REQUIREMENTS SPECIFICATION

Editor: Claes Arvidson

Version 1.0

Status

Controlled	[Namn]	[Datum]
Accepted	[Namn]	[Datum]

Project identity

TSRT62, CDIO project 2015 (autumn), Optimore
Linköpings tekniska högskola, MAI

Name	Responsibility	Phone number	Mail
Akdas Hossain	Document manager	073-784 12 58	akdho545@student.liu.se
Claes Arvidson	Requirement manager	070-345 73 04	claar324@student.liu.se
Emelie Karlsson	Project manager	070-519 48 20	emeka813@student.liu.se
Hendric Kjellström	GUI manager	073-671 27 84	henkj080@student.liu.se
Jonathan Andersson	Test manager	070-932 65 33	jonan860@student.liu.se
Victor Bergelin	Design manager	070-826 34 53	vicbe348@student.liu.se

Customer: Elina Rönnberg, Linköpings Universitet 581 83 LINKÖPING

Supervisor: Torbjörn Larsson

Customer description, 581 00 LINKÖPING,
Customer phone number 013-11 00 00, fax: 013-10 19 02

Responsible for customer (examinor): Danyo Danev, phone number 013-281335, e-mail address danyo.danev@liu.se
Supervisor: Torbjörn Larsson, phone number 013-282435, e-mail address torbjörn.larsson@liu.se

Table of Contents

Document history

Version	Date	Modifications	Made by	Revised by
0.1	2015-09-15	First draft	Project group	All
0.2	2015-09-24	Second draft from feedback	Project group	All
0.3	2015-09-29	Third draft from feedback	Project Group	All
1.0	2015-10-05	Final draft from feedback	Project Group	All

1 Project overview

An avionic system is defined as the electronic system in airplanes and consists of software, processing units, buses, sensors and actuators designed to achieve a desired function in the airplane. The functions performed by the avionic system can vary from the processing of sensor data, to the presentation of information in the cockpit, to realizing functions related to for example passenger comfort or on-board power distribution. Because of a pressing need to reduce weight in both civil and military airplanes, together with a growing demand for new functionality, the modern day avionic systems are forced to operate in modules, through which a large number of functions, or tasks as they shall be referred to in this document, will have to share hardware unit. This is completely different from the traditional way of designing avionic systems, where each function was associated with a certain hardware unit.[1]

In this project, it will be investigated how tasks with high complexity of conditions can be scheduled on hardware when the constraints are already known, i.e. static scheduling. Such constraints can include dependencies between tasks, so that they need to be placed in a certain order as well as limits to when a specific task can be executed. The scheduling will be realized by heuristic optimization methods and the goal of the project is to investigate how fast and how often the different methods can produce feasible solutions to the scheduling optimization problem.

The requirements on the system we intend to develop will be defined with the following parameters in the document:

- Requirement number, enumerating the requirements..
- Requirement type
  - Original
  - Modified. Modified requirements are those that have been renegotiated with the customer during the project.
- A short description of the requirement
- Priority
  - Priority 1: The requirement has to be fulfilled by the day of final delivery.
  - Priority 2: The requirement will be pursued if there is time available.
  - Priority 3: The requirement will be pursued if there is time available but only after all of the priority 2 requirements are fulfilled.
  - Priority 4: The requirement will be pursued if there is time available but only after all of the previous requirements are fulfilled.

The requirements are listed in tables such as:

Requirement number	Requirement type	A short description of the requirement	Priority

[1] AL SHEIK, p. 27-42

1.1 Concerned parties involved

The parties involved in this project are the group members of ‘Optimore’, the customer Elina Rönnberg, supervisor Torbjörn Larsson and the examinor Danyo Danev.

1.2 Problem description

The problem which the group will solve is a scheduling problem where a maximum of 10^4 tasks shall be distributed over a maximum of 30 timelines. These timelines represent a hardware module in the avionic system of an airplane. The tasks that are to be scheduled can be interdependent in such a way that they need to be scheduled in a certain order. The tasks belong to a certain timeline, but there can be dependencies between tasks on different timelines, creating one more dimension of complexity to the problem. A simple task scheduling on different timelines is illustrated in figure 2.

