineering”, “thermal engineering” and “programming”. The property type reflects whether the configuration item is developed or non-developed.

The documentation should be linked to the configuration item as well. A list of documents related to a configuration item gives new team members a quick overview of the work done and give cooperating team members a fast way to find what they are looking for.

### 7.1.3 Configuration Items, Hardware & Software

As noted in the previous section, the configuration items are a aggregation of hardware. The configuration items should be linked to the elements in the hardware breakdown. Every entry in the hardware breakdown is an hardware elements. These elements represent the designed hardware and all its properties. These hardware elements are linked to the hardware items. A hardware item is the actual piece of hardware the hardware element represents.

For example: The satellite body can be considered a configuration item. The hardware element that is included in this configuration item is the assembled satellite structure. One of the hardware items of this element is the baseline satellite body. Another one is the flight model satellite body.

These items are handled and tested by team members. Sometimes they will be mishandled and sometimes fixes need to be made. In order to control this the hardware items need to have a log that documents what happens to the hardware item.

A similar approach is taken for software. The software element represents the functions and their performance. The software items represent the actual code and properties. Software items will mostly distinguish themselves from each other in terms of version numbers.

### 7.1.4 Configuration Item Identification

How configuration items are identified depends on the approach that is taken for the systems engineering. For the university satellite methodology this approach is outlined in Section 7.3. The initial configuration items will encompass the known system, for example the breakdown as shown in Section 7.1.5. It can also include lower level systems that have been selected in the very early stages, for example payloads to be flown on the mission.

The identification should take into account the mapping of the configuration item on hard- and software, since as mismatch between them will lead to problems as was observed the Delfi-C³ project, as described in 3.4 [10]. A method to ensure this, is to first make the hardware and software breakdown, and subsequently identify the configuration items.

Once the configuration items are defined, work packages should be determined based on these configuration items. From this process it can be determined whether the configuration items need to be revised.
Figure 7.1: Configuration item properties and relations
7.1.5 Configuration Item Breakdown

Once a system concept is established, a break down can be generated. The actual breakdown is mission dependent, though all university satellites have a number of configuration items in common. The breakdown of a mission into segments is shown in Figure 7.2. As noted in Section 3.4 as well, this encompasses a system of systems. The segments are made for different objectives. Four segments are identified, which will be described in further in this section.

**Launch & Orbit Segment**

The launch and orbit segment considers the systems that are responsible of bringing the satellites into space. Since university satellites are often launched piggyback, the launch provider has a major influence on the orbit of the system. The system will have to conform and interface with the launcher. This is done either through a launch adapter or an interface on a mother satellite. These three elements can be considered the subsystems of the Launch & Orbit Segment (LOS), as is shown in Figure 7.2. For a specific mission either a launch adapter is used or a mother satellite interface. These three elements are separate configuration items; they are controlled by separate external entities, who deliver their own set of documentation. The launcher, launch adapter or mother satellite are not specifically made for the mission and likely hardly adapted, thus these are non-developed configuration items.

This definition is slightly broader than the definition given for the launch segment in ECSS [66]. The launch provider has a major influence on the orbit, and this is reflected in the definition.

**Satellite Segment**

The satellite segment compromises all hardware, software and functionality that is placed in orbit. All of these have to perform in orbit, and their interfaces tightly controlled.
satellite segment can be broken down in 2 categories of subsystems, payloads and bus systems. This is shown in Figure 7.2. Payloads compromises all hardware, software and functionalities that is used for experiments and observation. The Spacecraft bus compromises all hardware, software and functionalities that support the payloads.

### Ground Segment

The ground segment comprises all hardware, software and functionality related to systems on the ground that communicate with the space segment, whilst in operation. This system is not broken down further, since the configuration depends on mission specific design decisions.

### Test & Support Segment

The test & support segment consist of all hardware, software and functionalities that support the system during development on the ground. This is for example a clean room, but can also be as simple as a support bracket. These elements are often developed for a wider scope than just this single mission.

### 7.2 Work Packages

The work needed to be done on a configuration item can be broken up into work packages. The work package is shown in the definition diagram in Figure 7.3. A work package is owned by a single person who will execute the work package, under supervision of one or more supervisor. The work package has an (estimated) start date and an (estimated) end date. The start and end date are dependent on the workload of the work package, but also on the availability of the responsible person and, for example, product delivery.

The work package is part of a configuration item, and as such has a domain property. The package will result in a report, possibly other documents and designed items. Amongst others, the latter can be hardware, software, or models of the configuration item. There are a number of special type of work packages, which are related to the use cases of the 'student' stakeholder roles, which can be found in Section 5.4. By forming work packages in this way, clear and unambiguous assignments can be written for students, as was required according to Section 3.16.

The work packages should be related to a life cycle of the team members if applicable. For students these can be found in Section 5.2.1. The work package has its own life cycle, and thus work package should have gates and phases. The gates are deadline and review moments, and the phases are used to describe which activities are expected from the student.

The work packages need to have clear deliverables, such that completion of the work packages can be checked. Due to the result of the trade off in Section 6.2, the main output of a work package will be the report of thesis, internship or assignment. The content of these should be established for every work package. If any hardware, software or models need to be delivered these need to be included in the deliverables as well.
Figure 7.3: Work package definition diagram.
A configuration item should only contain work packages that have to be performed sequential. If this is the case it would be worthwhile to separate the configuration item in two separate configuration item, as given in the definition on page 68.

Work packages should be documented including a description and all properties in a central location. For easy assessment the work packages can be kept in a list, an example of which can be found in Appendix E.

7.2.1 Work Package Identification

Once a configuration item is known, the related work package can be identified. Before the work packages of a configuration item can be established, the available work package types for the project should be identified. This is dependent on the project and educational institute. For example in the Delfi programme, the work is mainly performed by thesis students and interns. The work a student does in a thesis or internship has been established by the universities. These can thus be used as work package types. The established work package types should be divided up into phases and life cycles, as has been done in general in Section 5.2.1. This means the gates and phases of the work package line up with the gates and phases of the student life cycle.

The first step is to identify the life cycle of the configuration item. The development of the configuration item should be divided in a number of phases, separated by gates. This is similar to the work packages. The next step is to map work package life cycles on configuration item life cycle. This likely is not an entirely sequential activity; the life cycle of the configuration item can be adapted to better fit the work package life cycles. This might mean that extra activities are added to the life cycle of a configuration item, in order to fit a work package life cycle. For example, the initial thermal analysis of the satellite might not be enough work for a thesis assignment. In order to increase the work, the student could make measurements on the thermal conductivity of key thermal components. For smaller tasks that remain after the distribution of the work packages a solution should be found. It is expected that the available manpower of the initiating institute plays a large role here. These tasks could, for example, be performed by technical staff, Doctor of Philosophy (PhD)s, or volunteers.

7.2.2 Work Package Control

There are several causes that might warrant changes in the work package. If the design of the system changes, the configuration items will be affected, and in turn the work packages should be adapted. If a student fails to perform the (complete) work package, subsequent work packages need to be changed as well.

The progress of a team member can be gauged at the gates in the life cycle. During these review the progress of the work package should be compared to what was expected in the work package, this process is shown in Figure 7.4a. If the work package is on track, there is no need to do anything. If the work done already has shown major design changes are needed, the configuration should change and a new configuration item definition is required. This will lead to new work packages as well.

---

1This was in fact done by a student working on the DelFFi satellite [67]
If the work package is delayed for any other reason. One of three options can be chosen. The work package can be extended. This will mean the project and team member work is delayed. The work package can be restarted, either with the same or a new team member. This will mean the project work is delayed. No other changes in the work packages are needed for these two options. A third option is to change the the content of the work package. This will change the planning as well, but on top of that this option will change the content of the successive work packages. It should be noted that due to the baseline approach outlined in Section 7.3, the impact of these decisions on the system are limited to the design iteration.

7.3 The Baseline Life Cycle

In Section 6.3 it was established that a evolutionary baseline model approach would be best fitted for a university class satellite project. This section will present the implimentation of this model approach into a university satellite methodology.

7.3.1 Baseline Initialisation

The baseline evolution approach need to be initialised. A university which has made previous satellite can use these as a baseline, if this is not the case the baseline needs to be initialised, a flow chart depicting that process is shown in Figure 7.5. In the kick off period a concept baseline will be defined. The concept baseline is a paper system which defines the system in terms of configuration items. The concept baseline stands in for the verified baseline and will be adapted during the baseline initialisation. For every item it will be established whether it is developed or non-developed. Additionally, initial work packages for the early development, detailed design and integration & test phase need to be established.

The concept baseline needs to take several elements into account, not all of which are shown in Figure 7.5:

- Interface and requirements of future payloads
- Research need of initiator (e.g. University)
- Technical capabilities of initiator
- Financial resources of initiator
- Systems available on the market

When the concept baseline is approved the non-developed items of the system can be acquired. In Figure 7.5 this is shown in the early development phase, though the activity can also be performed in the detailed design phase. It should be noted that the concept baseline should be treated the same as the normal baseline; the design work will only be taken into account if it can be successfully integrated into the baseline and the baseline should only be subject to improvement.

The developed configuration items will go through several design phases. Each of these design phases is ended by a review or verification. If these are positive the design iteration is included in the baseline, and the design can continue into the next phase. If the review concludes the design is insufficient, a redesign is needed. It should be noted that depending on the project
(a) The work package review process.  

(b) Tasks to perform before to initiate an iteration

Figure 7.4
and context a specific approach to this can be developed. For example, the element could be replaced by a COTS element to save time, or the complete phase could be redone for this element in order to keep in line with the student life cycle.

The system can be integrated when all the subsystems have been developed, or sooner if possible. Once this system is verified there is a bus baseline. It is empty since it is missing the payload subsystems. In Figure 7.5 the development of the payloads is not shown. Once these are complete, and verified, these can be integrated into the bus baseline. A verification of the complete system will lead to the mission baseline.

### 7.3.2 Baseline Iterations

The approach to the evolutionary baseline has already been shown in Figure 6.2. Bouwmeester and Gill [36] have defined an approach to every iteration which is shown in Figure 7.6. To perform the iterations the same approach can be used which is used for a complete system, which is shown in Figure 7.4b.

The baseline iterations should take into account the impact an iteration has on the complete system. For example for nearly every iterations an analysis will need to be made of the impact of the change on the thermal budget. This analysis can lead to changes in the surface properties of other subsystems.

### 7.3.3 Payloads and the Baseline

Payloads can be selected at appropriate times, which will address the concerns raised in Section 2.2 on the timeliness of the launch. The payloads can undergo similar iterations as done for the bus subsystems. The baseline should be ready to launch at any moment, thus technical demonstrators can be added at any time. Since there are more piggyback launch opportunities, the launch can be timed such that the mission is still relevant. This way the baseline approach can satisfy the needs of technology demonstrators as was raised in Section 2.2.

### 7.4 Requirements Management & Control

The requirements form the centre of the design, as can be concluded from Chapter 3. The definition diagram of the Requirements is shown in Figure 7.7. In the Delfi-n3Xt project the requirements were related to the configuration items, this is also the case in this methodology. What was lacking in the Delfi projects was the initial connection between requirements and verification methods, which is included in Figure 7.7. The requirements should be made to satisfy a need, and should be derived from a source document to encourage traceability. Requirements can be broken down into lower level requirements.

When a requirement is defined, it might contain unknowns. For example the satellite might need a certain pointing accuracy to get the needed ground resolution, but the accuracy is dependent on the chosen orbit, thus the accuracy requirement remains To Be Determined (TBD) up until the orbit is determined. These known unknowns can frustrate the engineers,
Figure 7.5: Flowchart of the start of the baseline approach. Baseline Imp. stands for Baseline Improvement. This lane is the same for every subsystem that the institute develops. For simplicity, not all design loops have been included.
as was seen in the DelfFi project, thus it needs to be clear when these can be resolved. Thus the prerequisites and estimated resolution date need to be determined for every TBD.

The requirements should mention any standards that need to be followed in the design. The requirement should note where to find the definition of the standard so team members do not spend time looking for it. This is needed to comply with the use case ‘Standard compliance’ as presented in Figures 5.12 and 5.15.

In the Delfi programme the requirements were successfully collected in a single list and this approach should be continued. An example template of such a file is contained in Appendix E. This file includes a link to the configuration items and TBDs.

### 7.5 Design Control

The design needs to be controlled, so that the system does not exceed the main parameters. One of the central tools for this in spacecraft design is the maintenance of budgets, which was represented by the ‘Technical parameter control’ use case in Section 5.4.1.
Figure 7.7: Requirement definition diagram.
Mass control

A mass budget is used to control the mass of the system. This has proven useful to the Delfi projects and has been uniformly implemented in these, as can be read in Section 3.6. For every configuration item with a hardware element the following characteristics are recorded:

- amount
- unit mass
- best estimate mass (of all components)
- maturity
- contingency (related to maturity)
- allocated mass (best estimate mass including contingency)

Changes in these should be recorded as soon as they are available. Documentation is only updated at the end of a work package, which might mean that waiting for the official documentation to be released would be too long. These values should thus be updated at every review for the configuration item. Note that the information update is limited to the iteration configuration items. The baseline configuration items mass properties are only updated when the baseline is updated.

Power control

A power budget can be used to control power consumption and generation of a satellite. As can be read in Section 3.6, there was no consistent approach to power control in the Delfi projects. Since the approach taken in Delfi-n3Xt is very similar to the mass budget, this approach is advised. Beside power characteristics of the hardware elements, the power budget should also contain information on the subsystem and system modes of the satellite.

For every subsystem the following data is recorded:

- subsystem modes
- maturity (TRL)
- contingency
- best estimate power consumption
- allocated power consumption (power consumption including contingency)
- peak power consumption

The subsystem modes should be related to the system modes.

7.6 Documentation Control & Information Sharing

The documentation and its control form a central element in the engineering process, as can be concluded from 3.15. A document has an author, a publication date and a reference number. From the trade-off in Section 6.2 it follows that the main documentation is assignment related documentation. From the perspective of the project this would mean that the main documentation is the document that results as final report on a work package.
7.7 Documentation Storage

Documents should be stored in a central location accessible to all team members. Documentation on the current baseline should be stored in a single folder. Every iteration track should have its own designated folder. All old baselines need to be archived. Since this folder is the source of all design information, it should be controlled by a select set of moderators. Information from external entities should also be stored in this document archive. This ensures that these documents are not lost and easily accessible by team members.

The documents should have a unique reference identifier and a version number, even if for many documents there are no iterations. Together with the above mentioned documentation structure, these measures satisfy the 'Consult existing information' use case presented in Figure 5.13.

7.8 Reviews

As was discussed in the previous sections, reviews form an important part of the university satellite methodology. There are different kind of reviews incorporated in this methodology. A visualisation of the reviews is shown in Figure 7.8. The reviews proposed here are placed at major decision points in the life cycle.

7.8.1 Work Package Review

As has been mentioned in Section 7.2, the gates of every work package line up with gates of the student life cycle. This means that there is no double reviews required. When these reviews are held is thus determined by the educational review cycle, not by specific technical milestones. Even though these reviews are mainly held to judge the work of the student, it is an opportunity to inform other team members about the progress and thus these people should also be involved.
7.8.2 Baseline Reviews

Baseline reviews are held at crucial points that affect the baseline. A Baseline Kick Off is held at the start of the project. This review establishes broad outlines of the initial baselines, and is also reflected in Figure 7.5. The Baseline Acceptance Review ends the baseline initiation period. This review checks the baseline design and determines whether it is indeed a verified baseline.

When an iteration is proposed, an iteration review is held. The credibility and desirability of the iteration is checked. This meeting approves whether resources can be allocated to the iteration. After the iteration is complete, an Iteration Acceptance Review is held. If this review accepts the iteration, a new baseline version will become available. During this review the baseline documentation will be updated.

At any point a Launch Readiness Review can be held. This reviews determines whether it is desirable to launch the system. A good moment to have this meeting is when a launch opportunity becomes available. When the baseline system is accepted for launch, the system undergo qualification testing. It will be transported to the launch site and brought into orbit. At the end of the early operations phase, a review will be held to commence the operations. This review documents the deployment and any irregularities.

When the Satellite has performed its mission and is no longer supported, the End of Life review will be held. This review officially ends the mission and archives all related data and documentations. The close out phase is initiated to shut down systems and perform analyses on the results of the mission.

7.9 Meetings

7.9.1 Team Meetings

Regular meetings were held in all Delfi projects to keep the team informed, as can be read in Section 3.14. Meetings are held to keep the team informed about the project progress and are informative in nature. The regular meetings should be kept short and to the point. Participants desire an agenda [47] and this should help to keep the meetings on topic. Minutes should be made in order for members that are not present to inform themselves.

7.9.2 Expert Meetings

To have more in depth discussions meetings are held between team members that are involved in the decision or topic. In Section 3.14, it has been observed that these occur naturally and do not need steering by a methodology.

7.10 Example: Deployable Solar Panels

In this section an example will be given of a single iteration. In this example a satellite will be given deployable solar panels to increase the power input. This example is chosen because
<table>
<thead>
<tr>
<th>#CI</th>
<th>name</th>
<th>TRL</th>
<th>Domain</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS</td>
<td>System</td>
<td>6</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.LOS</td>
<td>Launch &amp; Orbit Segment</td>
<td>9</td>
<td>Systems</td>
<td>non-dev.</td>
</tr>
<tr>
<td>SYS.GS</td>
<td>Ground Segment</td>
<td>9</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.TSS</td>
<td>Test &amp; Support Segment</td>
<td>6</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.TSS.1</td>
<td>Solar Panel Release Test Setup</td>
<td>2</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.TSS.2</td>
<td>ESATAN</td>
<td>9</td>
<td>Thermal</td>
<td>non-dev.</td>
</tr>
<tr>
<td>SYS.TSS.3</td>
<td>Shaker Table</td>
<td>9</td>
<td>MechS</td>
<td>non-dev.</td>
</tr>
<tr>
<td>SYS.TSS.X</td>
<td>Other Systems</td>
<td>9</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT</td>
<td>Satellite Segment</td>
<td>6</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS</td>
<td>Spacecraft Bus</td>
<td>6</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1</td>
<td>EPS</td>
<td>6</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.1</td>
<td>EPS board</td>
<td>9</td>
<td>Elec.</td>
<td>non-dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.2</td>
<td>Battery</td>
<td>9</td>
<td>Elec.</td>
<td>non-dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.3</td>
<td>Solar Panel Assembly</td>
<td>2</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.3.1</td>
<td>Solar Cells</td>
<td>9</td>
<td>Elec.</td>
<td>non-dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.3.2</td>
<td>Solar Panel Mechanics</td>
<td>2</td>
<td>MechS</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.3.3</td>
<td>Solar Panel Release Electronics</td>
<td>2</td>
<td>Elec.</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.3.4</td>
<td>Solar Panel Release Software</td>
<td>2</td>
<td>Software</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.2</td>
<td>Thermal Control Subsystem</td>
<td>6</td>
<td>Thermal</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.3</td>
<td>Satellite Structure</td>
<td>6</td>
<td>MechS</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.4</td>
<td>CDHS</td>
<td>6</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.4.1</td>
<td>On board software</td>
<td>6</td>
<td>Software</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.5</td>
<td>ADCS</td>
<td>6</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.BUS.X</td>
<td>Other systems</td>
<td>8</td>
<td>Systems</td>
<td>dev.</td>
</tr>
<tr>
<td>SYS.SAT.PL</td>
<td>Payloads</td>
<td>8</td>
<td>Systems</td>
<td>dev.</td>
</tr>
</tbody>
</table>

Table 7.1: Configuration items for the 'deployable solar panel iteration' example.

multiple domains are involved in such an iteration and the deployable panels will have a system wide impact.

The current baseline has body mounted solar panels. For simplicity it is assumed that the baseline satellite has an Electrical Power Subsystem (EPS) board and batteries that can handle additional solar panels. Similarly it is assumed that the on-board computer has interfaces available to command the release mechanisms. The addition of a deployable solar panel would have a desired effect on the power budget, but will also affect the thermal and structural characteristics of the satellite. On top of that, the on-board computer will have to take into account the deployment in LEOPS and send the required telemetry.

The configuration items involved in this iteration are shown in Table 7.1. The responsible institute and source for the configuration items have not been included since this is a fictional example. Only configuration items (and their parents) that are affected by this iteration are explicitly named. For the configuration items a TRL of 2 has been selected when the design has not been made yet, and 6 when this iteration changes the existing baseline design.

N. van der Pas

Master of Science Thesis
Table 7.2: Work Packages for the 'Deployable solar panel iteration' example.

<table>
<thead>
<tr>
<th>#CI</th>
<th>#WP</th>
<th>Name</th>
<th>Domain</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS.SAT.BUS.1.3</td>
<td>DSP.1</td>
<td>Concept Definition</td>
<td>Systems</td>
<td>thesis</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.3.4</td>
<td>DSP.2</td>
<td>SP Release Software Design</td>
<td>Software</td>
<td>intern</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.3.3</td>
<td>DSP.3</td>
<td>SP Release Electronics Design</td>
<td>Elec. intern</td>
<td></td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.3.2</td>
<td>DSP.4</td>
<td>SP Release MechS &amp; hinge Design</td>
<td>MechS</td>
<td>intern</td>
</tr>
<tr>
<td>SYS.SAT.BUS.2</td>
<td>DSP.5</td>
<td>Thermal Analysis &amp; Design</td>
<td>Thermal</td>
<td>thesis</td>
</tr>
<tr>
<td>SYS.SAT.BUS.3</td>
<td>DSP.6</td>
<td>Structural Analysis &amp; Design</td>
<td>MechS</td>
<td>thesis</td>
</tr>
<tr>
<td>SYS.SAT.BUS.1.3</td>
<td>DSP.7</td>
<td>Solar Panel Integration</td>
<td>Systems</td>
<td>intern</td>
</tr>
<tr>
<td>SYS.TSS.1</td>
<td>DSP.8</td>
<td>Deployment Test</td>
<td>Systems</td>
<td>thesis</td>
</tr>
<tr>
<td>SYS.SAT</td>
<td>DSP.10</td>
<td>Baseline Integration</td>
<td>Systems</td>
<td>thesis</td>
</tr>
<tr>
<td>SYS.SAT.BUS.4.1</td>
<td>DSP.9</td>
<td>Software integration</td>
<td>Software</td>
<td>task</td>
</tr>
<tr>
<td>SYS.SAT.BUS.5</td>
<td>DSP.11</td>
<td>ADCS impact analysis</td>
<td>Systems</td>
<td>thesis</td>
</tr>
<tr>
<td>SYS.SAT.BUS.5</td>
<td>DSP.12</td>
<td>ADCS update</td>
<td>Software</td>
<td>task</td>
</tr>
</tbody>
</table>

The work packages related to this design are shown in Figure 7.9 and summarised in Table 7.2. For this project there are three types of work packages:

**Thesis** The thesis work package refers to a seven month graduation report from a student for a Master of Science (MSc) accreditation. This is similar to the thesis defined in Section 5.2.1.

**Intern** The intern work package refers to an student performing a three month internship in the project. This is similar to the intern life cycle presented in Section 5.2.1.

**Task** The task work package is a small work package that cannot fit into a normal life cycle. The task will have to be performed by a volunteer or staff member.

From the work packages a planning can be generated using Program Evaluation and Review Technique (PERT). From Figure 7.9 the critical path can be determined, and is shown with the thick arrows. It should be noted that the 'Panel Integration' and the 'Deployment Test' package can run partly parallel, since the test will only need the panel after the test setup has been created. If it is assumed that these two work packages start at the same moment, the minimal length of this iteration is 28 months.

A requirement related to this example is the temperature range of the solar cells on the panels. This requirement is given below with a simplistic identifier.

**Requirement R1** The temperature of the solar cells on the deployable solar panels shall be maintained between <TBD1: 123> and <TBD2: 383> Kelvin.

It is assumed here that the solar cells have not been selected yet. This means that the exact temperature range is not known, and thus this example requirement contains two values that need to be determined. For these unknowns for now estimates have been entered. The TBDs are described in Table 7.3. The TBD can be determined after the solar cells have been selected. This is done in work package DSP.1, which is also noted in Table 7.3 under prerequisite.
Figure 7.9: Swim-lane diagram of the work packages in the 'deployable solar panel iteration' example. Every lane is a configuration item. The thick arrows follow the critical path.

Table 7.3: TBDs for the 'Deployable solar panel iteration' example. The (estimated) resolution dates and sources have been omitted since this is an example.
This section will answer the research questions in Section 8.1 and discuss the implication of the results in Section 8.2.

8.1 Conclusions

The conclusions will be presented in the following section. The three sections here will answer the three research questions presented in Section 1.3.

8.1.1 University Satellite Projects

University class satellites are mainly initiated for the education purposes, but can include technical demonstrators, science missions and services. The projects give hands on experience. This approach, however, might cause conflicts with the existing educational structure. The team that design the systems are small and have a high turnover. Several domains require to be involved in the projects. Team members bring needs to the project that might not align with the direct needs of the project.

Configuration items and requirements were central in the systems engineer for the Delfi projects. Both improve the shared mental model of the team. Configuration items need to reflect the actual hard and software of the system.

Due to the high turnover of the team in university satellite projects, documentation has a big role in information transfer. The documentation can form a burden for the team. During the design, team meetings are an important element to share information between team members.

8.1.2 University Satellite Context

The stakeholders were analysed with the use of stakeholder roles. The additional needs stakeholders bring to the project can be represented by secondary stakeholder roles. For
example the educational needs of a student stakeholder can be represented this way. These
stakeholder have their own life cycles which need to be integrated with the university satellite
methodology.

The key roles in these projects are the engineer, manager, student and researcher role. For
the key roles the use cases in relation to the university satellite methodology. The engineer is
involved in use cases related to the design of a system and the implementation of a method-
ology. The manager controls the team, the planning and the financial budgets. The students
want to perform educational assignments in the project. The researcher wants to perform
research while developing the system.

The success of a university methodology can be measured by the success of a number of
parameters, related to the performance of the project, the performance of students in the
project and the success of research related to the project. The performance of the project can
be parametrised by looking at the quality of the system, the project length and the project
cost. The performance of the students can be measured with their grades and the time they
spent on their assignments in comparison to their peers.

8.1.3 University Satellite Methodology

The reports students generate as part of their education should be central to a university
satellite methodology. This will generate a minimal amount of additional work, both in terms
of documentation and reviewing. During the work of the student information is easily shared
through meetings between team members.

In order to generate a robust methodology that is still flexibly in terms of work allocation, a
baseline model approach should be taken. The baseline approach has a set baseline system on
which iteration are performed to improve parts of the system. An iteration is only accepted
as part of a new baseline when it has proven to improve the system.

The system is represented by a breakdown of configuration items. The configuration item
reflect hard and software. A single configuration item consist of one or more work packages,
which represent the work that needs to be done to complete the work package. These work
packages are based on the life cycles introduced by the team members.

8.2 Recommendations & Discussion

Within the constraints of a master thesis sadly no validation of the proposed methodology
could be performed. It is advised to record the parameters given in Section 5.5 even before
implementation of this methodology to get a grip on the performance of the project. When
the data is available this methodology can be compared to this data to gain insight into its
performance. Due to the length of a project this is likely not work that can be done by a
bachelor or master student. Elements of the proposed methodology might be verified using
agent based modelling. With this method the team itself will need to be simulated.

The work presented here does lean heavily on the context of the Delft University of Technology
(TU Delft). It is expected that this will not give major problems for the implementation of
the university satellite methodology. Nevertheless it should be checked that the context in
which this systems engineering approach is implemented does not greatly diverge from the context presented in this report.

Similarly, the proposed methodology has assumed that the team members do not have a background in Model Based Systems Engineering (MBSE). If MBSE becomes common knowledge, it is likely that the proposed document based approach can be superseded by a model based approach. A system model can improve the timeliness of design updates and the common understanding of the satellite system.

While certain parts of the methodology proposed in Chapter 7 are specific to space projects, the core elements can also applied to other project in a university context. By organising the work with configuration items and work packages research projects will easily manage to combine education and science. Similar, technical developments in universities can robustly made using a baseline approach.

The proposed methodology has not been designed to finish the project as soon as possible. The planning for a project using this methodology is likely longer than that for other methodologies. This is traded for a better integration with education and more certainty in the work to be performed. In practice this might well mean this systems engineering approach has less delays and a bigger team.

The success of a project using the university satellite methodology is not guaranteed of success. The success of a project also depends on the resources that are allocated to the project, and the team members that can be attracted. The work packages and configuration items are categorised in a domain, thus the expertise for all of these domains needs to be attracted to the project. Similarly, if the work packages are unrealistic, the methodology will fail as well.
Appendix A

Systems Engineering in the Delfi programme

The Delfi programme has been chosen as a study case to research the implementation of systems engineering in a university context and to document how this is implemented in practice. The Delfi programme has been chosen because it is the satellite programme of the Delft University of Technology (TU Delft) and extensive internal documentation is available about these projects. This chapter will focus on several topics and describe how these were handled in the different Delfi projects. Note that the Delfi-C3 and Delfi-n3Xt projects have been completed, albeit the latter with a redesign. The DelFFi project is in progress during this research.

A.1 Programme Objectives

The objectives of the Delfi projects and programme are important to consider since they show why the projects are performed in the first place. The objectives cover not only the technical needs of the system, but also the additional outputs the development should generate. This section will start with a look at the objectives of the Delfi programme, which itself sets the bar for the Delfi family of satellites.

Delfi Programme

For the satellites within the Delfi programme educational, technical and programme objectives [37]. The educational objectives aim to give students both hands on experience and improving the scientific writing and communication skills of the students in order to prepare the students for a career in space. The technical objectives aim to qualify micro-technologies for space applications. The programme objectives aim to mature the nano-satellite platform.
In 2008 it was envisioned that a nano satellite should be created by the TU Delft every 2.5 years to meet these objectives[37]. It is important to note that while the Space Systems Engineering (SSE) chair took the lead, strong cooperation with Electrical Engineering, Mathematics and Computer Sciences (EEMCS) was seen as mandatory for success [37].

Delfi-C$^3$

The objectives of the Delfi-C$^3$ project have been divided in 3 categories: educational, operational and development goals. The operational goals were related to the performance of the payloads and will not be discussed here as they are system specific, the others are listed below, as found (amongst others) in reference [11].

1. Educational goals

1.1 Provide students with an opportunity to gain interdisciplinary hands-on engineering experience by providing a real-world application.
1.2 Prepare students for careers in aerospace and related industries by encouraging development of their teamwork, leadership and communication skills.
1.3 Be a motivation for students to participate in satellite projects and to choose the aerospace engineering discipline as a field of study in general and be a challenge for students to be innovative and efficient at the same time.
1.4 Interface with the MSc programs of the Faculty of Aerospace Engineering (AE) and Electrical Engineering, Mathematics and Computer Sciences (EEMCS) of the TU Delft.
1.5 Enable participation of other educational organizations in the project.

2. Operational goals

3. Development goals

3.1 Complete design of a reliable satellite bus system, according to the KISS principle.
3.2 Complete design of interfaces between satellite bus system and payloads.
3.3 Complete production of proto-flight hardware.
3.4 Complete production of ground support hardware and test equipment.
3.5 Complete and pass tests on subsystem level.
3.6 Complete and pass system level tests.
3.7 Complete ground system software.
3.8 Deliver satellite on time at launch broker Complete launch preparations in time for launch.

Items 1.1 and 1.2 relate to the experience the students should have in the project. Item 1.3 relates to both students and future students. Items 1.4 and 1.5 state that the experience of the students should be in agreement with elements of the educational institute of the student. It is important to note that item 3.1 names the Keep It Simple Stupid (KISS) principle as the primary philosophy behind the design. Even though Delfi-C$^3$ had sets of operational and development goals, it was primarily an educational project [11].
Delfi-n3Xt

The objective of the Delfi-n3Xt mission could be summarized in a one-sentence mission objective:

"Delfi-n3Xt will be an advanced nano-satellite, gathering scientific data related to the radiation from the sun and providing in orbit testing and qualification of innovative technologies." [37]

The innovative technological payloads were to be provided by universities and industry. Additionally, it was noted that the educational goal was to facilitate 20 Master of Science (MSc) and 10 Bachelor of Science (BSc) theses [37]. In a later 2011 report [12], this was revised to 30 theses or internships [12].

This 2011 report also mentioned the following mission statement:

"Delfi-n3Xt shall be a reliable triple-unit CubeSat of TU Delft which implements substantial advances in 1 subsystem with respect to Delfi-C3 and allows technology demonstration of 2 payloads from external partners from 2012 onwards." [12]

The subsystem to receive substantial advances was the attitude control system [12], though other subsystems also would receive improvements. It was expected that the test campaign could be done within three months, but the satellite was designed for two years of nominal operations [12].

DelFFi

The mission statement of DelFFi is as follows:

"The DelFFi mission shall demonstrate autonomous formation flying and provide enhanced scientific return within QB50 from 2015 onwards, by utilizing two identical triple-unit Cubesats of TU Delft which further advance the Delfi-n3Xt platform." [39]

The DelFFi mission applies the objective from the Delfi programme. Additionally, DelFFi is intended to perform within the QB50 programme [39]. Three main objectives were defined [38]:

1. Demonstrate autonomous formation flying using various Guidance, Navigation and Control (GNC) architectures.
2. Facilitate students with hands-on experience and cutting-edge technology.
3. Characterize low thermosphere with enhanced scientific return by using distributed observation on various geometric baselines.
The first objective is a technological objective and the second an educational objective. The educational objective was satisfied when 30 theses and internships could be facilitated by the project [38]. The third objective is a scientific objective, which makes DelfiFi the first to have such a scientific objective. This could be an indication that the Delfi project are maturing, though it could also only be a result of the participation in the QB50 project.

According to the project manager, the emphasis of DelfiFi is mostly placed on achieving the technical objectives, and less on the research objectives [68].

Conclusion

The objectives of the Delfi satellites are divided in three categories; educational, technical and programme objectives. The technical objectives are project dependent, and aim to test new technologies. The educational aim to give students hands on experience in teamwork and with satellite development and thus represents the student needs. In practice this translates to facilitating thesis work within the project. The programme objectives aim to mature the nano-satellite platform, and are thus related to engineering research. The technical objectives got more attention, and the programme objectives less. In the latest project DelfiFi also a scientific objective is set, which could indicate a maturing platform, but could also only be motivated by the partnership with QB50.

A.2 Participants

This section will take a short look at the participants in the different projects, and what their role was. More research into this topic is presented in Chapter 4 and Appendix B, but this section will look at what the contribution of the participants was. This section will be limited to the most important partners, omitting component providers. Naturally, all projects were conducted at the TU Delft, with students of the TU Delft.

Delfi-C³

In the Delfi-C³ project staff from Aerospace Engineering (AE) and EEMCS were involved as project management, support and supervision. MSc students from both faculties were involved in the design of the subsystems. Bachelor of Engineering (BEng) students from several institutes were involved in the design as well. EEMCS also provided the Advanced Transceiver experiment.

In the satellite an Pumpkin Inc. computer board was used [11]. The 3-unit structure was also bought from Pumpkin Inc., this required modification. Subsystems used a microcontroller from Microchip, which was given as in-kind sponsoring [11]. California Polytechnic State University (CalPoly) acted as launch provider.

The project had found industrial partners in SystematIC BV Design, Dutch Space and Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek (Netherlands Organisation for applied Scientific Research) (TNO). Dutch Space produced and provided the Thin Film Solar Cells (TFSC) experiment [69]. The TFSC experiment was identified
as the primary payload [7]. Dutch space also provided support in the field of mechanical engineering. An employee from Dutch Space was hired in order to design circuit boards [11]. SystematIC BV was directly involved in the design of the Electrical Power Subsystem (EPS) [8].

TNO was another primary partner, providing the Autonomous Wireless Sun Sensors (AWSS) experiment. The majority of the design was to be done by students from the faculty of EEMCS, based on spare parts from another mission [7], TNO did put a number of employees on this project. For this payload TNO was identified as partner, supervisor and customer [7]. Additionally test facilities of TNO were used for vibration testing [70]. Other tests were performed at Nationaal Lucht- en Ruimtevaartlaboratorium (National Aerospace Laboratory) (NLR) [71].

Additional support was given by several radio amateurs. this was done during the design, but also during operations. The radio amateurs received passed data through to a central server in Delft [11].

**Delfi-n3Xt**

The Delfi-n3Xt satellite was designed by the SSE chair of the faculty of AE of TU Delft. There was strong cooperation with the faculty of EEMCS and some cooperation with Mechanical, Maritime and Materials Engineering (3mE) [12].

The Delfi-n3Xt carried two payloads from external partners [12]. A propulsion system, T³μPS, was developed by TNO in cooperation with the TU Delft and University of Twente (UT). A transceiver platform was designed by Innovative Solutions In Space BV (ISIS), and developed in cooperation with TU Delft and SystematIC BV. Additionally a solar cell experiment was carried from Delft Institute of Microsystems and Nanoelectronics (DIMES). This was a conditional experiment, in that it would be carried unless it became a design driver [12].

Dutch Space provided solar cells and integration of the cells on a Printed Circuit Board (PCB) substrate[12]. The communication subsystem was produced by ISIS [16]. Some of the tests were performed at Thales and NLR. The launch of the satellite was contracted with Innovative Space Logistics BV (ISL), a subsidiary of ISIS.

**DelFFi**

The DelFFi satellites were mostly designed by staff and students from the faculty of AE. BEng students were involved in the design as interns. A limited amount of MSc students from EEMCS was involved as well. The staff was involved to supervise, design and manage the design.

Dutch Space provided solar cells and integration of the solar cells. A number of other subsystems were procured at ISIS.

A major participant in the DelFFi mission was Von Karman Institute (VKI) through the QB50 mission [38, 39]. The QB50 mission provides the launch and launch integration services. QB50 also provided the FIPEX and GAMALINK payloads. The QB50 mission constrains the design by placing requirements upon the satellites and by setting a schedule to be met. The
TU Delft supported QB50 with advice and participated in technical meetings and workshops [38].

Another major payload was the propulsion module, which was aimed to be designed by students from the chair of SSE.

Conclusion

The Delfi satellites were engineered at the TU Delft, with the chair of SSE involved as the project lead. Other faculties were involved to provide specific expertise. All Delfi satellites carried payloads for other partners. Some testing was performed at external institutes. Solar cells and the launch were brokered through external partners for all missions. Throughout the different satellites, more subsystems are procured. The DelFFi mission is an exceptional case in that it has one partner (QB50/VKI) which provides payloads, launch and requirements. What is notably missing, is the involvement of staff as researchers. The satellites did not carry a research payload from University staff or Doctor of Philosophy (PhD)s nor was their research connected to the satellites, with the exception of the propulsion system on DelFFi.

A.3 Systems Engineering Philosophy

Delfi-C³

Of six basic principles identified in the Delfi-C³ kick off meeting, two are important to mention in this context:

- 'Mission shall not depend on new technology flown' [72]
- 'Joint university-industry low-cost project' [72]

The primary item is an important statement about the engineering philosophy that will be followed, while the latter identifies the kind of project that is being pursued.

For Delfi-C³ an "flexible, on-the-fly (systems) engineering" approach was chosen. This meant that a baseline approach to systems engineering was chosen and subsequently all systems engineering processes (based on existing standards) would be adopted and tailored where necessary while the project progressed [7].

At the start the systems engineering tasks were envisioned to be split over three different systems engineers [7]. Later it was established that at least 2 systems engineers should be present at all times [8], needing at least 4 in the project planning.

The main approaches during the design were [7]:

- KISS approach
- Modularity approach, both for system breakdown and work package definition.
- Accessibility approach
- commercial of the shelve (COTS) and Space Hardware of the Shelve (SHOTS) approach
The KISS approach worked very well for a number of subsystems, but in other subsystems a seemingly simple solution created more complexity in the end [11]. This might have occurred due to a lack of understanding of the complexity, or because the initial approach to the problem did not remove the complexity, but moved it to a different level of the design [52].

**Delfi-n3Xt**

It was found that most of the systems engineering philosophies applied to Delfi-C^3^ were also applicable to Delfi-n3Xt [10]. Since no further explicit information on this topic was found it is assumed this was applied.

**DelFFi**

Not many systems engineering philosophies were defined for DelFFi. The operational concept strived for simplicity, complex options would be disregarded and only optionally used at the end of the mission [38].