Apart from dependency constraints, every task will also have three different timing constraints: time duration, starting time and finishing time. The starting and finishing times are to be viewed as hard deadlines between which the task must be executed. The timing constrains are illustrated in figure 1.

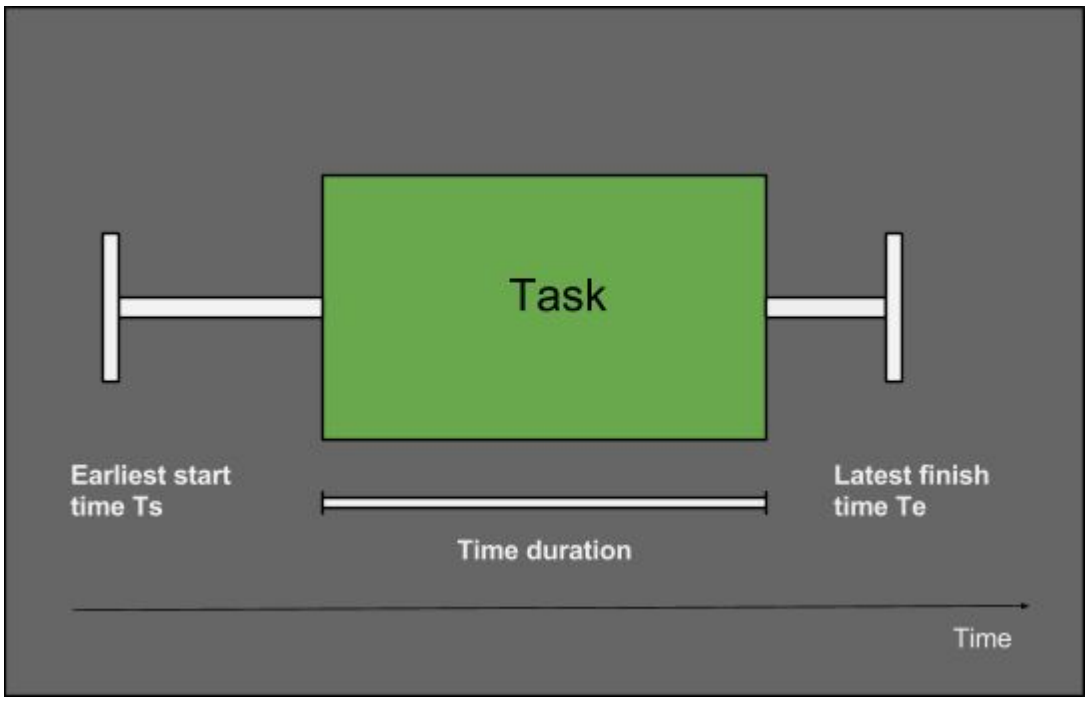

Figure 1: Timing constraints of a task.

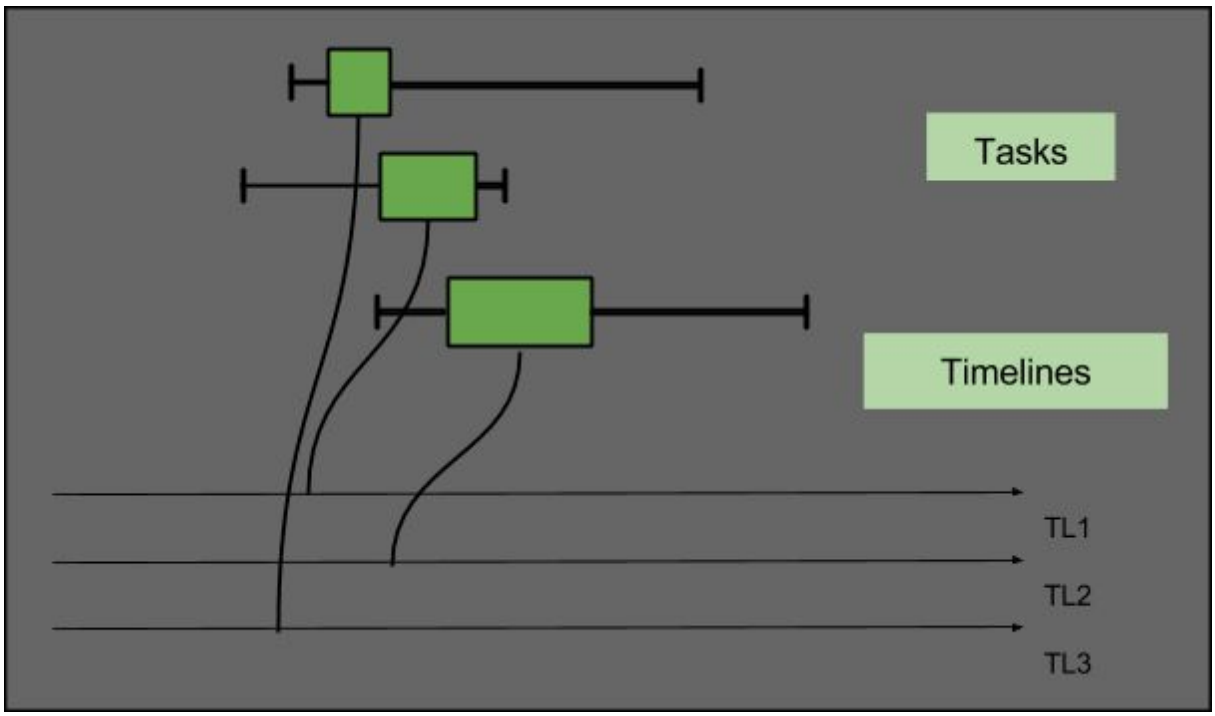

Figure 2: Scheduling of tasks on three different timelines.

1.3 Purpose and goal of the project

The purpose of the project is to examine different heuristic methods for solving the scheduling problem in order to evaluate their effectiveness in finding a feasible solution. As the project is limited in time, two heuristics with very different properties will be analyzed; tabu search and large neighbourhood search which will be compared to the commercially available CPLEX solver. The final conclusions from this analysis will then be presented in a report. This will provide the customer with a reference for choosing the correct heuristic for different avionic scheduling problems. The heuristic methods will be analyzed through extensive testing, using sets of test data designed according to the requirements in this document.

The aim of the project is not to find heuristics that outperform already existing commercial solvers such as CPLEX, but to investigate if the chosen heuristics will perform well enough on the specific type of data that is generated by avionic systems that it could give a faster and more reliable result than commercial solvers. If there is such a heuristic, this heuristic can be researched further by the customer as a potential option to commercial solvers.

1.4 Background information

This project will be conducted as part of the course TATA62 which is a CDIO project course at Linköping University. The class is a compulsory part of the education programme for the group members and is meant to have a real significance to the customer. The aim of the class is therefore to prepare the future engineers for the working life by giving practical experience in how to work in projects.

1.5 Definitions

- **Heuristic:** a technique designed for solving a problem more quickly than classical methods, usually with more approximation and relaxation.[2]
- **Avionics:** electronic systems used in air crafts.[3]
- **Tabu search:** A search algorithm that discourage the search from coming back to previously visited solutions.[4]
- **LNS - Large neighbourhood search:** A heuristic method used to explore complex neighbourhoods of data. The algorithm iterates and improves its search path.[5]

[2] Heurustic, https://en.wikipedia.org/wiki/Heuristic_(computer_science)