**Conclusions**

Really only for Delfi-C^3^ the approach to the engineering of the satellite was explicitly stated. For the KISS approach it was concluded that it could not successfully be applied to all subsystems.

### A.4 Configuration management

**Delfi-C^3^**

To manage the configuration and design in the later phases of the design, configuration control was implemented. For this the system was broken down into configuration items [7]. The configuration item breakdown is shown in Figure A.1. It can be seen that every configuration item as an identifier, except for the system. The configuration items are colour coded for with respect to the responsible subgroup, which can either be a cooperating institute or a group within the team. Next to every item the responsible persons are shown. The system was broken down in 3 segments, a launch segment, the satellite and a ground segment. The launch segment was broken down in the launch vehicle and the deployment pod. The ground segment in support equipment and a ground station network. The satellite segment was broken down in a number of subsystems. It can be observed that the Delfi-C^3^ system is a system of systems, since the launch segment is not designed for this system in particular. The ground system itself also has wider objective than the satellite, and is a system of systems because the test facilities are not necessarily designed only for this mission.

After every review a baseline was established [7]. The baseline could be changed by following the procedure detailed in Figure A.2. Part of this procedure is the Configuration Control Board (CCB), which was not a permanent board [7], even though the systems engineer and project manager were always included in the loop. Not confirming to the baseline was
Figure A.1: Configuration item breakdown of the Delfi-C³ project
considered possible, but a deviation (before integration of the configuration item) or waiver (after integration of the configuration item) needed to be documented.

Additionally a hardware assembly was made to assign certain hardware elements to team members [7]. This meant two breakdown (configuration items and hardware) where used in parallel. In order to control the progress and schedule of the project the maturity of the configuration items was recorded in a table as is shown in Figure A.3.

A systems engineering process was set up for every configuration item, to be applied by the team member working on it [7]. This process is shown in Figure A.4. As input into the process generally a system or customer need was used, as output was either input for a new cycle through the process, or a change request to the system [7]. An in depth explanation of this process is omitted here for brevity, but it can be noted that the process consist of a number of distinct processes, with specified documents related to them.

Delfi-n3Xt

The Delfi-n3Xt system was also broken down in configuration items. The configuration items were identified according to the following rules, as taken from [40]:

---

Figure A.2: Delfi-$C^3$ design change procedure [7]
Figure A.3: This is an example of how the maturity during Delfi-C³ was controlled [7]

- Configuration items are conveniently defined as items that are worthwhile to control separately.
- Disciplinary homogeneity within a CI is important.
- A configuration items should not be a combination of independent components.
- The responsibility of a configuration item should be unambiguously attributed, preferably to one person.
- A division along hardware-components (or software) is preferred.
- The combination of all the configuration items must envelop the complete Delfi-n3Xt system.

Every configuration item was given a number with a prefix based on the segment the item belonged to [40]. Changes to the configuration item tree were to be resolved and verified by a responsible engineer [40]. The configuration items were initially managed using Microsoft Visio, but during the revision of Delfi-n3Xt the list of configuration items was merged with the list of requirements. The original tree was colour coded, but it was unclear to what purpose. The latter tree is shown in Figure A.5 to a subsystem depth. The same three segments as in Delfi-C³ are defined in Delfi-n3Xt. With the rules given above it is unclear why the ground segment is a single configuration item, it does not seem that the support equipment has disciplinary homogeneity with the ground station network, or why it is convenient to control them together. It can also be noted that the ground support equipment has not further been broken down in elements.

The configuration items were related to the requirements, though not for every configuration item requirements were set up.

**DelFFi**

The configuration items were approached in the same way as in Delfi-n3Xt. Additionally, in reports, the heritage of every configuration item was registered [39]. Beside the system breakdown into configuration items, there was a breakdown of the project in work packages.

N. van der Pas

Master of Science Thesis
Figure A.4: Delfi-C³ general systems engineering process [7]
A note on the work package breakdown indicates that the items on the work breakdown structure relate to the configuration items, however inspection shows that this is not the case.

Conclusions

All three projects used configuration items to make a breakdown of the project. The implementation differed along the projects. In Delfi-C3 the configuration item was used to manage the engineering and control the status of the design. Formalised processes were used for this control. In Delfi-n3Xt the configuration items were related to the requirements, and were assigned to a single responsible person. In DelFFi it was attempted to related configuration items to work packages.

The implementation of configuration items seems to have a positive impact on the engineering process, but the exact form and impact they should have is debatable. It is clear that the configuration items give a breakdown of the structure which can be related to work to be done and can be used to connect requirements, hardware and work packages. As such it can be used as the central part of the design.

A.5 Requirements Management & Control

Requirements form the basis of a technical design. The requirements are distilled from the stakeholder needs and interfacing systems. After the requirement definition, requirements can still change, or need to be updated with determined values.

Delfi-C3

The requirements for Delfi-C3 were intended to be collected in a single file, maintained by the systems engineer, to guarantee consistency. The requirement discovery was performed with
A requirement discovery tree. The requirements were categorised in general requirements & constraints, functional requirements, performance requirements, operational requirements, Reliability, Availability, Maintainability and Availability (RAMS) requirements, interface requirements and testing requirements [7]. The interface requirements for every subsystem was that it should confirm to the single interface control document which was produced for the complete system [7].

In order to relate the requirements to the verification of the systems a requirement verification traceability matrix was set up. During the verification phase this matrix was rarely used [45], more use was made of the verification specification document.

**Delfi-n3Xt**

Initially the requirements for Delfi-n3Xt were determined by the mission objectives and configuration items [43]. The configuration items were first determined after which requirements for them were established [43].

The requirements were ordered into requirement types [43]:

- **Functional requirements**
  These describe the functions of the configuration item.

- **Interface requirements**
  These determine the physical interfaces between configuration items.

- **Performance Requirements**
  These quantify how well the configuration item performs a function. For example, they set the required pointing accuracy for the Attitude Determination and Control System (ADCS).

- **Operational requirements**
  These specify the operation under which the function should be performed.

- **Constraint requirements**
  These constrain the design. For example these can exclude the use of certain materials.

- **Assembly, integration, verification and testing / reliability, availability, maintainability and safety requirements.**
  These are requirements related to the topics listed in the name.

- **General requirements**
  These requirements are the requirements that do not fall in any other type, such as conformance to the budgets.

The requirements were numbered with an identifier that reflected the configuration item to which it related and the requirement type [43]. For the budgets which were expected to change a lot (power, data, volume and mass), the requirements were allowed to refer to the specific budget, instead of mentioning exact values [43].

In order to manage the requirements an excel based requirement management tool was developed [43]. This tool featured traceability between the requirements, the status of the requirement and the source of the requirement. Only the responsible requirements engineer was allowed to enter new requirements into the requirement management tool. New requirements could be submitted to the requirements engineer through a requirement creation form, which was to be entered into a specific folder on the project disk.
This approach was changed to a simpler list due to "impracticalities ... and change of mission" [44]. The new list comprised approximately the same information, albeit in a somewhat different format. This new list did not contain automated features and the strict requirement creation form was also abandoned. The requirement types were maintained except for the Assembly, Integration and Verification (AIV) requirement type.

**DelFFi**

For the DelFFi project the top level requirements were established in the initial Phase A report [38]. Additional requirements were imposed by the QB50 project. The QB50 requirements were readily available through the project website.

All the requirements were also collected in a requirements overview. This overview was based on the requirement overview from Delfi-n3Xt. The overview was incomplete; sources were missing or incomplete, To Be Determined (TBD)s were left to be determined and requirements had incomplete or non-unique identifiers. The overview was rarely updated and changes to QB50 requirements were not incorporated. In addition no verification methods were included with the requirements. It was observed that this frustrated team members. These team members generated their own list of requirements, these however did not become available to the complete team. The list of requirements were under the responsibility of the project manager, and at one point the systems engineer, both staff members. Both have signalled that they did not have enough time assigned to the project to conclude all tasks successfully.

**Conclusions**

Both for Delfi-C3 as Delfi-n3Xt an ambitious approach was taken to requirements, only to later be replaced by a simpler approach. It is suspected that for Delfi-n3Xt the approach failed because of the cumbersome requirement change process, which put the requirement control strictly with a single person and added additional tasks to the other team members. For DelFFi the requirement management was also strictly with a single person. It can be concluded the requirement process is important, and requirements are desired by the team members, however, the requirement process should not pose additional tasks on the team members, and should not be dependent on the inputs from a single person. Remarkably none of the projects related the requirements to a verification approach.

**A.6 Design Control**

This section will detail how design control was performed in the different project, mainly by looking at budget control. As noted in Section A.5 in the Delfi-n3Xt project budgets were used to capture both the requirements for the most important parameters. Additionally budgets capture the state of the design. The Link budget will not be discussed here because it is very specific to the communications related subsystems. Similarly the thermal budget
Maturity Code | Uncertainty | Description
--------------|-------------|---------------
E             | 18 %        | Early estimated power/mass
L             | 12 %        | Estimated power/mass at time of long lead procurement
P             | 8 %         | Estimated power/mass at Preliminary Design Review
C             | 4 %         | Estimated power/mass at Critical Design Review
X             | 2 %         | Actual power/mass at Unit Assembly, Integration and Test
A             | 0 %         | Certified power/mass prior to flight

Table A.1: Power and mass margins in the Delfi-C\(^3\) project [74, 75] In practice the L margin does not seem to have been used.

Delfi-C\(^3\)

In the Delfi-C\(^3\) the power, mass and link budget were controlled.

The power budget was stored in an excel based tool. In this tool, every subsystem had a dedicated sheet which contained information on [73]:

- the board voltage
- the status of the board in the different modes
- the best estimate of the current of the components and circuits on the board
- the design status of the components and circuits on the board (see Table A.1)
- the power consumption of the board
- the efficiency of the power converter on the board.

A dedicated sheet calculates the power available and another sheet summarizes the typical and peak power of all boards and the complete system [73]. Uncertainties in the design were taken into account by giving every circuit or component a margin based on the design status. These are summarised in Table A.1.

The mass budget was contained is a excel tool as well [75]. A single sheet contained information on every assembly and its components. Similar to the power budget, every component was given a maturity code to indicate the maturity of the component and the applicable margins. These codes are given in Table A.1. The sheet stored the maturity code, the quantity of the components used in the assembly, the best estimate unit mass, the best estimate mass with and without contingency and the mass of the component previous to the latest revision. From the revision sheet it seems that this tool was only applied to halfway into the project.

Delfi-n3Xt

In the Delfi-n3Xt project a power budget, a mass budget, a link budget, a volume budget and a thermal budget were used. The Delfi-C\(^3\) tools were not reused for Delfi-n3Xt.

The power budget was contained in an excel file. It contains data on the performance of the power system. All subsystems were gathered on a single sheet. This sheet details the subsystem modes and what subsystem mode is active during what system mode. Every subsystem is broken down in a number of parts, if the part is a configuration item, it is...
Table A.2: Power and mass margins in the Delfi-n3Xt project [76]

related to this configuration item. For these parts a maturity level is set, which is detailed in A.2. For every part the best estimated power consumption for every subsystem mode is given, both including and excluding margin.

The mass budget was stored in an excel file. The file details subsystems and its components. Every component has a maturity code, similarly as those found in the power budget. For the mass the contingency differs from the contingency in power, as is shown in Table A.2. The components have the following related to them:

- amount
- unit mass
- best estimate mass (of all components)
- maturity
- contingency (related to maturity)
- allocated mass (best estimate mass including contingency)

Additionally centre of mass characteristics along the Z-axis in the CubeSat stack have been noted and in another sheet old mass budgets have been archived.

For the Delfi-n3Xt project a volume budget was kept as well. Since the Delfi-n3Xt project was a CubeSat project, the driving geometry is the height of the PCBs in the stack. The budget notes the height, location and connector location of all subsystems.

It was noted that the budgets are central to the design and keeping them up to date is a very good way of budget control.

**DeFFi**

The mass budget for the DeFFi project was a replication of the mass budget of Delfi-n3Xt. The Power budget was structured differently. In this budget every subsystem was related to a mode. Every mode had a nominal power consumption related to them. For all mission phases, such as deploy and formation flying, and the different power states, such as Sun and Eclipse, it was noted what percentage the subsystem used from its nominal power. Based on this the total consumed power and total available power are calculated. It is unclear where the parameters for these calculations come from and no contingencies are used in this budget.

---

1Conversation with J. Bouwmeester.
Conclusion

In all projects budgets were used to control the design.

The mass budget related to a hardware components, and tabulates the following characteristics:

- amount
- unit mass
- best estimate mass (of all components)
- maturity
- contingency (related to maturity)
- allocated mass (best estimate mass including contingency)

The power budget is approached differently in the different project. Delfi-C$^3$ takes the most detailed approach, calculating the power consumption from the current that the components draw. In Delfi-n3Xt the power consumptions are given per part of the subsystem. And in DelFFi the subsystem power consumption was given as a percentage of the nominal power consumption.

In both Delfi-C$^3$ and Delfi-n3Xt the power has, similar to the mass budget, contingencies to take into account uncertainties. What all budgets share is that the power consumption is related to the different subsystem and system modes. Similarly, the power budgets also include how much power the satellite can generate or store.

A.7 Model Philosophy

The model philosophy of a project outlines the models that should be produced within the development of the final system. The model philosophy is closely related to the life cycle, since the models often form the central deliverable of a phase. The development phases are described in Section refsec:LifeCycle.

Delfi-C$^3$

For the Delfi-C$^3$ project a proto-flight approach was chosen [7, 9, 36], it was expected that this approach would save time and cost. An exception was made for certain high risk subsystems and for all subsystems also a development model would be made [7]. The proto-flight model would undergo qualification testing, while the real satellite would undergo acceptance testing at the launch provider [45]. In practice most prototype hardware did not work to specification [36] and it was concluded that the proto-flight approach was not working [45].

Delfi-n3Xt

The Delfi-n3Xt project based the model philosophy on the lessons learned from Delfi-C$^3$ [37]. Two design iterations of the complete satellite were envisioned [37]. The first would result in
a satellite model, allowing for a number of open issues[37, 10]. The second design iteration should result in a complete prototype[37, 10]. Production of flight hardware was only to be commenced when remaining issues would not effect the subsystem to be produced[37, 10]. This approach did not work due to lack of time and ultimately a proto-flight approach was followed for a number of subsystems [36] and testing [16].

**DelFFi**

For the DelFFi satellites an evolutionary approach was chosen [38], more information on this can be found in A.8. A baseline would be upgraded by improving elements. This meant that for every element an appropriate model approach was to be chosen. A successful development of an element would result in a new flight model. In the end this would result in a integrated flight model. This approach failed since the baseline model was never fully established, and the system diverged to much from the previous satellite. At the moment of writing a number of engineering models are present for certain subsystems.

**CubeSat Model Approach**

Based on research in two university satellite projects Elstak proposed a CubeSat Model Approach [9]. This approach is shown in A.3. The approach aims to guarantee an early start of testing while not investing heavily in a complete prototype.

<table>
<thead>
<tr>
<th>CubeSat Model Approach</th>
<th>Basic Model</th>
<th>Protoflight Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purpose</td>
<td>Proof of subsystem functions</td>
<td>Achieve an integrated and functional architecture</td>
</tr>
<tr>
<td>Functionality</td>
<td>Subsystem subfunctions</td>
<td>Comparable to EM boards, EPS architecture functional, CHDS architecture functional</td>
</tr>
<tr>
<td>Verification</td>
<td>Verify functions</td>
<td>Qualification testing, Acceptance testing</td>
</tr>
</tbody>
</table>

**Table A.3:** The CubeSat Model Approach as proposed by Elstak [9]

**Conclusions**

Very different model philosophies have been applied to the different projects, none were successful. The team is to inexperienced for a proto-flight approach, a double prototype approach as in Delfi-n3Xt is very sensitive to delays in the project and the evolutionary approach never really took of because a baseline model was not fully established and not robust enough to the difference in mission. The model philosophy has to take into account
A.8 Development Phases & Life Cycle Model

This section will discuss the development phases and life cycle model chosen by the different projects.

Delfi-C³

The phases of the Delfi-C³ development were based on consecutive development phases. It was recognised that the industry standard phases not realistic within a student project, and thus a number of phases were planned in parallel as shown in the Delfi-C³ diagram of the development phases in Figure A.6 [7]. For the different elements the phases would still be consecutive to each other [8]. It was assumed that the different elements could be developed separately, up until the integration of the system [7]. At the start it was decided to limit full scale testing as much as possible, out of cost considerations [7]. The reviews are planned at points were it was expected that 75% of the system was at the correct maturity level for the review [7]. This way of planning was seen as a project risk which could lead to delays. As a mitigation measure progress meetings were planned and it was noted that systems that are not at the correct maturity level should be strictly controlled after the reviews. Special sessions (δ-session) were planned to control specific subsystem. A δ-session was a review at subsystem level, and would require less preparation and less people involved [8]. This seems based on evolutionary development [19]. It should be noted that at the first Preliminary Design Review (PDR) none of the subsystems were mature enough for the review to take place [7].

The life cycle model is shown in Figure A.7. Note that The project initiation and the operations phase are not included. The life cycle model is a classical Vee-model adapted to the Delfi-C³ project.
The design of subsystems was non parallel due to the different moments students join the team. In order to cope with this δ-sessions were implemented. The effect of these sessions is shown in Figure A.8. The horizontal axis is the axis of time, during a review it can be decided to have a δ-session for a subsystem. This subsystem is further developed up until in a δ-session it is decided that the subsystem is mature enough to continue further into the life cycle. In practice lagging systems remained less mature than the other systems [9]. Additionally, the production and verification phase was neglected.

In the initial stages the Delfi-C3 was designed on paper only [9]. In the phase A the systems engineering was considered very successful, partially due to previous expertise of the team members [11]. The transition to real hardware was hard and resulted in little clear progress [11]. Systems engineering was less well applied in later phases, due to a combination of lack of manpower and underestimating the importance of continuous systems engineering by team members. Too little time was scheduled for the AIV phase [16, 45]. Test were only ultimately scheduled weeks before delivery, these problems were due to lack in maturity of the electrical and software systems, which was caused by little involvement of EEMCS students [9, 11]. During testing, developments were still going on to some subsystems [11]. It was found that it was not possible to make a perfect design on paper, produced subsystems needed testing and revisions [9, 37]. As a result parallel design of subsystems with a subsequent flawless integration did not happen and revision were necessary in the final system. In the end the project was dependent on launch delays for a timely delivery [9], even so last minute replacements and software uploads were performed [16].

**Delfi-n3Xt**

The general planning for Delfi-n3Xt consisted of short preliminary design phase, followed by two design design iterations [37, 10]. The first detailed design phase will aim to produce a
A.8 Development Phases & Life Cycle Model

Figure A.8: Representation of the \( \delta \)-sessions [7]

functional engineering model with issues. The focus during this phase would be on interface control, bread boarding and prototyping of critical system elements [10]. For non-critical parts this design phase would result in a design on paper. The second detailed design phase should result in a fully functional prototype. When there are no more remaining issues, a flight model will be produced [37, 10] for the AIV phase. It was expected that engineering models of the payloads could be available halfway the first detailed design phase [37]. It was observed that the planning was unrealistic [15]. In the preliminary design phase there was a delay due to the launch of Delfi-C\(^3\) [10]. In the first detailed design phase there was a delay of several months while these phases were supposed to take only 7 to 8 months [10]. The AIV phases had considerable delays due to the late arrival of procured subsystems, subsystems that were simply not ready and systems, both inhouse and external build, that did not function according to the specifications [16]. The project was dependent on a launch delay for completion and proper testing [16].

**DelFFi**

At the start of the DelFFi project a new design approach was chosen [38, 36]. This approach started with a baseline, the previous satellite Delfi-n3Xt, and evolved the design from there with rigorous testing. This approach is shown in Figure A.9. From the baseline a component or subsystem would be selected for improvement, after which this component or subsystem would completely be designed, manufactured and tested before integration in the subsystem or system. If the new design resulted in an improvement, this new design was to be adapted as the new baseline. Note that for every new integration into a higher hardware level required new verification. Every design cycle should be adapted to the time available to the designer. It was recognised that this development cycle had limitation, specifically that disruptive improvements are difficult, and changes to interface systems would require backward compatibility.
In practice the DelFFi design diverged from the Delfi-n3Xt satellite baseline early on for a number of reasons. As a result this design approach was not implemented. In practice rough life cycle phases were followed, starting with the establishment of an initial design. From there a detailed design was made, which gradually went over into a design and integration phase. While this was in progress, an integration, testing and validation phase was initiated. Not all subsystems had the same maturity and thus were not developed in parallel. The planning slipped continually throughout the project.