[3] Avionics, https://en.wikipedia.org/wiki/Avionics

[4] Tabu search, https://en.wikipedia.org/wiki/Tabu_search

[5] E.-G. Talbi, From design to implementation. Wiley, 2009

2 Overview of the system

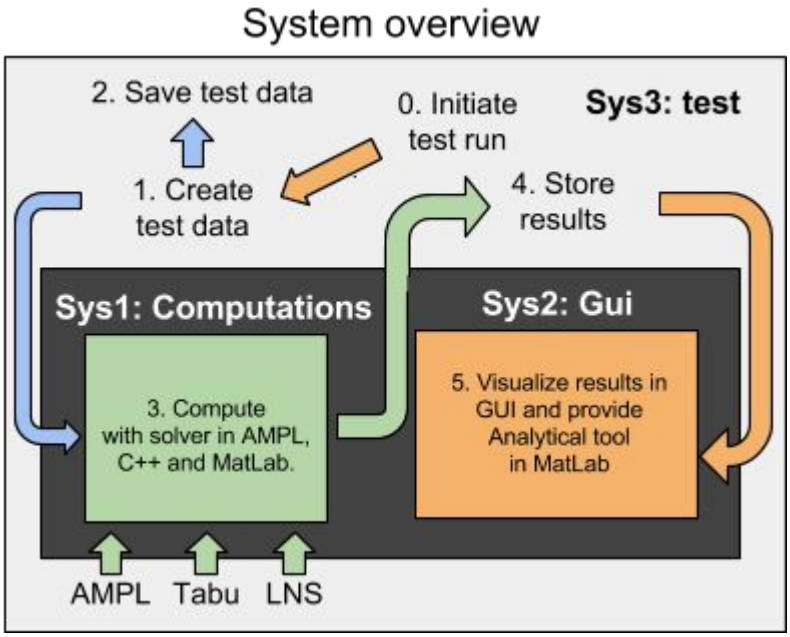

Figure 3: *A picture showing an overview of the complete system and three subsystems.*

2.1 Subsystems

Since the purpose of the project is not to deliver a product, but a report on how well the chosen heuristic methods function for task scheduling problems in avionic systems, a thorough testing environment will be developed. This environment will be referred to as the system and will consist of three subsystems: the computational software which will solve the scheduling problem, a graphical interface which will be used to present the results and configure testing, and a system to preform and create the test cases. The first and the second subsystem will communicate automatically so that user input given through the GUI will be automatically handled by the computational system. The third subsystem will generate data for the other two systems and can thus be viewed as an external system, as illustrated in figure 3.

2.2 Design philosophy

The product should be intuitive to use and the research should be both exploratory and concrete. The software should be written to be time efficient without compromising other requirements, in order for the group to be able to assess the efficiency of a particular heuristic methods. In order to make it possible for the project code to be further developed in the future, the code is to be written intuitively, upholding good coding standards. The project group will try to adhere to Google's coding standards for C++.

2.3 General requirements for the system

The following requirements are general requirements for the entire system.

Requirement 1	Original	All code should include comments	**1**
Requirement 2	Original	All code should in principle follow Google's coding standards[6]	**1**
Requirement 3	Original	Technical documentation will be written	**1**
Requirement 4	Original	At least two heuristic models will be investigated	**1**

[6] https://google-styleguide.googlecode.com/svn/trunk/cppguide.html

3 Subsystem 1 - Algorithms

Subsystem 1 consists of software that schedules tasks on timelines. The system takes a simple data file as input, containing a certain number of tasks as well as information about the dependency and timing conditions of each task. Depending on what solution method has been chosen, the data will be processed using one selected module; either by the CPLEX solver directly or by using one of the two heuristics investigated in the project. The final output of the subsystem consists of a data file with statistical information resulting from the scheduling computations. This is illustrated in figure 4.