**Elstak CubeSat phasing**

Based on experience with satellite projects at Aalborg University (AAU) and TU Delft (with Delfi-C3), Elstak proposed a phasing for university class CubeSat project. This phasing is shown in Figure A.10. Note that the basic model should show that all subsystems have enough maturity.

**Conclusion**

Several life cycle models were attempted in the different projects. It is clear that parallel development and maturity of all subsystem is not reached in any of the projects. Methods to address this issue have so far been unsuccessful. Planning remained optimistic, specially the later phases were neglected in the schedule.

**A.9 Change Management**

During the development, the design of the satellite can change. This includes small adjustments, but also the removal of complete subsystems. These changes are needed to solve
problems, or deal with development issues. A decisions for which change to be made needs to be taken by taking into account the impact the change has on the development and system.

**Delfi-C³**

For Delfi-C³ an ambitious change procedure was envisioned, which is shown in Figure A.2. Two types of changes were identified: minor and major changes [7]. Minor changes could be adjusted immediately, and mostly refer to errors in descriptions and such. For major changes the causes and consequences needed to be studied, after which the changes should be discussed in a CCB, a team of relevant specialist, de systems engineer and the project manager. The CCB will make a decision on the actions to be taken. In practice changes were documented mostly in meeting minutes [8].

During the Verification and Validation (V&V) phase changes and repairs were tracked in PCB configuration files [45]. The file lists modifications, the status of the modification, non conformances and a repair log. There was no process in place or used to report changes, but the files were updated every two weeks by inquiring with team members about changes. This did cause date-fields on the forms to remain empty [45].

**Delfi-n3Xt**

It was found that most change control was done in informal meetings between team members [10]. A problem with this is that not all involved engineers can be present at these meetings [10]. Genbrugge [10] observed that successful change management should be light on documentation and not include any unnecessary communication links. Genbrugge proposed a change control approach where in a first phase the involved engineers discuss the change. If there is a change it is documented and checked by the systems engineer [10].

For changes to requirements and configuration items a special change control process was devised [10], these processes are shown in A.11a and A.11b. In both processes the initiative for change is taken by a team member. In the next step the impact of the change is researched.
together with related team members. If needed the systems engineer is informed, who will implement the change across the systems documentation.

**DelFFi**

There was no official change management implemented in DelFFi. In practice design changes were taken by, or in discussion with, the project manager and systems engineer (if available). These changes were communicated in the team meetings. It was observed that these changes could surprise the people affected by the change.

**Conclusions**

Change management seems naturally to occur in meetings between experts. Minor changes can be made without major interference, while major changes need more rigour. The changes should be communicated and documented well. The systems engineer and project manager can be included in the discussions if needed. The rest of the team should be made aware of the change.

**A.10 Interface Management**

**Delfi-C**

In the initial phases of the Delfi-C project extensive interface control were developed [14]. Every interface between configuration items was defined in an Interface Control Document (ICD) [9]. The project also tried to simplify the interfaces by introducing an interface board [9]. The interface control procedures were neglected once deadlines increased the pressure to make progress [14].

Also an initial study using an N2-chart was done [8]. For a more formal interface control it was attempted to make a database, but it was too much work to create a user friendly version and perform good version tracking with it [8]. It was suggested a wiki-based interface control website might offer a solution, however this was never implemented [8].

**Delfi-n3Xt**

Interface management and control became an important feature on Delfi-n3Xt. Initially the satellite would feature 5 external payloads.

An observed problem during the interface control was that contingencies and margins were taken into account by different parties [15]. It was concluded that the contingencies and margins were to be taken into account at a satellite level, in this case by the structure subsystems engineer [15].

A interface design procedure was designed which relied on ICDs, which would be changed by a change proposal form [15]. The ICDs had a standard layout and table of contents, such that all possible interfaces would be considered [15]. Control of the status of all ICDs would
Figure A.11: Change control in Delfi-n3Xt [10]
be done excel base interface tracking tool, and comments on an ICD would be collected in an excel based interface sign-off matrix[15]. The ICDs would first be base lined in the interface definition phase, after which the ICDs would only be controlled. The interface definition phase was initially suspected to be complete for the PDR, however in practice it seemed more realistic to aim for the Mid-term Design Review (MDR) [15]. An ICD would only be published if all involved approved of the new document [15], according to Lebbink [15] this practice slowed the release of ICDs. Lebbink proposed to publish conditional ICDs next to the officially published ICDs. While all subsystem engineers were supposed to sign off an ICD, in practice this responsibility was given to a single team member [15]. Subsequently, some team members felt like their input was not taken into account [15]. A small questionnaire was held on the satisfaction among team members and payload partners. The results showed ambivalence towards the procedure [15], with only the contact between Delfi-n3Xt team and payload partners being rated as satisfactory [15]. A personal comment described the interface control procedure as bloated, which was also echoed in a questionnaire with new team members [15]. The questionnaire was performed in the initial phases of the design, which could have contributed to the ambivalent response [15]. During the application of the interface management, still non-conformities were found [15]. A majority of these non-conformities were due to outdated data in the documentation [15]. It should be noted that this means that the same data was duplicated in numerous documents. The solution proposed by Lebbink for this was to maintain the ICD as only true source of information [15]. Lebbink recommended to improve the involvement of the project participants and increase the availability of information [15].

It was considered to make a generic ICD for all payloads and record deviations in specific ICDs [15]. In practice it turned out that a majority of the ICD entries were payload specific [15].

**DelFFi**

The interface control in the DelFFi project was different than the previous projects. Since the satellites would be accommodating centrally designed mission payloads and existing COTS subsystems, the design team had to take into account the existing interfaces. The payload requirements and interfaces were communicated with ICDs.

For the Propulsion system meetings were held between involved engineers to agree on the interfaces between the propulsion module and the satellite.

**Conclusions**

It is important that it is clear what the partners are designing and what their requirements are, to avoid double work, such as was the case with contingencies in Delfi-n3Xt. The interfaces have been documented in most project using ICDs, which was successful, however the related processes should not become bloated. Meetings should be held regularly to ensure that all partners are aware of the status of the satellite and payloads.
A.11 Integration, Verification and Validation

The time available for testing has been identified as a major contributor to the system reliability [36].

Delfi-C³

The intended manufacturing process is represented in Figure A.12. Note that part of the testing was supposed to start during this process already. Manufacturing of parts and components was not necessarily done in a cleanroom environment, but parts did need to be cleaned for assembly.

The integration process in practise was hectic, with no planning for integration, the clean room offered too little space [45]. Actual PCBs only started to arrive less than half a year prior to the first planned delivery [9].

The intended verification approach is shown in Figure A.13. In order to save costs, it was decided to perform tests at the lowest possible level, and limit system tests [7]. The proto-flight model would be subject to acceptance testing. As noted previously, the proto-flight approach did not succeed. The integration and verification plans were collected in a single document [45]. Verification plans for subsystems were available, though not complete due to time pressure [45]. Design documentation also supported the verification effort [45].

Testing of subsystems was set up in 4 categories: Mechanical testing, Electrical testing, functional testing and environmental testing [45]. The mechanical testing is known as inspection in literature [19, 56], and checked geometry, centre of gravity mass and visual check for defects. The electrical tests checked circuit design, component failures and PCB integration failures.
the board is first inspected un-programmed and subsequently programmed. In this inspection it is checked whether the correct voltages are present in the system. The functional testing were focussed on the performance of the software in the on-board environment. The flight software responded to test scripts. These scripts were first made for the dedicated software test and adapted for the flight software. The environment tests placed the subsystems in a vacuum oven at TU Delft. The boards were inspected before, during and after the test conditions were reached. The complete system also underwent thermal vacuum cycling test and a vibration test [45].

The test were documented in logbooks. Every subsystem and every test engineer had its own logbook. Subsystem handling and programming logs were initiated, but abandoned [45]. Non-conformance records were hardly ever used [45]. Changes to PCBs were tracked in PCB configuration files. These were updated regularly by actively asking around, instead of using an official change management process [45].

The status of the flight (and spare) stack was being tracked in a status document, which included the modification date, the stack placement, an identifier of every subsystem and the status of every subsystem [45].

The V&V took more time than initially estimated [16], due to shifting shipments and a decreasing team size [16]. The increasing time pressure decrease the quality of the work done [45].
Delfi-n3Xt

The initial approach to testing was divided in two distinct phases[16]. In the first phase the subsystem engineering models would be verified[16]. The second phase would verify the flight model[16]. This strict division was not kept due to design changes between engineering and flight models. Different types of test were identified and ordered by priority, from highest to lowest priority[16]:

1. Interface tests
   To verify the interfaces between the subsystems. These tests were given highest priority due to experiences with Delfi-C3.

2. Functionality tests
   To verify the compliance of the subsystem with their functional requirements.

3. Performance tests
   To characterise the subsystem. These were performed if time permitted.

It was required that every test was documented in a test plan and a test result document[16]. The test plan contained the requirements, test goals, equipment involved, an outline of the setup of the test and a step-by-step procedure of the plan[16]. The last item was included to prevent damage to the systems and other undesired events.

At one point another approach to testing was taken due to a smaller team and time pressure[16]. The extensive documentation of tests were abandoned, and to guard the quality, at least two persons were required to perform the tests, with a third unrelated person present to make sure no crucial mistakes occurred during the test[16]. During this time more mistakes occurred, resulting in damage on hardware and the need to repeat tests, and when the schedule relaxed the initial approach returned[16]. The test setup was still checked by another unrelated team member, since it was suspected that this reduced errors[16].

In the final phase of testing interface test were performed continuously, since the last subsystem was not finalised, and still changed in such a way that the interfaces needed verification[16]. These changes should be avoided at all costs[16]. It can be noted that Delfi-n3Xt encountered the same issues as Delfi-C3; shifting delivery dates and a decreasing size of the test team.

The project suffered several delays when ordered systems arrived late. Not all external partners were able to deliver the agreed upon models either[16]. Based on the experience with V&V in Delfi-n3Xt, Boerci recommends to start stack testing early, and incorporate possible delays of subsystems into the planning[16]. When a subsystem faces considerable delays, he recommends to revisit the planning and simplify the functionality of the system in question[16]. Clear delivery dates and intensive contacts with project partners would help as well. Another recommendation was to agree on the delivery of an interface and functional model of the subsystem early in the project to initiate the team with the verification activities[16].

Inexperience within the team resulted in incorrect planning of the testing and mistakes in handling of the hardware[16]. Boerci[16] recommends that more time is invested in the preparation of the V&V phase and experienced team members and staff are consulted often. The preparation of the V&V phase should also start earlier in the project. He also notes that the availability of base versions of the software early on, will give the test team a better view of the functionality, and the software engineers more time to complete their work.
DelFFi

During this research the integration had not started yet. Only some subsystems were produced by team members. Procurement of parts was sometimes hindered by delayed financing on the part of TU Delft, this caused delays in delivery. Currently another subsystem is facing delays due to production delays at the side of a partner.

At the moment of this research no considerable V&V effort had occurred within the project. A staff member was present to supervise the V&V effort.

Conclusions

All projects have seen delays in the delivery of components and subsystems. There are various reasons for this. These delays caused further delays in the project and gave limited possibility to completely verify and characterise the designs.

The V&V should be organised according to a verification plan. Both Delfi-C³ and Delfi-n3Xt encountered delays and a decreasing team size in this phase. At moments of time pressure careful documentation and planning was abandoned, which resulted in a lower quality of the work and errors being made. The design work was not necessarily done when verification started. Verification activities can be separated based on what is being verified.

A.12 Risk & Reliability Management

Delfi-C³

In order to establish a risk management approach for Delfi-C³, four risk categories were identified [7], and addressed in some way:

- Technical risk:
  Part of this risk was addressed by the Failure Mode, Effect and Criticality Analysis (FMECA) and the one Single Point Failure (SPF) survivability philosophy. Additionally items that had a high risk of being delayed during design were identified.
- Cost Risk:
  A cost breakdown was produced and a low-cost philosophy was implemented. This risk was addressed by staff involved in the project.
- Schedule Risk:
  Items with a long lead time were collected in a Long Lead Item List. The risk of student not joining subsequently was not addressed, "as it is not within the control of both the Project Manager and the Project Leader." [7]. This seems to be a particular bad way to deal with this risk.
- Programmatic Risk:
  This was not clearly addressed.

To handle risk it a top level FMECA was planned to be performed. Additionally a one SPF survivability philosophy was adapted, which means that when one SPF occurs, the primary
mission objective should still be reachable [7]. It was also observed that making a complete SPF free design was practically impossible [36], specifically for the electrical interfaces. However, by combining redundancy with error detection and correction a failure tolerant design could be achieved [36].

The failures within the project were categorised by severity [36]:

- Catastrophic failures, resulting in a loss of the system.
- Critical failures, which result in a failure to meet the main mission objectives.
- Major failures, which degrade mission performance.
- Minor failure, which have no lasting impact on mission performance.

**Delfi-n3Xt**

A qualitative approach to reliability was taken, since a quantitative approach was not practical due to missing reliability information on COTS components and limited test time. [12]. A number of approaches were used to increase reliability:

- SPF should be avoided, the design was aimed to be near SPF free. Failure mode analysis was used to identify SPFs [12].
- Critical subsystems and components were made redundant [12].
- Critical interfaces were designed to be failure tolerant [12].
- Critical functionalities were to be tested, including the simulation of failure cases [12].
- Time critical failure cases should be solved by the system (On-Board Computer (OBC)) itself. Non time critical failure cases were left to be solved by ground control to avoid complexity in the design. The latter implies that all subsystems should generate housekeeping data for ground control to inspect. [12].
- Lessons learned from Delfi-C³ were to be implemented, specifically with respect to Inter-Integrated Circuit (I²C) communications and data handling at the ground segment.

For Delfi-n3Xt it was stressed that testing was the best way to design a reliable system [12, 36]. The total approach to reliability was to implement simple mitigation, and ignore low likelihood failures with complex mitigation solutions [12]. Note that these approaches related to technical risk.

**DelFFi**

The reliability approach taken for the DelFFi satellite was the same as the approach taken in the Delfi-n3Xt project [38]. In practise a limited top level risk analysis was done prior to the PDR and the Critical Design Review (CDR), this also included the identification of programmatic risks. Prior to the verification a more in depth analysis was started.

**Conclusion**

The reliability approach taken was mostly implemented towards technical risks. SPF was to be avoided in all projects. For Delfi-n3Xt and DelFFi the same reliability approach was taken. It was noted that commercial grade components failure rates were a minor issue compared to overall system reliability and reliability issues due to planning and resource issues [36].

Master of Science Thesis  N. van der Pas
A.13 Team Structure & Characteristics

Delfi-C3

The organisation of Delfi-C3 was defined as shown in Figure A.14 at the initial stages of the project [7]. The team was intended to consist of students [8]. The advisory groups advised the teams on particular parts of the design. These advisory groups also review the design, which was considered usual for university projects according to Bonnema [7]. In contrast the graduation supervisors guided the students during the design and their thesis.

The hierarchy of the students in the team is essentially flat [45]. This can cause problems in the implementation of the systems engineering processes [45]. It was found that BEng students need a supervisor by an expert staff member and support by a systems engineering team member.

The team was located at the faculty of AE and the faculty of EEMCS, with respectively 15 and 8 student work spaces[50]. Team size varied between 10 members at the start and end of the project and 25 members during the development phase, according to Vaartjes [50]. Vaartjes adds that in total 60 students worked on the project, of which 15 were MSc graduation students. Elstak counted over 30 MSc students involved and over 35 BEng students involved [9].

It was attempted to include MSc course students in the project by setting up assignments [8]. This was unsuccessful; students needed to learn the course, the satellite mission and the tools involved, which proofed to be too much [8]. For a number of work packages that were very independent from the configuration of the satellite, assignments were successfully implemented [8].

Staff members and industry partners did actively take part in the design next to their advisory role [9, 8]. In practice few students of the faculty of EEMCS joined the team, possibly because of the gap between project needs and EEMCS thesis requirements [11]. AE students instead
did electrical design for the project [9]. During the design both SystematIC BV and TNO stepped in to fill work packages which were not or not satisfactory filled by students [8]. To finish the satellite ultimately team members were hired [9].

An good team spirit boosted the work quality and output [11]. It was concluded that a strong staff backbone was needed to supervise the project team members [11, 45] and to fill gaps in the team if necessary [11].

**Delfi-n3Xt**

The team hierarchy intended for Delfi-n3Xt is shown in Figure A.15. The general staff members offer mostly support, while the project manager and students form the core team [34]. The system and subsystems engineers are master students. The hierarchy shown here is more a hierarchy of influence, in terms of power it was a flat organisation [34].

The team size decreased near the end of the project [16]. In order to dampen the effect, a number of recent graduates were hired [16]. Boerci advises careful planning which incorporates student and project needs to maintain the continuity of the project [16].

Student motivation was dependent on the reason they participated in the project [16]. The students involved have very little experience with testing and often underestimate the planning and time needed for testing [16]. This inexperience also resulted in incorrect handling of hardware.

**DelFFi**

With respect to the previous Delfi projects, staff members would be more heavily involved [77]. The system would be designed by students and staff members would carry the responsibility for the subsystems [38]. It was hoped this would lower managerial risks.

The team consisted of MSc literature study, MSc thesis, BEng internship and BEng thesis students. MSc students from AE were the majority of the team. A number of staff members were involved, most in a supervisory role. Another staff member was hired for the design of a number of the electronics.
The team structure was based on a work package breakdown. The hierarchy is shown in Figure A.16. In practice students felt that their concerns were not taken into account and decisions were made without their input. The students felt like staff and students were held to a different standard, in terms of deliverables. The students were involved in the project and team up to a different degree depending on their thesis assignment and personality.

Conclusions

Students form the core part of the team, supervised by staff. The project is managed by a staff member. AE students are best represented, there is a shortage of team members from EEMCS. Alumni and staff from external partners have been hired in the later stages of the project to finish the project.

Student have little experience, staff is needed to supervise the students. the project hierarchy in the team is essentially flat, though staff members can have the upper hand in decisions. Students are motivated to a degree depending on team spirit and the reason they are involved in the project.

A.14 Meetings

Delfi-C³

The team met regularly in team meetings at least every week [8]. These meetings were focussed on keeping the team on the same track and inform all team members of design progress. Every two weeks a meeting with supervisors and advisers was held, to keep those up to date. For specific design issues or part team members set up meetings themselves with involved members, advisers and supervisors.

There were also progress meetings with external parties to discuss design maturity of the systems [8].
Delfi-n3Xt

The progress meetings changed format in March 2009, team members were required to prepare announcements and important discussion points. Additionally, every progress meeting one team member would prepare a 20 minute presentation, so that all team members would learn more about the system [79]. It was observed by a student that the project meetings lacked interaction [46].

Regular progress meetings to keep members up to date with the status of the projects were seen as essential [16].

DelFFi

There were different types of meetings in DelFFi. The progress meetings were regularly scheduled meetings with the complete team. Beside that there were ad-hoc experts meetings on several topics, these meetings were organised by the involved team members whenever the need arose. The staff involved in the project also met, these meetings were held behind closed doors, and the rest of the team members did not consider these meetings to be transparent [47].

Progress Meetings

The progress meetings occurred regularly from December 2013 onwards. The most prominent feature of these meetings was the round table, were every team member reported on their progress and problems. Sometimes the meetings were started with a specific subject. The meetings were primarily informative, most decisions were postponed to dedicated subsystem meetings.

In October 2014 the format of the meetings were changed, in that the student team members gave a short presentations with a few slides in a standard format on what they had done since the last meeting and were planning to do. While there was initial hesitation to this change, it was in general experienced as improving the meetings by the people involved. Even so, team members still felt the team meetings to be ineffective. A better preparation for meetings was expected and making use of action items was expected to improve the meetings [47].

System Review

On the 11th of December 2014 an internal system review was performed. This consisted of an extended of all people involved and other department members. The status of all subsystems was presented, as well as schedule conformance, technical issues, interface issues and none assigned work. The review was well received, members found it useful to get an overview of the complete system status. It was suggested to repeat this sort of system review on a regular basis.