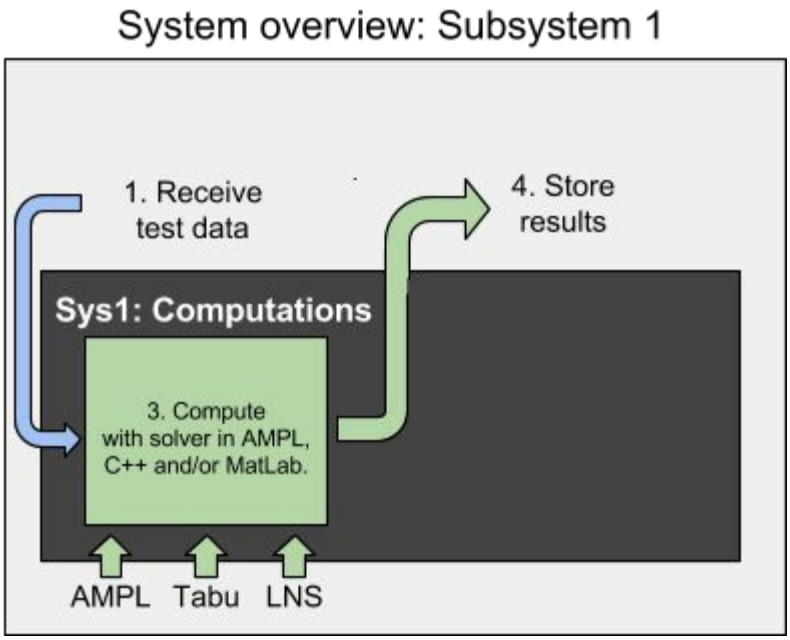

Figure 4: Input and output of subsystem 1.

3.1 Module 1: modelling in AMPL

In order to the verify that a solution exists to the test data generated by Subsystem 3, the problem can be formulated as an optimization problem with a mathematical model, solvable by any commercial solver. Such a mathematical model was provided by the customer to the group. Using this model, AMPL ("A Mathematical Programming Language") code will be written so that the data can be run using the CPLEX solver. The advantage of having this as a module in Subsystem 3 is that the results from this solver can be compared to the heuristic modules that will be implemented during the project.

3.2 Module 2: Tabu Search function

One of the heuristics that is to be investigated is the tabu search heuristic. The function will be implemented using C++ in order to be able to create time and memory efficient data structures and thus make the function time efficient. This function will not use a commercial solver, instead the module should include all the tools needed to perform a tabu search.

3.3 Module 3: LNS function

The last module is the large neighbourhood search module. The function will be based on the search heuristic with the same name and, as opposed to the tabu search function, the LNS function will use the CPLEX solver for its computations. This function will use the mathematical model in AMPL and the CPLEX solver will be used to solve sub problems as part of the LNS algorithm. The code will be written in AMPL, but transferred to C++ if the other requirements are met.

3.4 Requirements on subsystem 1

All of the requirements that affect how the three modules are designed are included in the table. The different types of data referred to are defined under subsection 5.2.

Requirement 5	Original	Develop a computing system to test heuristics.	1
Requirement 6	Original	There should be a basic implementation of the three modules	1
Requirement 7	Original	Each module should handle three time lines. All with a fixed length of 10^9 points.	1
Requirement 8	Original	Each module should be able to schedule level A and B data.	1
Requirement 9	Original	Each module should be evaluated with 10^4 tasks and 30 different time lines.	1
Requirement 10	Original	The system should be able to handle dependencies that can exist between pairs of tasks.	1
Requirement 11	Original	The system should be able to handle chains of dependencies.	1
Requirement 12	Original	The system should seek one allowed solution and finish if requirements are met or surrender after given time.	1

Requirement 13	Original	Test and report with comparison of AMPL with CPLEX, Tabu search and LNS.	1
Requirement 14	Original	The two heuristic functions should give feasible solutions for level C data.	2
Requirement 15	Original	The system should utilize cyclic time lines.	2
Requirement 16	Original	The LNS solver implemented in C++.	3
Requirement 17	Original	Design and evaluate a new type of heuristic.	4

4 Subsystem 2 - Graphical User Interface

Subsystem 2 is a MatLab implemented software which will be used to present the results of the computations in Subsystem 1 in graphs. This is illustrated in figure 5. This subsystem will also be using the statistical information provided by Subsystem 1 in order to present statistical information based on several runs. Another important role of the GUI is that it will be used to initialize the computations in Subsystem 2 for a specified test data file.