Master of Science Thesis N. van der Pas
Conclusions

All projects have used regularly meetings to keep team members up to date with the progress of the project. General team meetings are mostly informative and sometimes lack interaction. Decisions are often made in dedicated meetings between subsystem engineers.

A.15 Documentation

Overall the Delfi projects have taken a document based systems engineering approach, thus most design information was stored in (digital) documents. Documentation quality was identified as the most important factor in information transfer [36]. This section will look at the documentation structure and structure of the documents.

Delfi-C

Documents were stored on two locations, both the TU Delft server and an internet forum [7, 9]. The latter was implemented to make documents available for people located outside of the TU Delft. To team members that followed the initial team, the documentation structure was not clear [45]. Every document was also added to the single list of references [8].

The overall design data was expected to be stored in review reports. Additionally members were expected to produce technical notes on smaller parts of the project [7]. It was expected that thesis reports could be used for a member to report on the work they did, and as such make sure no double work was done. Additionally the following templates were produced to support the documentation:

- Technical notes
- Document approval sheets
- Document change record
- Document input sheet
- Comment input sheet
- Design data control sheet
- Standard list of references
- List of acronyms and units

All above mentioned sheets and the document change record were supposed to be appended to other documents to control the data and changes. Additionally it was proposed to keep logbooks and set up lessons learned documents [8], these never seem to have been implemented.

The documents were issued after they were checked by the systems engineer and were supposed to be approved after review by the project manager. In practice many documents were not approved [8]. Technical notes were used by team members as close out documents, similar to the theses [8].

Genbrugge et al. [34] observed that the focus of the systems engineering shifted from design control to documentation control. There was a wide array of documentation types and control
sheets. New members often generated new documents, without updating the documents of their predecessors [34, 9]. In the end information was duplicated and documents were scattered [34]. The complex control structures did not have the support of the team, and were thus not effective [34]. In the later phases of the project, documents from the early phases were not found to be useful by the team members [9]. Later team members experienced difficulties adopting the systems engineering setup and documentation structure [45]. Data input sheets and design data control sheets were rarely used [8, 45]. For technical documents such as test plans and design drawings, version control was better [8]. Elstak [9] advised to use living documentation.

Additionally in a number of phases it was assumed only a minimal approach could be taken to documentation [8]. It was reasoned that since team members cooperate closely, a concurrent approach could be taken to the design, even in the later phases [8]. As long as the team members were still involved in the design this did not give problems [8].

During the AIV phase less documentation was done [45], it is noted by Vaartjes [45] that this was done in a response to time pressure. The lack of documentation stopped the information flow and made information less accessible [45]. As a result, simple tasks took more time than needed.

Due to the switch from a proto-flight approach to a prototype approach during the AIV phase, the time pressure on the project increased [45]. This time pressure let to a decrease in quality of activities, which in turn increased problems [45]. Due to the time pressure new team members did not get a chance to properly familiarise themselves with the project.

There was a discontinuity in the workforce. As a result time pressure increased and members were not properly introduced to the systems engineering approach [45, 50]. Vaartjes [45] concluded that in order to minimise the effects of workforce discontinuity, documentation should be up to date, information should be easily accessible and the systems engineering activities should be actively taught. Additionally Vaartjes [45] advised for an overlap period between a predecessor and a successor.

**Golden Notes**

Early on in the project it was recognised that new team members need to be introduced quickly into the project by informing them about the systems engineering, project management and other processes in the project [7]. For this the 'golden notes' were introduced. The golden notes were to include:

- a summary of the project’s history,
- a summary of the project’s objectives,
- a summary of all management plans,
- practical information on the daily routines,
- practical information on workspace and facilities
- practical information on rules of conduct within the team and
- an overview of team members and their thesis topic.
Later additionally budgets and thermal control files were added to the golden notes [8]. The golden notes were abandoned somewhere in the project, because they needed continues maintenance\(^2\).

**Delfi-n3Xt**

During Delfi-n3Xt the documentation was collected in a central project disk [48]. The disk has a strict structure and a approval of the project manager was needed to upload documents to certain folders to ensure the quality of the documents [34], only version documents with version numbers higher than 1.0 needed review though [48]. It was encouraged to update existing documentation and early release of documents [48]. Every document was given a unique code. In order to keep information unique, only a few types of documentation were allowed. Also a living documentation approach was adapted [34], to ensure that the information on the disk was up to date. Every document was accompanied by a single document control sheet [34], which kept basic information about the version, authors and a change log. Observations of the documents show that these sheets were filled in, in contrast to a number of the sheets of Delfi-C\(^3\), which often were left empty.

Every member was responsible for the document about his or her subsystem. Responsibility transferred to a successor when a member left the team.

During operations part of the documentation was also stored in the form of emails. Small pass reports were emailed to a central email server and are stored there.

It was recommended to spend more time on documentation [16]. When time pressure rose, extensive documentation was abandoned [16]. While this task has normally low priority, in the end it will lead to a more efficient methodology and allow shorter project introduction periods [16]. In the end the documentation for Delfi-n3Xt was incomplete, leading to complications when DelFFi attempted to reuse the design.

**Hardware Logs**

The state of the hardware was documented in hardware logs. These logs contained information about what had been done to the hardware item, what problems were encountered and whether these problems were resolved. The logs also noted which software, including version, is installed on the hardware.

**DelFFi**

During the DelFFi project initially the same document storage structure and naming convention as in Delfi-n3Xt was adopted [49]. It was envisioned to use new tools and methods to minimize the documentation [49], however, it was not detailed what tools and methods these were supposed to be. There were no templates available for standard documentation. The intention was to check documents intensively by work package leaders and supervisors[49]. Documentation would be a prerequisite for the approval of technical work [49]. However

\(^2\)From conversations with Jasper Bouwmeester

N. van der Pas

Master of Science Thesis
documents were rarely approved and in practice the document inbox became the standard place for documentation. Some members abandoned the documentation structure completely. Team members noted that this could cause interface issues and new team members had a hard time familiarizing themselves with the project. The official documentation that was available was often from or for external partners. The Delfi email box was used by some members, though others got access to this information source very late.

It was observed that for a lot of information people directly talk to others, a much heard question within the team was "do you know this, or do you know who knows this?". This indicates that much of the information was based with the team members, and not in generally accessible documentation. There were no documents available for a quick introduction to the project, with the result that new team members spent a significant amount of time familiarising themselves with the project [47].

It was also observed that documents were not made available to the team. In certain situation team members were asked to keep documents secret from other team members. What purpose this served is unclear. Information also did not always flow down from the staff members to the students, this included decisions and technical parameters. It seemed that this happened both with and without intent. Information that the student team members produced was scarcely released as well. Most information was released in the final thesis, but this means the information can take up to a year to be available.

**Conclusions**

In Delfi-C\(^3\) it was seen that extensive document control was used, this resulted in the document control rarely being used. The documentation was kept at different location, which made it difficult for new members to find what they needed. It also led to duplicate documentation. Team members within Delfi-C\(^3\) started new documentation for their own work, leaving previous documents not updated. In the Delfi-n3Xt project these issues were addressed, documentation was kept at a single documentation disk with a rigid structure and documentation control sheets were simplified. Documents were also transferred to successors. This had success. The DelFFi project attempted to use the same approach, however due to time pressure, the document manager did not find time to update the documentation. This lead to team members using only preliminary documentation or abandoning the documentation structure completely.

Time pressure forms a danger to documentation, which is often the first activity to be abandoned. However, experience in all projects shows that this has a negative impact on the quality of the design. It can be seen that without documentation mistakes are made, new members will need to do more work, work cannot be reviewed, and designs cannot be replicated. It is shows that team members need a proper introduction into the documentation and enough time need to be reserved to produce documentation.

The experiences within the DelFFi team show that clear documentation does not automatically arise when a team cooperates together. the structure and documentation quality needs to be controlled.
A.16 Systems Engineering in Practice

This sections collects the remaining observations made no the projects. Issues that could not be captures within the above mentioned sections will be collected here, to get a complete view of the projects. In all the projects the transfer of a task did not go smoothly. In half of the cases, there was no overlap between a team member and a successor [36]. As a result the knowledge transfer takes up a significant amount of the students time [16].

Delfi-C3

While the theory behind the systems engineering and project management was described in detail for Delfi-C3, in practice the application had problems.

There were two general breakdowns used in the project, a configuration and a hardware breakdown. The requirements were linked to the configuration breakdown, however the hardware breakdown was used more by the team, making the list of requirements less than useful [34]. The design maturity was based on these configuration items as well. Overall team members had a bad knowledge of the project status [9], Elstak recommended weekly meeting or status mailings to improve this[9].

The students participated in the project as part of their graduation project. The student is sometimes faced with deciding whether he or she gives the project or thesis priority [50, 11], in these situations a thesis supervisor with an interest in the project will not be able to give unbiased advise to the student [50]. Vaartjes [50] advises that the thesis supervisor should be independent of the project, the thesis assignment should be well defined and should contain alternatives in case of delays in the project. In other cases the student might not have the right capabilities and knowledge to successfully perform his or her work package [8, 11]. For the project managers it is important to know both what work is needed for the project and what work is allowed within a thesis or other educational project [8]. Bouwmeester [11] advised that a proper human resource approach needs to be implemented to safeguard the needs of both the student and project.

During the development early assumptions turned out to be incorrect. While this impacted the design and there were no resources and time available to resolve these problems, the system-level failure tolerance approach proved successful and mitigated any impact on the overall mission [36].

The planning for Delfi-C3 encountered a number for delays. during the AIV phase this led to waves in activities; a lot of activity right before a deadline, a low in activity when news of a delay was announced [45]. Vaartjes [45] notes that part of the issue was the communication towards the team, an immediate and equal level communication with the students would, in combination with weekly targets, have mitigated these work intensity waves.

Delfi-n3Xt

It was noted that unrealistic planning and delay prognoses inhibited the design work [15]. Not at all times the mass and volume budgets were performed properly, which was likely caused by a distribution of these two related volumes over two system engineers [15]. It was noticed
that if things were seemingly going well, contact between payload partners and team where watering down, in order to keep up to date with the design status and identify problems on time it was suggested to agree on a set meeting structure with payload partners [15].

It was observed that students could spend too much time on non-thesis related tasks, while important for the project, student could face that work done was not of the required level for a thesis [46]. The project was lacking manpower [46].

Breadboarding interfaces early proved a successful strategy [36], but for other newer and more complex systems the same errors were made in Delfi-n3Xt as in Delfi-C3 [36]. The systems were prototyped late, with errors showing up when there was no time any more to solve them [36].

Right before the integration phase of Delfi-n3Xt, a tasks management approach was implemented. These tasks were implemented using Microsoft Outlook. The task were distributed and stored on the Delfi mail account. This can be seen as an implication of the weekly targets proposed by Vaartjes. This approach worked, but was not used any more when only a small core team was left, since the overview of work was easily kept without the tool at that point 3. The tool was not reinstated in a new project.

**DelFFi**

It was aimed to base the DelFFi satellites on the Delfi-n3Xt satellite. As such a life cycle based on iterative improvements of the Delfi-n3Xt baseline was proposed. However the DelFFi baseline design started to divert from the Delfi-n3Xt baseline quickly. There were a number of reasons for that. For starters, the DelFFi satellites would operate in a different environment than the Delfi-n3Xt satellite. Not all systems of the Delfi-n3Xt satellite performed up to standards in orbit. And the documentation for the systems of Delfi-n3Xt was substandard or not up to date, and as such, it was found easier to start a new design, than to try to reverse engineer the Delfi-n3Xt systems and software. It should be noted that Delfi-n3Xt also aimed to reuse systems from Delfi-C3, but this did not work since many issues encountered in the later phases of Delfi-C3 were not documented [36].

Systems engineering was not applied visibly to the team [47]. In practice team members were missing access to essential documents. Reviews and requirement control did not seem to be performed. As a consequence, the team was uninformed about the status of the design. Additionally, the planning was unrealistic, and thus mostly ignored in the design processes.

The team size was changing over the course of the project. Some new team members left the project after being disappointed about the possibilities within the project and fears for extra delays in their own educational plans. The students within the team advised to set up thesis assignments in advance to attract students to the project [47].

The students of the team members did not feel involved in the design process. They felt like decisions were taken without their input [47]. The student requested more communication towards them, even if decisions were not taken yet [47]. Students performed tasks that they felt were not necessarily beneficial for their thesis and thesis assignments often changed during the thesis.

---

3According to a conversation with Jasper Bouwmeester
Student project or University satellite?

It was observed that the staff saw the DelFFi project as an educational student project. On the other hand, students indicated to feel as if they were simply performing work for the University, in other words in their eyes it seemed like a university project.

It was observed that most major decisions were taken by the staff, and that the thesis assignments of the students were written such that the project was not a central element in it. This leads in effect to the strange situation that the needs of both staff and students (research and graduation) were not satisfied through the project.

Conclusions

All project experience difficulties combining theses with the project. Tasks done were not of the required rigour and quality required for a thesis. As mentioned in Section A.13, this partly explains a lack of electronics and programming students in the project. It was recommended to have a project related supervisor and an external project thesis supervisor, this never seems to have been implemented. It is suspected that having two active thesis supervisors is not realistic within the University context. A proper approach for this has to be devised.

Planning is another aspect often mentioned in this section. Both for Delfi-n3Xt and DelFFi the planning was considered unrealistic. This had a negative impact on the design work. New systems often had delays, adding delays to an already tight schedule. In DelFFi, still in its early phases, this led to the planning being mostly discarded by team members. In all projects trying to meet deadlines resulted in incomplete or missing documentation (see also A.15), resulting in more delays later on in the project and design errors. Setting short terms tasks seem to have a positive impact on this issue.

Both in Delfi-C^3 and DelFFi it was observed that communication towards the student team members can be improved. The design effort did advance slow when kept in the dark on decisions, and afterwards the students had to scramble to update their designs. This relates to the question whether the project is a student or university project. For a student project, the students need to be involved in all aspects of the design, for a university project the students require clear and unambiguous assignments.

A.17 Conclusions

The objectives of the Delfi satellites are divided in three categories; educational, technical and programme objectives. The technical objectives are project dependent, and aim to test new technologies. The educational aim to give students hands on experience in teamwork and with satellite development and thus represents the student needs. In practice this translates to facilitating thesis work within the project. The programme objectives aim to mature the nano-satellite platform, and are thus related to engineering research. The technical objectives got more attention, and the programme objectives less. In the latest project DelFFi also a scientific objective is set, which could indicate a maturing platform, but could also only be motivated by the partnership with QB50.
The Delfi satellites were engineered at the TU Delft, with the chair of SSE involved as the
project lead. Other faculties were involved to provide specific expertise. All Delfi satellites
carried payloads for other partners. Some testing was performed at external institutes. Solar
cells and the launch were brokered through external partners for all missions. Throughout the
different satellites, more subsystems are procured. The DelFFi mission is an exceptional case
in that it has one partner (QB50/VKI) which provides payloads, launch and requirements.

What is notably missing, is the involvement of staff as researchers. The satellites did not
carry a research payload from University staff or PhDs nor was their research connected to
the satellites, with the exception of the propulsion system on DelFFi.

While an systems engineering philosophy could give direction to project, really only for
Delfi-C3 the approach to the engineering of the satellite was explicitly stated. For the KISS
approach it was concluded that it could not successfully be applied to all subsystems, some-
times increasing the complexity in the end.

All three projects used configuration items to make a breakdown of the project. The imple-
mentation differed along the projects. In Delfi-C3 the configuration item was used to manage
the engineering and control the status of the design. Formalised processes were used for this
control. In Delfi-n3Xt the configuration items were related to the requirements, and were
assigned to a single responsible person. In DelFFi it was attempted to related configuration
items to work packages.

The implementation of configuration items seems to have a positive impact on the engineering
process, but the exact form and impact they should have is debatable. It is clear that the
configuration items give a breakdown of the structure which can be related to work to be
done and can be used to connect requirements, hardware and work packages.

Both for Delfi-C3 as Delfi-n3Xt an ambitious approach was taken to requirements, only to
later be replaced by a simpler approach. It is suspected that for Delfi-n3Xt the approach
failed because of the cumbersome requirement change process, which put the requirement
control strictly with a single person and added additional tasks to the other team members.
For DelFFi the requirement management was also strictly with a single person. It can be
concluded the requirement process is important, and requirements are desired by the team
members, however, the requirement process should not pose additional tasks on the team
members, and should not be dependent on the inputs from a single person. Remarkably none
of the projects related the requirements to a verification approach.

Very different model philosophies have been applied to the different projects, none were
successful. The team is to inexperienced for a proto-flight approach as shown during Delfi-C3,
, a double prototype approach as in Delfi-n3Xt is very sensitive to delays in the project and
the evolutionary approach of DelFFi never really took of because a baseline model was not
fully established and not robust enough to the difference in mission. The model philosophy
has to take into account the resources available, the expertise of the team, the sensitivity to
delays and its dependence on heritage.

Several life cycle models were attempted in the different projects. It is clear that parallel
development and maturity of all subsystem is not reached in any of the projects. Methods to
address this issue have so far been unsuccessful. Planning remained optimistic, specially the
later phases were neglected in the schedule.
Change management seems naturally to occur in meetings between experts. Minor changes can be made without major interference, while major changes need more rigour. The changes should be communicated and documented well. The systems engineer and project manager can be included in the discussions if needed. The rest of the team should be made aware of the change.

It is important that it is clear what the partners are designing and what their requirements are, to avoid double work, such as was the case with contingencies in Delfi-n3Xt. The interfaces have been documented in most project using ICDs, which was successful, however the related processes should not become bloated. Meetings should be held regularly to ensure that all partners are aware of the status of the satellite and payloads.

All projects have seen delays in the delivery of components and subsystems. There are various reasons for this. These delays caused further delays in the project and gave limited possibility to completely verify and characterise the designs.

The V&V should be organised according to a verification plan. Both Delfi-C$^3$ and Delfi-n3Xt encountered delays and a decreasing team size in this phase. At moments of time pressure careful documentation and planning was abandoned, which resulted in a lower quality of the work and errors being made. The design work was not necessarily done when verification started. Verification activities can be separated based on what is being verified.

The reliability approach taken was mostly implemented towards technical risks. SPF was to be avoided in all projects. For Delfi-n3Xt and DelFFi the same reliability approach was taken. It was noted that commercial grade components failure rates were a minor issue compared to overall system reliability and reliability issues due to planning and resource issues [36].

Students form the core part of the team, supervised by staff. The project is managed by a staff member. AE students are best represented, there is a shortage of team members from EEMCS. Alumni and staff from external partners have been hired in the later stages of the project to finish the project either because their expertise was needed or because student have little experience, staff is needed to supervise the students. the project hierarchy in the team is essentially flat, though staff members can have the upper hand in decisions. Students are motivated to a degree depending on team spirit and the reason they are involved in the project.

All projects have used regularly meetings to keep team members up to date with the progress of the project. General team meetings are mostly informative and sometimes lack interaction. Decisions are often made in dedicated meetings between subsystem engineers.

In Delfi-C$^3$ an extensive document control structure was envisioned, this resulted in the document control rarely being used. The documentation was kept at different location, which made it difficult for new members to find what they needed. It also led to duplicate documentation. Team members within Delfi-C$^3$ started new documentation for their own work, leaving previous documents not updated. In the Delfi-n3Xt project these issues were addressed, documentation was kept at a single documentation disk with a rigid structure and documentation control sheets were simplified. Documents were also transferred to successors. This had success. The DelFFi project attempted to use the same approach, however due to time pressure, the document manager did not find time to update the documentation. This lead to team members using only preliminary documentation or abandoning the documentation structure completely.
Time pressure forms a danger to documentation, which is often the first activity to be abandoned. However, experience in all projects shows that this has a negative impact on the quality of the design. It can be seen that without documentation mistakes are made, new members will need to do more work to familiarize themselves with the project, work cannot be reviewed, and designs cannot be replicated. It is shows that team members need a proper introduction into the documentation and enough time need to be reserved to produce documentation.

The experiences within the DelFFi team show that clear documentation does not automatically arise when a team cooperates together. the structure and documentation quality needs to be controlled.

All project experience difficulties combining theses with the project. Tasks done were not of the required rigour and quality required for a thesis. As mentioned in Section A.13, this partly explains a lack of electronics and programming students in the project. It was recommended to have a project related supervisor and an external project thesis supervisor, this never seems to have been implemented. It is suspected that having two active thesis supervisors is not realistic within the University context. A proper approach for this has to be found.

Planning is another aspect often mentioned in this section. Both for Delfi-n3Xt and DelFFi the planning was considered unrealistic. This had a negative impact on the design work. New systems often had delays, adding delays to an already tight schedule. In DelFFi, still in its early phases, this led to the planning being mostly discarded by team members. In all projects trying to meet deadlines resulted in incomplete or missing documentation (see also A.15), resulting in more delays later on in the project and design errors. Setting short terms tasks seem to have a positive impact on this issue.

Both in Delfi-C$^3$ and DelFFi it was observed that communication towards the student team members can be improved. The design effort did advance slow when kept in the dark on decisions, and afterwards the students had to scramble to update their designs. This relates to the question whether the project is a student or university project. For a student project, the students need to be involved in all aspects of the design, for a university project the students require clear and unambiguous assignments.
Appendix B

Delfi Team Research

B.1 Data Collected

A total of 11398 data points were collected from 646 sources. A source can be a document, a package of documents or a package of emails. The distribution of the sources for the datapoints can be found in Table B.1, about half (46%) can be found on the Delfi project disk and about another half (51%) in the Delfi mail folder. It is worthwhile to note that the data from the two is different. The documents on the project disk are mostly generated specifically for documentation over a significant amount of time, such as theses or technical notes. The emails on the mail server are documents that are generated as a result of communication instantaneously.

The distribution of the datapoints in time is shown in Figure B.1. The median number of datapoints found in a month is 54. In the figure it can be seen that there are also a number of months for which very little data points were found. This can be attributed either to a lack of documentation effort by team members involved, or by a lack of team members involved in these periods. The peak in September 2008 is caused by a review and its related documentation. The peak in December 2006 is caused by the release of a large amount of technical drawing with related documentation. The peak in March 2005 is caused by a high number of meetings and related documentation. The peak in July 2012 is caused by documentation related the hardware integration and testing. In November and December 2013 there is a peak in email communications related to the launch of Delfi-n3Xt. The last bar in the graph the number of data points for which the date was incomplete, these are 704 data points.

Another important observation is that the mail server delivers data since 2008, and has been used most heavily since July 2011.

The division of the datapoints over the persons involved is shown in Figure B.2. About 35% of the data points are related to 7 persons who have more than 200 data points. These are members that were involved with the complete system, and thus were present at a lot of
Figure B.1: Distribution of datapoints in time. The last bar is a summation of all points without a month, a year or neither of those. The colours indicate the source of the data. The boxplot on the right represents the spread of the point distribution for all months, excluding the points without a complete date.
B.2 Data analysis

Before the analysis of the data could be started, a number of filters were applied to the data to make the data ‘cleaner’. A project filter was applied to relate as many of the datapoints to a project as possible. Two filters were applied to fill gaps in the data and remove incidental occurrences of persons (filter 1 and 2). The filters are described in detail in Sections B.2.1, B.2.2, B.2.3. In section B.2.4 the categorisation of student levels is discussed.

The effect that the filters have on the data is shown in Figure B.5. Filter 1 only adds months, as expected. A total of 252 months is added by filter 1 compared to the raw data. After filter 1 and filter 2 are both applied, the difference is distributed normally with a mean of \(-0.61\) with a corrected sample standard deviation of 1.69. A total of 259 months is removed when compared to the original data.

<table>
<thead>
<tr>
<th>Source</th>
<th>Sub-source</th>
<th>Datapoints</th>
<th>Percentual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Disk</td>
<td></td>
<td>5270</td>
<td>46.2%</td>
</tr>
<tr>
<td>[DC3] Delfi-C3</td>
<td></td>
<td>1908</td>
<td>16.7%</td>
</tr>
<tr>
<td>[DNX] Delfi-n3Xt</td>
<td></td>
<td>2046</td>
<td>18.0%</td>
</tr>
<tr>
<td>[DFF] DelFFi</td>
<td></td>
<td>78</td>
<td>0.7%</td>
</tr>
<tr>
<td>[DPG] Delfi Program General</td>
<td></td>
<td>673</td>
<td>5.9%</td>
</tr>
<tr>
<td>1. Document Inbox</td>
<td></td>
<td>147</td>
<td>1.3%</td>
</tr>
<tr>
<td>2. Working Directory</td>
<td></td>
<td>371</td>
<td>3.3%</td>
</tr>
<tr>
<td>5. Events</td>
<td></td>
<td>47</td>
<td>0.4%</td>
</tr>
<tr>
<td>Mail Server</td>
<td></td>
<td>5854</td>
<td>51.4%</td>
</tr>
<tr>
<td>1. Hardware Orders</td>
<td></td>
<td>858</td>
<td>7.5%</td>
</tr>
<tr>
<td>2. General Information</td>
<td></td>
<td>70</td>
<td>0.6%</td>
</tr>
<tr>
<td>3. Radio Amateurs</td>
<td></td>
<td>59</td>
<td>0.5%</td>
</tr>
<tr>
<td>4. Delfi-n3Xt Operations</td>
<td></td>
<td>390</td>
<td>3.4%</td>
</tr>
<tr>
<td>5. Delfi-n3Xt Pass Reports</td>
<td></td>
<td>40</td>
<td>0.4%</td>
</tr>
<tr>
<td>Delfi-C3</td>
<td></td>
<td>927</td>
<td>8.1%</td>
</tr>
<tr>
<td>Delfi-n3Xt</td>
<td></td>
<td>3294</td>
<td>28.9%</td>
</tr>
<tr>
<td>DelFFi</td>
<td></td>
<td>216</td>
<td>1.9%</td>
</tr>
<tr>
<td>Other</td>
<td></td>
<td>274</td>
<td>2.4%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>11398</td>
<td>100%</td>
</tr>
</tbody>
</table>

meetings and involved in a lot of documentation or were very active in their email communications. In contrast the 129 people with one data point only take up 1% of the total amount of data points.

The distribution of names over the amount of data points is also shown in figure Figure B.3. The median of data points per name is 5. The outliers which are named in Figure B.2 are also clearly visible in this figure. It should also be noted that not to every datapoint contains a date. Of the 428 names identified, 33 had no complete date associated with them. Of these 33, 30 were related to the Delfi-C3 project and 3 to Delfi-n3Xt.
**Figure B.2:** Distribution of data points over names

**Figure B.3:** Distribution of names by amount of data points, excluding outliers with over 300 data points.
The two persons who gain the most months due to the filter are persons who have been involved in the projects as manager. The three that lose the most months have been involved as staff member and supervisor.

**B.2.1 Project filter**

The aim of the project filter is to related data points which are not related to one of the three projects to one of these projects. The project filter looked at all the projects related to a single name. If there where data points for which the project was unknown and the other data points were only related to a single other project, it was assumed that the person had only been part of one project. Subsequently the data points with an empty project entry were filled in with the relevant project. The diagram shown in Figure B.4 illustrates the decision structure.

**B.2.2 Filter 1**

Filter 1 aims to fill gaps in the collected data. This filter works on the assumption that if a persons is found to be involved in a month, is not found in a second subsequent month, but is found again in a third subsequent month, that person was likely part of the team in all three months. For example if 'John' was found to be part of the team in March and May, but no evidence was found of him in April, it is still assumed he was part of the team in April. This assumption is made (and this filter is applied) because it seems likely that a person might not leave a paper trail every month. For example if a person does research for 5 weeks for a technical document in a period when there were no documented meeting, this person would not leave a paper trail. The people who have only been associated with the first month (November 2004) or last month (August 2014) are not affected by this filter, since there is no earlier or later month to compare these dates with.

**B.2.3 Filter 2**

This filter implements the assumption that someone who was only present in a single month, without being found in the previous and next month was only involved in the team incidentally. For example, if ‘Jane’ is found in April, but not in March and May, it is assumed she was not a significant part of the team. Combined with filter 1 this means that occurrences of a person that have a gap of two months or more in between occurrences are filtered out.
This filter will filter out all people who have only a single date associated with them. The people who have only been associated with the first month (November 2004) or last month (August 2014) are not filtered out, since there is no earlier or later month to compare them with. An important group that is filtered out are the people associated with launch and early operations. These people are very active in the period of launch, but not in the months before and after. It is important to realise that after filter 2 is applied, the operations team is omitted. This happens since the operations teams is mostly active in the month of the launch and does not leave much subsequent documents.

B.2.4 Student Levels

In order to gain more insight into who the team members in the project were, and in what role they participated in the project, it was attempted to divide the team members along their educational level. Some people have only participated at one point in their career, while other have done so in different stages. Jasper Bouwmeester for example has been involved in the project as student and staff member. The different categories in which the people were divided are given in table B.2. Note that this categories give no insight into whether someone was involved as a thesis student or as an intern.

B.3 Results

This section discusses the results find by the analysis of the data. In section B.3.1 the size of the team will be discussed. In section B.3.2 the division of the different student levels will be discussed.
B.3 Results

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msc</td>
<td>only found to be involved as a master of science student.</td>
</tr>
<tr>
<td>msc/Other</td>
<td>found involved as a master of science student and in a non-student position.</td>
</tr>
<tr>
<td>bsc</td>
<td>only found to be involved as a bachelor of science student.</td>
</tr>
<tr>
<td>bsc/Other</td>
<td>found involved as a bachelor of science student and in a non-student position.</td>
</tr>
<tr>
<td>beng</td>
<td>only found to be involved as a bachelor of engineering student.</td>
</tr>
<tr>
<td>beng/Other</td>
<td>found involved as a bachelor of engineering student and in a non-student position.</td>
</tr>
<tr>
<td>Student</td>
<td>found to be involved as a student, no particular level information was found.</td>
</tr>
<tr>
<td>Student/Other</td>
<td>found as student and in other non-student positions.</td>
</tr>
<tr>
<td>Various</td>
<td>found to be involved at various student levels.</td>
</tr>
<tr>
<td>Various/Other</td>
<td>found to be involved at various student levels and in non-student positions.</td>
</tr>
<tr>
<td>Unknown (S)</td>
<td>found to have a certain position, however it could not be confirmed whether the person was a student.</td>
</tr>
<tr>
<td>Empty</td>
<td>not related to any position. No level information was found.</td>
</tr>
<tr>
<td>Other</td>
<td>only found in non-student positions.</td>
</tr>
</tbody>
</table>

Table B.2: Table of Student Level Categories
B.3.1 Team Size

The number of people involved in Delfi projects are graphed in Figure B.6, including launch months and month in which the satellites were delivered. For Delfi-C$^3$ the delivery was in the same month as the launch, so this is not annotated. In Figure B.7 the amount of people is also shown, but now split out for the three projects. Comparing the middle and bottom graph it can be seen how the operation team is partly filtered out after filter 2 is applied, specially for Delfi-C$^3$. All graphs show less people involved in the months prior to the launch and delivery. Delfi-C$^3$ shows a low team size from April 2007 until launch. Delfi-n3Xt shows little activity from July 2009 to September 2010. Also initially a project will have a small team size. Apart from these there does not seem to be a clear trend in the team size over the project life cycle. The team size seems rather characterised by short periods of (more or less) constant team size interchanged sudden changes in the team size.

In Figure B.8 the histogram of the team size in all the months is shown. No proper fit can be found for the distribution of the team sizes.

The statistics of the team size per month based on the data obtained with filter one and two applied is shown in Figure B.9. The median team size throughout the programme is estimated to have been 19, higher than the median of the separate projects, due to overlap. For Delfi-C$^3$ the median estimated team size is 12.5. For Delfi-n3Xt the estimated team size is 16.5, including weeks between delivery and launch, or 18, without these months. For DelFFi the team size is estimated to be 10. For Delfi-n3Xt and the complete programme the upper quartile is very spread out, while for Delfi-C$^3$ the third quartile is very spread out. In order to

Figure B.6: Approximate People related to Delfi projects over time. The project filter was applied to all three graphs.
Figure B.7: Approximate People related to the separate Delfi projects over time, colour coded for every project. The project filter was applied to all three graphs. Note that someone who was found in multiple projects in a single month is imaged for both projects in this graph.

Figure B.8: Histogram of the team size during the whole programme and the different projects. These are the team sizes after filter 1 and 2 were applied.
gain more insight in this decline, the same data has been analysed split up into the different years. The result is shown in Figure B.10 on the left. It should be noted that every boxplot is made with a maximum of twelve points. It becomes clear, that even though a decline is still visible, there are a number of years with a remarkable low median. For the year 2004 the reason for the low median is clear, the program was started in November 2004 with a few dedicated team members. Both 2007 and 2013 are years before the launch. In 2013 DelFFi was initiated slowly, while Delfi-n3Xt was waiting for launch without ongoing development. It is unclear why the team size is so low in 2010, though it makes the low team size years remarkable periodic. In the same Figure B.10 on the right the same data has been divided over the academic years. The data suggest a slow decline in the team size through the years 04/05 to 07/08, the years of Delfi-C$^3$. Academic year 11/12 shows the highest median and year 08/09 the largest spread. Academic year 09/10 shows one of the lowest medians and lowest spread, remarkable after the academic year with the second highest team size. Similar academic year 12/13 shows the lowest team size after the academic year with the highest team size, though academic year 12/13 coincides with the time between delivery and launch of Delfi-n3Xt. In order to check whether the team size changed depending on month, the statistics of the team size were also split up by months. This is shown in Figure B.11. A difference between months could be expected, since students should follow a set schedule throughout their master education. The lowest median team sizes are found in June, July and August, while the highest are found in December, January (after application of filters) and February. It should however be noted that the spread of the data is very large, only of July after the filters are applied it can be said with confidence that it has the lowest median, though also the largest outlier.

In Figure B.12 it is shown how many months were spend on the project by the persons involved. Looking at the filtered boxplot of the complete program, half of the persons involved spent between 4 and 13 months within the project. The upper quartile spreads towards 26
Figure B.10: Boxplot on estimated team size divided over the different years.

Figure B.11: Boxplots on estimated team size divided over the months.
Figure B.12: Statistics of the amount of months persons were involved in the project. Outliers who were involved for over 30 months have been excluded from this graph for clarity. Note that the figure compares the statistics found both before (boxplots 1 to 5) and after the filters (boxplots 6 to 9). From boxplots 2 to 9, persons who were not found to be involved in the project have been excluded.

months. The spread is largest for Delfi-n3Xt. The small spread and lower median of DelFFi can be explained by the fact that the project only recently started and has not seen many people graduating within that project. The median for the complete programme, Delfi-C³, and Delfi-n3Xt is 7 months after filtering.

It can be concluded that throughout the project the median team size has been 19, separated over the projects. In the months leading up to a launch or delivery, the team size seems smaller. In the initial stages of a project the team size is also smaller. There are indicators that the team size varies periodically during the year, with a minimum in July, and a maximum in January. Except for these observation, there is no clear trend visible in estimated team size. Team size can change suddenly. Half of the team members spend between 4 and 13 months on the project, with a median of 7 months.

B.3.2 Team Member Characteristics

Student involvement

The people found were divided into the student level categories as defined in table B.2. The result can be seen in Figure B.13. In the first chart in that image the distribution is given without weight, it can be seen that for only about a third of the people found it could be
confirmed that the person was a student. Also for almost half of the persons no information of their level was found. In order to get a better insight into the contribution of each person to the projects, the members were weighted by the months they were found to be involved. A weighted distribution is given in the other charts in Figure B.13, it can be seen that the contribution of the students is a lot bigger than that of the other persons. Note that msc/other group shows the highest increase in contribution, this group includes persons who worked during their master education on the projects and were subsequently hired.

In figure Figure B.14, the weighted distribution of the members over the student categories has been split up per project. It is striking that for Delfi-C\textsuperscript{3} there are hardly any people of whom no level information was found. This can be attributed to a better administration of the members and their role in the project. During Delfi-C\textsuperscript{3} about 40\% of the people involved was a master of science student as one point in the project. It should be noted that during Delfi-C\textsuperscript{3} a number of students were hired to continue work on the project. Since Delfi-C\textsuperscript{3} was also the first project in the program, it is likely that all these persons started as a master of science student in the project and became staff member or were involved as a reviewer in this or one of the future projects.

The contribution of bachelor of engineering students changes throughout the program. In Delfi-C\textsuperscript{3} they contributed about 20\% to the project, while in Delfi-n3Xt it is only 7\%. The
bachelor of science students are only significantly involved in Delfi-n3Xt.

For both Delfi-n3Xt and DelFFi, there are a large number of persons of whom no information was found on their level (‘empty’). For DelFFi this could be due to the early stage in which the project is, and the little amount of information which is available. It is possible that the ‘empty’ category mostly represents staff members, since it is not common that these mark down their level or document their work in the form of thesis or internship reports like students do, however this could not be confirmed. Additionally, there would need to be an explanation on why this hypothesis does not apply to Delfi-C3.

For all the projects master of science students remain the biggest contributors to the satellites. Master of science students are the largest contributors to the Delfi program projects. Bachelor of Science students form only a small part, and the role of Bachelor of engineering students is diminishing. Currently DelFFi is the only project where the confirmed students give a minority contribution.

Figure B.14: Pie charts of the distribution of the team members, weighted by months found, over the student levels divided by project.
B.4 Conclusions

A large amount of data was subtracted from Delfi documents and emails based on the assumption that everyone involved in the project leaves a paper trail. The data came from documents spread over a nearly 10 year period.

The data shows that a small number of people left a very large paper trail, while half of the people left 5 or less data points.

The data was filtered to fill in gaps and to exclude incidental occurrences. The combined results of these filters removed on average 0.6 month from each person.

Half of the people are involved 7 months or more, though the average is a lot higher due to outliers.

Students have delivered the largest contribution in both Delfi-C^3 and Delfi-n3Xt.
Appendix C

Recommendations Elstak

Elstak [9] has made a number of recommendation for the project management and systems engineering for university CubeSat projects. He compared a project from Aalborg University (AAU) and Delft University of Technology (TU Delft) (Delfi-C3). His findings are considered important because his research had a similar approach and similar objectives as the current research.

Elstak proposed a model philosophy as shown in table C.1.

<table>
<thead>
<tr>
<th>CubeSat</th>
<th>Model Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Prototypes</td>
</tr>
<tr>
<td>Purpose</td>
<td>Proof of subsystem functions</td>
</tr>
<tr>
<td>Functionality</td>
<td>Subsystem subfunctions</td>
</tr>
<tr>
<td></td>
<td>Comparable to Engineering Model (EM)</td>
</tr>
<tr>
<td></td>
<td>EPS architecture functional</td>
</tr>
<tr>
<td></td>
<td>CHDS architecture functional</td>
</tr>
<tr>
<td>Verification</td>
<td>Verify functions</td>
</tr>
<tr>
<td></td>
<td>Qualification testing</td>
</tr>
<tr>
<td></td>
<td>Acceptance testing</td>
</tr>
</tbody>
</table>

|               | Basic Model                             |
| Purpose       | Achieve an integrated and functional architecture |
| Functionality | Subsystem functionality                 |
|               | EPS architecture functional             |
|               | CHDS architecture functional            |
| Verification  | Qualification testing                   |
|               | Acceptance testing                      |
|               | Protolflight Model                      |
| Purpose       | Fully functional satellite              |
| Functionality | Subsystem functionality                 |
|               | EPS architecture functional             |
|               | CHDS architecture functional            |
| Verification  | Qualification testing                   |
|               | Acceptance testing                      |

Table C.1: The CubeSat Model Approach as proposed by Elstak [9]

Elstak proposed a project phasing as shown in Figure C.1.