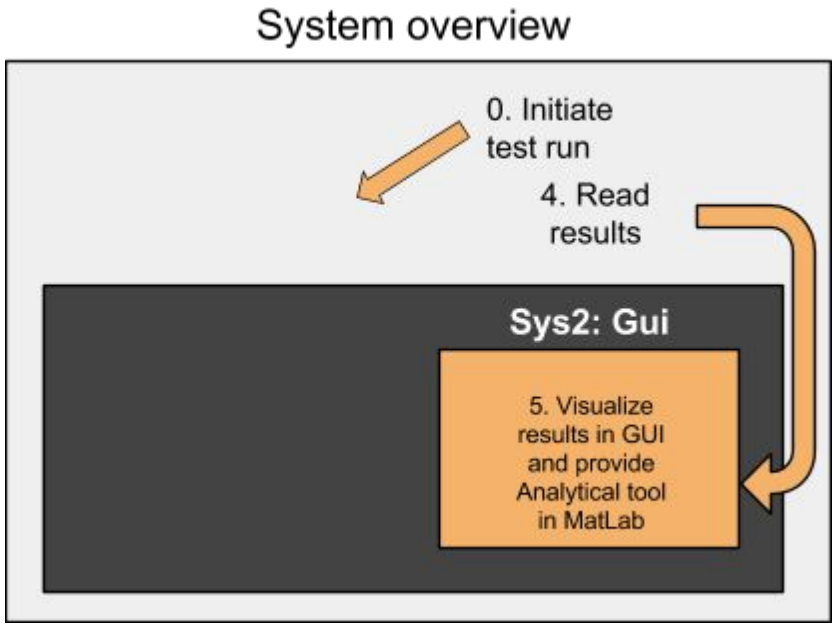

Figure 5: Picture of Subsystem 2

4.1 Requirements on subsystem 2

Requirement 18	Original	Present performance data from results using graphs.	1
Requirement 19	Original	The GUI can be used to initialize computations on a specified data file.	1
Requirement 20	Original	Present data consequent to optimization run.	1
Requirement 21	Original	Include options of what results to include.	1

Requirement 22	Original	Information about the GUI will be written in the user manual.	1
Requirement 23	Original	The GUI will show the parameters of the data.	1
Requirement 24	Original	The GUI will be able to analyze and show differences between heuristics.	1
Requirement 25	Original	The GUI will be able to compare statistical information retrieved from many runs.	2

5 Subsystem 3 - test system

A vast majority of the project will boil down to testing and evaluating the different heuristics. To facilitate this, the project will need to be initiated with generation of test data. Sets will vary both in complexity and size. For example, a data set may consist of only 100 tasks with no dependencies at all, whereas a different data set may consist of thousands of tasks with thousands of dependencies.

To evaluate solvers for the scheduling problem a program for testing called test system will be constructed. With the test system the user shall be able to generate a large set of test data using a test data generator. The user will be able to customize the test data generator. The test data will be saved for later use. In the test system the user should also be able to test different solvers by letting different solvers solve the same test data. When a solver is tested the resulting data from the test will be shown in the form of graphs and text.