The final recommendation of the report of Elstak are listed below, as taken from [9]:

• Team
Figure C.1: Project Phases as proposed by Elstak [9]

- A balanced team of students disciplined in: PMSE, electrical engineering and mechanical engineering
- Experienced staff members in the above mentioned fields
- Firm base of hours available for all students involved
- Steady rate of student availability during the project
- Regular progress meetings (2 weekly)

• Phasing and scheduling
  - Phasing as proposed in Figure C.1
  - Hard phase separation closed with reviews to ensure equal maturity at start of FM MAIV
  - Use of delta sessions and invest manpower in lagging subsystems
  - The MAIV phase should be equally long as the design phase

• Design and Integration
  - Top level, structured level PMSE from the start of the project
  - Academic accreditation of the entire project
  - Basic Model combined with ProtoFlight Model approach
  - Integrated development approach (HW, SW and electronics)
  - Early prototyping and breadboarding

• Information
  - Use of living documents that are kept up to date
  - Clear documentation tree and structure
  - A wiki or CVS based online verification system should be used
  - Clear requirement and interface definition
### D.1 Systems Engineering Philosophy

#### D.1.1 Criteria Comparison

<table>
<thead>
<tr>
<th>Criteria 1</th>
<th>Criteria 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-C-1</td>
<td>SEP-C-2</td>
<td>SEP-C-2</td>
<td>3</td>
<td>Adoption is a one-time investment (per team member), accessibility is needed by everyone throughout the project.</td>
</tr>
<tr>
<td>SEP-C-1</td>
<td>SEP-C-3</td>
<td>SEP-C-3</td>
<td>3</td>
<td>Adoption is a one-time investment (per team member), timeliness of updates is needed by everyone throughout the project.</td>
</tr>
<tr>
<td>SEP-C-1</td>
<td>SEP-C-4</td>
<td>SEP-C-4</td>
<td>3</td>
<td>Adoption is a one-time investment (per team member), error sensitivity is needed by everyone throughout the project.</td>
</tr>
<tr>
<td>SEP-C-1</td>
<td>SEP-C-5</td>
<td>equal</td>
<td>1</td>
<td>Both criteria judge a time investment from the team members.</td>
</tr>
<tr>
<td>SEP-C-1</td>
<td>SEP-C-6</td>
<td>SEP-C-6</td>
<td>3</td>
<td>Work force changes are a crucial problem in university satellite projects.</td>
</tr>
<tr>
<td>SEP-C-1</td>
<td>SEP-C-7</td>
<td>equal</td>
<td>1</td>
<td>Both criteria are investments from the stakeholders, either in terms of workforce availability, or in terms of finances.</td>
</tr>
<tr>
<td>SEP-C-1</td>
<td>SEP-C-8</td>
<td>SEP-C-8</td>
<td>5</td>
<td>The main objective of a university class satellite is education (see Section 2.1).</td>
</tr>
<tr>
<td>SEP-C-2</td>
<td>SEP-C-3</td>
<td>equal</td>
<td>1</td>
<td>Both criteria judge the ease with which the current state of the system can be determined.</td>
</tr>
<tr>
<td>SEP-C-2</td>
<td>SEP-C-4</td>
<td>equal</td>
<td>1</td>
<td>Both criteria will decrease the quality of the design.</td>
</tr>
</tbody>
</table>

*Continued on the next page*
<table>
<thead>
<tr>
<th>Criteria 1</th>
<th>Criteria 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-C-2</td>
<td>SEP-C-5</td>
<td>SEP-C-2</td>
<td>3</td>
<td>Non-accessible information can lead to a higher workload, thus this is valued slightly higher.</td>
</tr>
<tr>
<td>SEP-C-2</td>
<td>SEP-C-6</td>
<td>equal</td>
<td>1</td>
<td>No preference, driven by other comparisons.</td>
</tr>
<tr>
<td>SEP-C-2</td>
<td>SEP-C-7</td>
<td>equal</td>
<td>1</td>
<td>No preference, driven by other comparisons.</td>
</tr>
<tr>
<td>SEP-C-2</td>
<td>SEP-C-8</td>
<td>SEP-C-8</td>
<td>3</td>
<td>The main objective of a university class satellite is education (see Section 2.1).</td>
</tr>
<tr>
<td>SEP-C-3</td>
<td>SEP-C-4</td>
<td>equal</td>
<td>1</td>
<td>Both can lead to errors in design.</td>
</tr>
<tr>
<td>SEP-C-3</td>
<td>SEP-C-5</td>
<td>SEP-C-3</td>
<td>3</td>
<td>Slow information updates can lead to a higher workload, thus this is valued slightly higher.</td>
</tr>
<tr>
<td>SEP-C-3</td>
<td>SEP-C-6</td>
<td>equal</td>
<td>1</td>
<td>No preference, driven by other comparisons.</td>
</tr>
<tr>
<td>SEP-C-3</td>
<td>SEP-C-7</td>
<td>equal</td>
<td>1</td>
<td>No preference, driven by other comparisons.</td>
</tr>
<tr>
<td>SEP-C-3</td>
<td>SEP-C-8</td>
<td>SEP-C-8</td>
<td>3</td>
<td>The main objective of a university class satellite is education (see Section 2.1).</td>
</tr>
<tr>
<td>SEP-C-4</td>
<td>SEP-C-5</td>
<td>SEP-C-4</td>
<td>3</td>
<td>Errors can lead to a higher workload, thus this is valued slightly higher.</td>
</tr>
<tr>
<td>SEP-C-4</td>
<td>SEP-C-6</td>
<td>equal</td>
<td>1</td>
<td>Both represent the sensitivity of the methodology.</td>
</tr>
<tr>
<td>SEP-C-4</td>
<td>SEP-C-7</td>
<td>SEP-C-4</td>
<td>3</td>
<td>The cost can be relatively small compared to satellite and personal cost.</td>
</tr>
<tr>
<td>SEP-C-4</td>
<td>SEP-C-8</td>
<td>SEP-C-8</td>
<td>3</td>
<td>The main objective of a university class satellite is education (see Section 2.1).</td>
</tr>
<tr>
<td>SEP-C-5</td>
<td>SEP-C-6</td>
<td>SEP-C-6</td>
<td>3</td>
<td>Workload is a demand towards the student, if the option cannot handle the changing workforce, it will likely not succeed.</td>
</tr>
<tr>
<td>SEP-C-5</td>
<td>SEP-C-7</td>
<td>equal</td>
<td>1</td>
<td>Both represent an investment from the stakeholders.</td>
</tr>
<tr>
<td>SEP-C-5</td>
<td>SEP-C-8</td>
<td>SEP-C-8</td>
<td>5</td>
<td>The main objective of a university class satellite is education (see Section 2.1).</td>
</tr>
<tr>
<td>SEP-C-6</td>
<td>SEP-C-7</td>
<td>SEP-C-6</td>
<td>3</td>
<td>The shifting workforce is a major constraint. Controlling cost is important but implementation cost can be relatively small compared to satellite and personal cost.</td>
</tr>
<tr>
<td>SEP-C-6</td>
<td>SEP-C-8</td>
<td>SEP-C-8</td>
<td>3</td>
<td>The main objective of a university class satellite is education (see Section 2.1).</td>
</tr>
<tr>
<td>SEP-C-7</td>
<td>SEP-C-8</td>
<td>SEP-C-8</td>
<td>5</td>
<td>The main objective of a university class satellite is education (see Section 2.1).</td>
</tr>
</tbody>
</table>

Table D.1: Systems engineering philosophy criteria comparison

D.1.2 Options Comparison
D.1 Systems Engineering Philosophy

Option 1 | Option 2 | Winner | Score | Rationale
--- | --- | --- | --- | ---
SEP-O-1.1 | SEP-O-1.2 | equal | 1 | There is no difference.
SEP-O-1.1 | SEP-O-1.3 | SEP-O-1.3 | 3 | There is no need to learn about the different types of documentation in SEP-O-1.3.
SEP-O-1.1 | SEP-O-2.1 | SEP-O-1.1 | 9 | Intensive training is needed to apply MBSE
SEP-O-1.1 | SEP-O-2.2 | SEP-O-1.1 | 5 | Training is required to understand the model
SEP-O-1.1 | SEP-O-3 | SEP-O-1.1 | 4 | More training is needed to know how to use concurrent engineering
SEP-O-1.2 | SEP-O-1.3 | SEP-O-1.3 | 3 | See comparisons with SEP-O-1.1
SEP-O-1.2 | SEP-O-2.1 | SEP-O-1.2 | 9 | See comparisons with SEP-O-1.1
SEP-O-1.2 | SEP-O-2.2 | SEP-O-1.2 | 5 | See comparisons with SEP-O-1.1
SEP-O-1.2 | SEP-O-3.0 | SEP-O-1.2 | 4 | See comparisons with SEP-O-1.1
SEP-O-1.3 | SEP-O-2.1 | SEP-O-1.3 | 9 | See comparisons with SEP-O-1.1 & Table D.2
SEP-O-1.3 | SEP-O-2.2 | SEP-O-1.3 | 7 | See comparisons with SEP-O-1.1 & Table D.2
SEP-O-1.3 | SEP-O-3.0 | SEP-O-1.3 | 6 | See comparisons with SEP-O-1.1 & Table D.2
SEP-O-2.1 | SEP-O-2.2 | SEP-O-2.2 | 5 | Solely reading a model is easier than also making the model.
SEP-O-2.1 | SEP-O-3.0 | SEP-O-3 | 5 | Concurrent engineering is more intuitive.
SEP-O-2.2 | SEP-O-3.0 | SEP-O-3 | 2 | Concurrent engineering is more intuitive.

Table D.2: Systems engineering philosophy option performance on criteria SEP-C-1, adoption period

Table D.3: Systems engineering philosophy option comparison on criteria SEP-C-1, adoption period
Option performance

SEP-O-1.1 Documents are stored in a (single) digital archive, accessible by everyone with reader access. Information is linked through sourcing, though users will need to find the appropriate documents.

SEP-O-1.2 Documents are stored in a (single) digital archive, accessible by everyone with reader access. Information is linked through sourcing, though users will need to find the appropriate documents.

SEP-O-1.3 Documents are stored in a (single) digital archive, accessible by everyone with reader access. Information is linked through sourcing, though users will need to find the appropriate documents.

SEP-O-2.1 Information is stored in a single model, all information is linked.

SEP-O-2.2 Information is stored in a single model, all information is linked. Since there is a single model, the model will likely be more consistent than when there are more modellers.

SEP-O-3.0 Not all information is documented, information needs to be sought by communicating with other members, who need to be available.

Table D.4: Systems engineering philosophy option performance on criteria SEP-C-2, accessibility of information

<table>
<thead>
<tr>
<th>Option 1</th>
<th>Option 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.2</td>
<td>equal</td>
<td>1</td>
<td>No difference</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.3</td>
<td>equal</td>
<td>1</td>
<td>No difference</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>5</td>
<td>All information is linked is MBSE</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>6</td>
<td>All information is linked is MBSE</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.1</td>
<td>5</td>
<td>Not all information is documented in concurrent engineering</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-1.3</td>
<td>equal</td>
<td>1</td>
<td>No difference</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>5</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>6</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.2</td>
<td>5</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>5</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>6</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.3</td>
<td>5</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>2</td>
<td>A single modeller will result in a more coherent model.</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-3.0</td>
<td>SEP-O-2.1</td>
<td>7</td>
<td>Not all information is documented in concurrent engineering</td>
</tr>
<tr>
<td>SEP-O-2.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-2.2</td>
<td>8</td>
<td>Not all information is documented in concurrent engineering</td>
</tr>
</tbody>
</table>

Table D.5: Systems engineering philosophy option comparison on criteria SEP-C-2, accessibility of information
Option Performance Estimation

SEP-O-1.1 It is estimated that 2 weeks are spent making the document and 2 weeks are spent reviewing. 4 weeks

SEP-O-1.2 It is estimated that 2 weeks are spent making the document 2 weeks

SEP-O-1.3 In the worst case scenario, new designs and analysis are only documented after a complete educational period. This could be between 3 and 7 months. 5 months (20 weeks) is used for the estimated. 20 weeks

SEP-O-2.1 The information is nearly instantly available, with the maximum of half a week a few days.

SEP-O-2.2 It takes time to transfer new design information to the modeller, after it is immediately available. 1 week

SEP-O-3.0 The information is nearly instantly available, with the maximum of half a week a few days.

<table>
<thead>
<tr>
<th>Option 1</th>
<th>Option 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.2</td>
<td>SEP-O-1.2</td>
<td>2</td>
<td>4/2 = 2</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.3</td>
<td>SEP-O-1.1</td>
<td>5</td>
<td>20/4 = 5</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>8</td>
<td>4/0.5 = 8</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>4</td>
<td>4/1 = 4</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>8</td>
<td>4/0.5 = 8</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-1.3</td>
<td>SEP-O-1.2</td>
<td>10</td>
<td>20/2 = 10</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>4</td>
<td>2/0.5 = 4</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>2</td>
<td>2/1 = 2</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>4</td>
<td>2/0.5 = 4</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>40</td>
<td>20/0.5 = 40</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>20</td>
<td>20/1 = 20</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>40</td>
<td>20/0.5 = 40</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.1</td>
<td>2</td>
<td>1/0.5 = 2</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-3.0</td>
<td>equal</td>
<td>1</td>
<td>0.5 = 0.5</td>
</tr>
<tr>
<td>SEP-O-2.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>2</td>
<td>1/0.5 = 2</td>
</tr>
</tbody>
</table>

Table D.6: Systems engineering philosophy option performance on criteria SEP-C-3, Speed of information updates

Table D.7: Systems engineering philosophy option comparison on criteria SEP-C-3, speed of information updates. The rationale shows the calculation done for the score, based on the approximation in table D.6
The review process will likely eliminate a significant number of errors from the documentation.

There is no specific mechanic in place to mitigate or correct errors.

The review process will likely eliminate errors from the documentation.

A single model will mitigate errors, since values and concepts are linked within the model. Errors can still persist and models can represent nonsense.

A single model will mitigate errors, since values and concepts are linked within the model. However in the communication towards the modeller errors might occur. Errors can still persist and models can represent nonsense.

The closeness of the design team facilitates the discovery of errors.

<table>
<thead>
<tr>
<th>Option 1</th>
<th>Option 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.2</td>
<td>SEP-O-1.1</td>
<td>7</td>
<td>SEP-O-1.2 has no processes to correct errors, while SEP-O-1.1 has a review process</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.3</td>
<td>equal</td>
<td>1</td>
<td>No difference</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.1</td>
<td>SEP-O-1.1</td>
<td>3</td>
<td>The review process actively searches and corrects errors.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-1.1</td>
<td>4</td>
<td>The review process actively searches and corrects errors.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.1</td>
<td>4</td>
<td>The review process actively searches and corrects errors.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-1.3</td>
<td>SEP-O-1.3</td>
<td>7</td>
<td>See comparisons of SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>5</td>
<td>SEP-O-1.2 lacks any error correcting/mitigating mechanism</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>4</td>
<td>SEP-O-1.2 lacks any error correcting/mitigating mechanism</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>4</td>
<td>See comparisons of SEP-O-2.1</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.1</td>
<td>SEP-O-1.3</td>
<td>3</td>
<td>See comparisons of SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.2</td>
<td>SEP-O-1.3</td>
<td>4</td>
<td>See comparisons of SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.3</td>
<td>4</td>
<td>See comparisons of SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.1</td>
<td>3</td>
<td>The communication between team and modeller can be a source of errors.</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-3.0</td>
<td>equal</td>
<td>1</td>
<td>Both options rely on communication.</td>
</tr>
<tr>
<td>SEP-O-2.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-2.2</td>
<td>3</td>
<td>See comparisons of SEP-O-2.1</td>
</tr>
</tbody>
</table>
Both documentation and the review process will cost significant investment from the team members.

The documentation will cost time.

The documentation that is being made, needs to be made already for courses, internships or thesis. These reports were also already reviewed. The extra effort is minimal.

Modelling is part of the design, thus no extra time is spent on other tasks.

The communication of the design with the modeller will take extra time from the team members.

Interaction between the team members is part of the design process and replaces documentation. More communication between team members is needed.

### Table D.10: Systems engineering philosophy option performance on criteria SEP-C-5, workload

<table>
<thead>
<tr>
<th>Option 1</th>
<th>Option 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.2</td>
<td>SEP-O-1.2</td>
<td>5</td>
<td>Option SEP-O-1.1 will cost additional review time.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.3</td>
<td>SEP-O-1.3</td>
<td>6</td>
<td>No extra work is required for option SEP-O-1.3.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>7</td>
<td>Modelling is part of the design.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>6</td>
<td>Option SEP-O-1.1 will cost additional review time.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>7</td>
<td>Option SEP-O-1.1 will cost additional review time.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-1.3</td>
<td>SEP-O-1.3</td>
<td>4</td>
<td>No extra work is required for option SEP-O-1.3.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>5</td>
<td>Option SEP-O-1.2 requires work on extra documentation.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.2</td>
<td>4</td>
<td>Option SEP-O-1.2 requires work on extra documentation.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>5</td>
<td>Option SEP-O-1.2 requires work on extra documentation.</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>2</td>
<td>No extra work is required for option SEP-O-1.3.</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.2</td>
<td>equal</td>
<td>1</td>
<td>Neither requires extra documentation.</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>2</td>
<td>No extra work is required for option SEP-O-1.3.</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.1</td>
<td>3</td>
<td>The interaction with the modeller will cost time.</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-3</td>
<td>equal</td>
<td>1</td>
<td>Both have a small impact, the design process is integrated with the documentation process.</td>
</tr>
<tr>
<td>SEP-O-2.2</td>
<td>SEP-O-3</td>
<td>SEP-O-3.0</td>
<td>3</td>
<td>The interaction with the modeller will cost time.</td>
</tr>
</tbody>
</table>

### Table D.11: Systems engineering philosophy option comparison on criteria SEP-C-5, workload

Master of Science Thesis

N. van der Pas
Option Performance

SEP-O-1.1 The reviewers are essential, without them no documents will be released.
SEP-O-1.2 There are no key roles.
SEP-O-1.3 The reviewers are essential, but they are very likely to be present since the reviewing is part of the education of the students.
SEP-O-2.1 There are no key roles.
SEP-O-2.2 The modeller forms a key role, without the modeller there will be no model to work with.
SEP-O-3 In a good implementation of concurrent engineering, all engineers of all subsystems should be present at any time.

Table D.12: Systems engineering philosophy option performance on criteria SEP-C-6, workforce sensitivity
<table>
<thead>
<tr>
<th>Option 1</th>
<th>Option 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.2</td>
<td>SEP-O-1.2</td>
<td>5</td>
<td>No key roles in SEP-O-1.2</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.3</td>
<td>SEP-O-1.3</td>
<td>5</td>
<td>Reviewer is essential in SEP-O-1.1, while in SEP-O-1.3 the review is nearly guaranteed.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.1</td>
<td>SEP-O-2.1</td>
<td>5</td>
<td>No key roles in SEP-O-1.2</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.2</td>
<td>equal</td>
<td>1</td>
<td>Both philosophies fail if a single task remains unfulfilled.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>4</td>
<td>Concurrent design requires all subsystems to be present, but would still be somewhat functional when one is missing. SEP-O-1.1 fails if reviewers are not present.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-1.3</td>
<td>equal</td>
<td>1</td>
<td>No difference in practice.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.1</td>
<td>equal</td>
<td>1</td>
<td>Both have no key roles</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.2</td>
<td>SEP-O-1.2</td>
<td>5</td>
<td>If the single modeller is missing SEP-O-2.2 fails.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.2</td>
<td>2</td>
<td>Concurrent engineering requires all subsystem engineers to be present.</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.1</td>
<td>equal</td>
<td>1</td>
<td>No difference in practice.</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.2</td>
<td>SEP-O-1.3</td>
<td>5</td>
<td>If the single modeller is missing SEP-O-2.2 fails.</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.3</td>
<td>2</td>
<td>Concurrent design requires all subsystems to be present, but would still be somewhat functional when one is missing.</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-2.1</td>
<td>5</td>
<td>If the single modeller is missing SEP-O-2.2 fails.</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-3.0</td>
<td>SEP-O-2.1</td>
<td>2</td>
<td>Concurrent design requires all subsystems to be present, but would still be somewhat functional when one is missing.</td>
</tr>
<tr>
<td>SEP-O-2.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-3.0</td>
<td>4</td>
<td>If the single modeller is missing SEP-O-2.2 fails.</td>
</tr>
</tbody>
</table>

Table D.13: Systems engineering philosophy option comparison on criteria SEP-C-6, workforce sensitivity

<table>
<thead>
<tr>
<th>Option</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-O-1.1</td>
<td>No significant additional investment needed.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>No significant additional investment needed.</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>No significant additional investment needed.</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>Modelling software and software licenses needed for all team members</td>
</tr>
<tr>
<td>SEP-O-2.2</td>
<td>Model reading software required for all team members,</td>
</tr>
<tr>
<td>SEP-O-3.0</td>
<td>Co-location for the team needs to be arranged and an integrated computernetwork needs to be set up.</td>
</tr>
</tbody>
</table>

Table D.14: Systems engineering philosophy option performance on criteria SEP-C-7, cost
### Option 1 | Option 2 | Winner | Score | Rationale
---|---|---|---|---
SEP-O-1.1 SEP-O-1.2 | equal | 1 | No difference between options |
SEP-O-1.1 SEP-O-1.3 | equal | 1 | No difference between options |
SEP-O-1.1 SEP-O-2.1 | SEP-O-1.1 | 7 | A significantly larger investment for SEP-O-2.1 is needed due to the continued licensing of modelling software. |
SEP-O-1.1 SEP-O-2.2 | SEP-O-1.1 | 5 | A significantly larger for SEP-O-2.2 investment is needed due to the continued licensing of modelling software. |
SEP-O-1.1 SEP-O-3.0 | SEP-O-1.1 | 7 | A significant investment is needed |
SEP-O-1.2 SEP-O-1.3 | equal | 1 | No difference between options |
SEP-O-1.2 SEP-O-2.1 | SEP-O-1.2 | 7 | See comparisons with SEP-O-1.1 |
SEP-O-1.2 SEP-O-2.2 | SEP-O-1.2 | 5 | See comparisons with SEP-O-1.1 |
SEP-O-1.2 SEP-O-3.0 | SEP-O-1.2 | 7 | See comparisons with SEP-O-1.1 |
SEP-O-1.3 SEP-O-2.1 | SEP-O-1.3 | 7 | See comparisons with SEP-O-1.1 |
SEP-O-1.3 SEP-O-2.2 | SEP-O-1.3 | 5 | See comparisons with SEP-O-1.1 |
SEP-O-1.3 SEP-O-3.0 | SEP-O-1.3 | 7 | See comparisons with SEP-O-1.1 |
SEP-O-2.1 SEP-O-2.2 | SEP-O-2.2 | 3 | A single license (as for SEP-O-2.2) will cost less than |
SEP-O-2.1 SEP-O-3.0 | equal | 1 | It is assumed that a continued investment in licenses will in practice incur the same costs as the investment in co-location and a computer network. |
SEP-O-2.2 SEP-O-3.0 | SEP-O-3.0 | 3 | See comparisons with SEP-O-2.1 |

**Table D.15:** Systems engineering philosophy option comparison on criteria SEP-C-7, cost

### Option performance

<table>
<thead>
<tr>
<th>Option</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-O-1.1</td>
<td>Assignment reports are optionally part of the project documentation.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>Assignment reports are optionally part of the project documentation, but not checked in advance.</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>Assignment reports are the sole part of the documentation.</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>No reporting is performed in this approach, thus this needs to be written separately by students.</td>
</tr>
<tr>
<td>SEP-O-2.2</td>
<td>No reporting is performed in this approach, thus this needs to be written separately by students.</td>
</tr>
<tr>
<td>SEP-O-3</td>
<td>No reporting is performed in this approach, thus this needs to be written separately by students. Staff involved in the project will need to be present at the same moment in the same space as the rest of the team, making it difficult to combine this with other activities.</td>
</tr>
</tbody>
</table>

**Table D.16:** Systems engineering philosophy option performance on criteria SEP-C-8, secondary stakeholder needs

---

N. van der Pas

Master of Science Thesis
<table>
<thead>
<tr>
<th>Option 1</th>
<th>Option 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.2</td>
<td>equal</td>
<td>1</td>
<td>Performance practically the same.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-1.3</td>
<td>SEP-O-1.3</td>
<td>3</td>
<td>In option SEP-O-1.3 all time can be spent on assignment documentation, and none other.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.1</td>
<td>SEP-O-1.1</td>
<td>5</td>
<td>No reporting done in SEP-O-2.1, so reports need to be written separately.</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-2.2</td>
<td>SEP-O-1.1</td>
<td>5</td>
<td>See comparisons with SEP-O-2.1</td>
</tr>
<tr>
<td>SEP-O-1.1</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.1</td>
<td>7</td>
<td>No reporting done in SEP-O-3.0 and staff needs to be present in same space and time, making it difficult to combine the project with other activities.</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-1.3</td>
<td>SEP-O-1.3</td>
<td>3</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.1</td>
<td>SEP-O-1.2</td>
<td>5</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-2.2</td>
<td>SEP-O-1.2</td>
<td>5</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.2</td>
<td>7</td>
<td>See comparisons with SEP-O-1.1</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.1</td>
<td>SEP-O-1.3</td>
<td>6</td>
<td>No reporting done in SEP-O-2.1, so reports need to be written separately.</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-2.2</td>
<td>SEP-O-1.3</td>
<td>6</td>
<td>See comparisons with SEP-O-2.1</td>
</tr>
<tr>
<td>SEP-O-1.3</td>
<td>SEP-O-3.0</td>
<td>SEP-O-1.3</td>
<td>8</td>
<td>No reporting done in SEP-O-3.0 and staff needs to be present in same space and time, making it difficult to combine the project with other activities.</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-2.2</td>
<td>equal</td>
<td>1</td>
<td>Equal performance</td>
</tr>
<tr>
<td>SEP-O-2.1</td>
<td>SEP-O-3.0</td>
<td>SEP-O-2.1</td>
<td>3</td>
<td>Staff needs to be present in same space and time in SEP-O-3.0, making it difficult to combine the project with other activities.</td>
</tr>
<tr>
<td>SEP-O-2.2</td>
<td>SEP-O-3.0</td>
<td>SEP-O-2.2</td>
<td>3</td>
<td>See comparisons with SEP-O-2.1</td>
</tr>
</tbody>
</table>

Table D.17: Systems engineering philosophy option comparison on criteria SEP-C-8, adoption period
D.2  Model Approach

D.2.1 Criteria Comparison

<table>
<thead>
<tr>
<th>Criteria 1</th>
<th>Criteria 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA-C-1</td>
<td>MA-C-2</td>
<td>MA-C-2</td>
<td>5</td>
<td>Time has shown to be less important, while the cost of the models have a major impact on the cost of the project.</td>
</tr>
<tr>
<td>MA-C-1</td>
<td>MA-C-3</td>
<td>MA-C-3</td>
<td>5</td>
<td>Time has shown to be of less important. Students deliver (relatively) low quality work, increasing the need for inherent reliability.</td>
</tr>
<tr>
<td>MA-C-1</td>
<td>MA-C-4</td>
<td>MA-C-4</td>
<td>4</td>
<td>Time has shown to be less important, a less robust methodology will likely indirectly increase the project-time, due to time needed to fix issues die to differences in maturity.</td>
</tr>
<tr>
<td>MA-C-2</td>
<td>MA-C-3</td>
<td>equal</td>
<td>1</td>
<td>Reliability issues can lead to the reduction of mission objectives. The cost of the models have a major impact on the cost of the project.</td>
</tr>
<tr>
<td>MA-C-2</td>
<td>MA-C-4</td>
<td>MA-C-2</td>
<td>3</td>
<td>The cost of the models have a major impact on the cost of the project.</td>
</tr>
<tr>
<td>MA-C-3</td>
<td>MA-C-4</td>
<td>MA-C-3</td>
<td>2</td>
<td>Students deliver (relatively) low quality work, increasing the need for inherent reliability. A less robust design will however increase the need for reliability and vice versa.</td>
</tr>
</tbody>
</table>

Table D.18: Model Approach criteria comparison

D.2.2 Options Comparison
A single Engineering Model (EM) should be made and verified, after which a redesign is done, a flight model produced, which is verified.

A first prototype will need to be made and verified, though this prototype only needs to show functionality and some issues can be remaining. The second EM should be a full EM which needs to be produced and verified, after a redesign the Flight Model (FM) is produced and verified.

A proto-flight model needs to be made and verified, after which parts need to be adapted for the final flight model, this will take less time than the production of a new model.

A basic model should be made and verified that shows functionality, and some environmental verification. A proto-flight model needs to be produced and verified, and quick fixes to the last model.

This is difficult to determine, since the project can be finished at any point. For an assumption a single complete renewal is assumed.

<table>
<thead>
<tr>
<th>Option 1</th>
<th>Option 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA-O-1</td>
<td>MA-O-2</td>
<td>MA-O-1</td>
<td>5</td>
<td>The double prototype approach will require significantly more time since a second system cycle is needed.</td>
</tr>
<tr>
<td>MA-O-1</td>
<td>MA-O-3</td>
<td>MA-O-3</td>
<td>4</td>
<td>The proto-flight approach requires only a refurbishment cycle, where the prototype approach requires the development of a new model.</td>
</tr>
<tr>
<td>MA-O-1</td>
<td>MA-O-4</td>
<td>equal</td>
<td>1</td>
<td>Both will likely require the same amount of time</td>
</tr>
<tr>
<td>MA-O-1</td>
<td>MA-O-5</td>
<td>MA-O-5</td>
<td>3</td>
<td>Since the baseline approach has a flexible end date, it scores slightly better than the prototype approach.</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>MA-O-3</td>
<td>MA-O-3</td>
<td>7</td>
<td>The proto-flight has a significantly shorter cycle than the double-prototype approach.</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>MA-O-4</td>
<td>MA-O-4</td>
<td>5</td>
<td>MA-O-1 and MA-O-4 score equally, thus see MA-O-1 versus MA-O-2.</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>MA-O-5</td>
<td>MA-O-5</td>
<td>7</td>
<td>The three models will take significantly more time to produce and is not flexible.</td>
</tr>
<tr>
<td>MA-O-3</td>
<td>MA-O-4</td>
<td>MA-O-3</td>
<td>4</td>
<td>MA-O-1 and MA-O-4 score equally, thus see MA-O-1 versus MA-O-3.</td>
</tr>
<tr>
<td>MA-O-3</td>
<td>MA-O-5</td>
<td>MA-O-5</td>
<td>2</td>
<td>Since the baseline approach has a flexible end date, it scores slightly better than option MA-O-3</td>
</tr>
<tr>
<td>MA-O-4</td>
<td>MA-O-5</td>
<td>MA-O-5</td>
<td>3</td>
<td>MA-O-1 and MA-O-4 score equally, thus see MA-O-1 versus MA-O-5.</td>
</tr>
</tbody>
</table>

Table D.19: Model Approach option performance on criteria MA-C-1, time

Table D.20: Model Approach option comparison on criteria MA-C-1, time
168 Trade Off Comparisons

Option Performance Approximation

<table>
<thead>
<tr>
<th>Option</th>
<th>Performance</th>
<th>Approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA-O-1</td>
<td>Two models need to be produced, and EM and a FM</td>
<td>2</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>Three models need to be produced, albeit the first is not a full model</td>
<td>3</td>
</tr>
<tr>
<td>MA-O-3</td>
<td>A proto-flight model will need to be made, after which some parts need to be renewed.</td>
<td>1.5</td>
</tr>
<tr>
<td>MA-O-4</td>
<td>Two models need to be made</td>
<td>2</td>
</tr>
<tr>
<td>MA-O-5</td>
<td>A baseline need to be produced, and updated. A flight model might need to be produced.</td>
<td>2</td>
</tr>
</tbody>
</table>

Table D.21: Model Approach option performance on criteria MA-C-2, cost

Option 1     Option 2     Winner     Score     Rationale

<table>
<thead>
<tr>
<th>MA-O-1</th>
<th>MA-O-2</th>
<th>MA-O-1</th>
<th>3/2</th>
<th>2 &lt; 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA-O-1</td>
<td>MA-O-3</td>
<td>MA-O-3</td>
<td>4/3</td>
<td>2 &gt; 1.5 and 2/1.5 = 4/3</td>
</tr>
<tr>
<td>MA-O-1</td>
<td>MA-O-4</td>
<td>equal</td>
<td>1</td>
<td>2 = 2</td>
</tr>
<tr>
<td>MA-O-1</td>
<td>MA-O-5</td>
<td>equal</td>
<td>1</td>
<td>2 = 2</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>MA-O-3</td>
<td>MA-O-3</td>
<td>2</td>
<td>3 &gt; 1.5 and 3/1.5 = 2</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>MA-O-4</td>
<td>MA-O-4</td>
<td>3/2</td>
<td>3 &gt; 2</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>MA-O-5</td>
<td>MA-O-5</td>
<td>3/2</td>
<td>3 &gt; 2</td>
</tr>
<tr>
<td>MA-O-3</td>
<td>MA-O-4</td>
<td>MA-O-3</td>
<td>4/3</td>
<td>1.5 &gt; 2 and 2/1.5 = 4/3</td>
</tr>
<tr>
<td>MA-O-3</td>
<td>MA-O-5</td>
<td>MA-O-3</td>
<td>4/3</td>
<td>1.5 &gt; 2 and 2/1.5 = 4/3</td>
</tr>
<tr>
<td>MA-O-4</td>
<td>MA-O-5</td>
<td>equal</td>
<td>1</td>
<td>2 = 2</td>
</tr>
</tbody>
</table>

Table D.22: Model Approach option comparison on criteria MA-C-2, the rationale refers to the approximations from D.22
Option Performance Opportunities

MA-O-1 The EM offers a single extensive opportunity to find errors. Further errors can be discovered in the FM, which can result in quick fixes. 1 to 1.5

MA-O-2 There are three opportunities to discover errors, there are two opportunities to fully solve then and one opportunity for quick fixes. The Double prototype approach was implemented in Delfi-n3Xt, but was not successfully implemented, thus no conclusion can be made on basis of this case. 2 to 2.5

MA-O-3 There is one opportunity to discover errors, but the opportunities to solve these are limited. The Proto-flight approach proved to be unsuccessful in Delfi-C3. about 0.5

MA-O-4 There is a change on subsystem level to find errors and subsequently one opportunity after testing of the basic model. 1.5 to 2

MA-O-5 Every part is verified within the current baseline, giving ample of opportunity to handle errors, either by improving the subsystem design, or by rejecting the change. -

Table D.23: Model Approach option performance on criteria MA-C-3, reliability. Opportunities gives an approximation of the number of opportunities to find and fix errors. Since there are more elements to take into account in this trade off, this is not the sole basis for the comparison

Option 1 | Option 2 | Winner | Score | Rationale
--- | --- | --- | --- | ---
MA-O-1 | MA-O-2 | MA-O-2 | 4 | 1-1.5 opportunities versus 2-2.5 opportunities.
MA-O-1 | MA-O-3 | MA-O-1 | 9 | The Proto-flight approach proved to be unsuccessful in Delfi-C3.
MA-O-1 | MA-O-4 | MA-O-4 | 2 | 1-1.5 opportunities versus 1.5-2 opportunities.
MA-O-1 | MA-O-5 | MA-O-5 | 5 | In the baseline evolution full verification is performed on changes to baseline.
MA-O-2 | MA-O-3 | MA-O-2 | 9 | The Proto-flight approach proved to be unsuccessful in Delfi-C3.
MA-O-2 | MA-O-4 | MA-O-2 | 2 | 1-1.5 opportunities versus 2-2.5 opportunities
MA-O-2 | MA-O-5 | MA-O-5 | 2 | 2 to 2.5 versus a full verified baseline.
MA-O-3 | MA-O-4 | MA-O-4 | 9 | The Proto-flight approach proved to be unsuccessful in Delfi-C3.
MA-O-3 | MA-O-5 | MA-O-5 | 9 | The Proto-flight approach proved to be unsuccessful in Delfi-C3.
MA-O-4 | MA-O-5 | MA-O-5 | 3 | 1.5-2 opportunities versus full verified baseline.

Table D.24: Model Approach option comparison on criteria MA-C-3, reliability
### Trade Off Comparisons

<table>
<thead>
<tr>
<th>Option</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA-O-1</td>
<td>Both the EM and the FM can be delayed. The EM must wait till all design work has finished. The FM must wait until all discovered errors are resolved.</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>The first model can have issues remaining and is thus less effected by the non-parallel design. The EM must wait until all design work is finished, but due to the first model, some issues are already resolved, resulting in less issues to solve before the FM model.</td>
</tr>
<tr>
<td>MA-O-3</td>
<td>The proto-flight model can only be made after all parts have completely been designed.</td>
</tr>
<tr>
<td>MA-O-4</td>
<td>The basic model has a little bit of flexibility, but most elements should be finished. For the flight model all elements should be finished.</td>
</tr>
<tr>
<td>MA-O-5</td>
<td>Since parts are primarily improved, it can handle non-parallel design very well.</td>
</tr>
</tbody>
</table>

#### Table D.25: Model Approach option performance on criteria MA-C-4, robustness

<table>
<thead>
<tr>
<th>Option 1</th>
<th>Option 2</th>
<th>Winner</th>
<th>Score</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA-O-1</td>
<td>MA-O-2</td>
<td>MA-O-2</td>
<td>4</td>
<td>Due to the spread in design in the double prototype approach is more flexible.</td>
</tr>
<tr>
<td>MA-O-1</td>
<td>MA-O-3</td>
<td>MA-O-3</td>
<td>5</td>
<td>The proto-flight model only has a sole model to finish, thus more flexibility to distribution the tasks before completion of this sole model.</td>
</tr>
<tr>
<td>MA-O-1</td>
<td>MA-O-4</td>
<td>MA-O-4</td>
<td>7</td>
<td>The deadlines of the Elstak approach are more flexible.</td>
</tr>
<tr>
<td>MA-O-1</td>
<td>MA-O-5</td>
<td>MA-O-5</td>
<td>7</td>
<td>The baseline approach can be adapted completely based on the available workforce.</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>MA-O-3</td>
<td>MA-O-3</td>
<td>3</td>
<td>The proto-flight approach has a single model that requires tasks to be complete, while the double prototype approach has 2.</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>MA-O-4</td>
<td>MA-O-4</td>
<td>3</td>
<td>Elstak approach has less strict demands for the several models.</td>
</tr>
<tr>
<td>MA-O-2</td>
<td>MA-O-5</td>
<td>MA-O-5</td>
<td>7</td>
<td>The baseline approach can be adapted completely based on the available workforce.</td>
</tr>
<tr>
<td>MA-O-3</td>
<td>MA-O-4</td>
<td>MA-O-4</td>
<td>3</td>
<td>Elstak approach has less strict demands for the several models.</td>
</tr>
<tr>
<td>MA-O-3</td>
<td>MA-O-5</td>
<td>MA-O-5</td>
<td>5</td>
<td>The baseline approach can be adapted completely based on the available workforce.</td>
</tr>
<tr>
<td>MA-O-4</td>
<td>MA-O-5</td>
<td>MA-O-5</td>
<td>3</td>
<td>The baseline approach can be adapted completely based on the available workforce.</td>
</tr>
</tbody>
</table>

#### Table D.26: Model Approach option comparison on criteria MA-C-4, robustness
Attached to this document is a CD-rom with digital files related to this research. The files can also be found on http://1drv.ms/1DMsjck. This appendix list their file names and describes the content.

TeamResearch.xlsx This file contains the data and data change log for the research presented in Chapter 4 and Appendix B.
System_and_Requirement_List.xlsx This file contains an empty configuration item list, work package list, requirement list and TBD list.
EXAMPLE_System_and_Requirement_List.xlsx This file contains the configuration item list, work package list, requirement list and TBD list for the example given in Section 7.10.
EXAMPLE_Configuration items.docx This file contains an example description of a configuration item. The example is related to the example given in Section 7.10.
EXAMPLE_Work Package Description This file contains an example of a work package description. The example is related to the example given in Section 7.10.
Bibliography


[23] ESA, “European Student Earth Orbiter / Education / ESA.” http://www.esa.int/Education/European_Student_Earth_Orbiter [accessed 3 Feb 2015].


[53] ECSS secretariat, “ECSS-M-ST-10C Rev. 1 Space Project management; Project planning and implementation,” 2009.


N. van der Pas

Master of Science Thesis


Budget, 104–107
bus baseline, 77

Concept Baseline, 75
Configuration item, 12–13, 67–72, 97–102

Developed Configuration Item, 68
flexible, on-the-fly, systems engineering, 96
golden notes, 127
Life Cycle, 36–41

Methodology, 3
Non-developed Configuration Item, 68

Project cost parameter, 50
Project quality parameter, 50
Project time parameter, 50

Requirement, 102–104

Stakeholder Role, 29–36
Student quality parameter, 48
Student time parameter, 49

Use Case, 42–48

Work package, 72–75