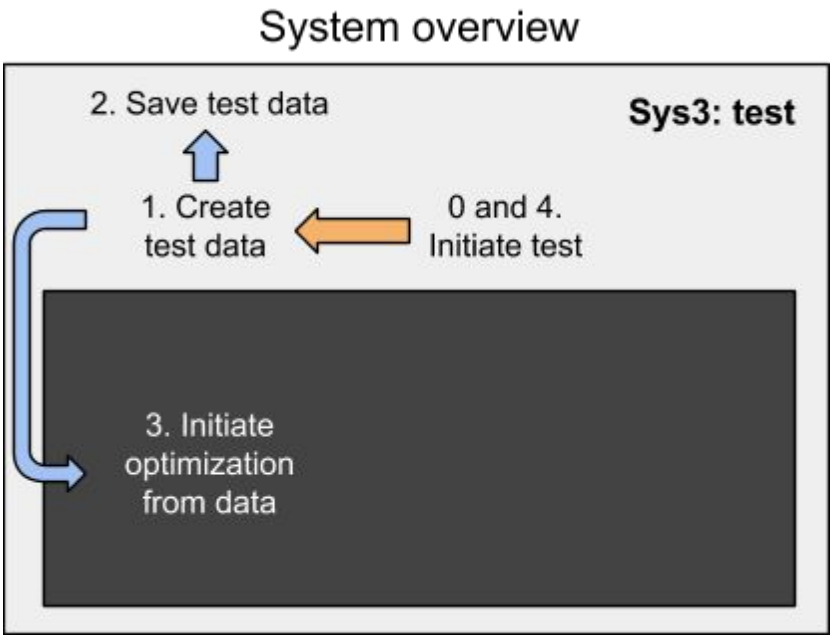

Figure 6: Picture of Subsystem 3

5.1 Introduction to the test system

The test system is a handler that will be scheduling, creating and running continuous tests of the heuristic methods in many different sets of test data, see figure 5. The testing with this system should be autonomous after initial setup, exploring all specified combination of test data (see next sub-section for more information).

One discussed evolution of the test system is to require further automation and assistance in creating the test cases.

5.2 Requirements for test system

Requirement 26	Original	Testing should be able to run automatically from set boundaries of the test data	1
Requirement 27	Original	The system should save the test data after creation	1
Requirement 28	Original	All results of the tests should be logged for consistency and for analyzing the tests	1

Requirement 29	Original	Create a graphical interface for creating test cases	2
Requirement 30	Original	The test system should be helping in choosing test cases to and provide a more autonomous experience	3

5.3 Introduction to the test data

A test data generator will be constructed that generate a collection of data sets. The generator will be generating a number of such data sets randomly according to a chosen probability distribution that decides probability for the parameters: execution time, earliest start time, latest stop time, number of timelines and dependencies. These parameters can be dependent of each other. In order to analyze the data structure, compare the results and allow reproducibility we will store all test data.

5.4 Requirements for test data

The priorities are ordered with simple, medium and hard. Each priority is directly linked to each milestone from MS4 to MS6. The complexity of the data is labeled as level A for the simplest data, and level C for the most complex test data.

Requirement 31	Original	Generate sets of data containing tasks with varying execution times, starting time and ending time (no dependencies).	A	1
Requirement 32	Original	Include dependencies in data generation.	A	1
Requirement 33	Original	Generate sets of data that include chains of dependencies.	B	1
Requirement 34	Original	A set of data containing three time lines and 100 tasks.	B	1
Requirement 35	Original	Tasks that can be periodic on a subset of the timelines	C	1
Requirement 36	Original	Dependencies can exist between tasks on different time lines.	C	1
Requirement 33	Original	The tasks are distributed over 30 timelines and 10^4 tasks with cycles are created.	C	1

6 Economy

In this case the economy will be the hours available for the group members to deliver the product.

Requirement 37	Original	The group is to spend 240 hours per person on the project. A total of 1440 hours.	1

7 Delivery requirements

These are the requirements the project has to deliver to the customer.

Requirement 38	Original	A final report, project review and the code will be delivered to the customer.	1
Requirement 39	Original	A presentation of the work will be held for the customer.	1

8 Documentation

These following documents are those that will be produced during the course of the project.

Document	**Language**	**Target audience**	**Format/media**
Time plan	English	Customer	Google spread sheet
Project plan	English	Customer	PDF
System sketch	English	Customer	PDF
User manual	English	Customer	PDF
Design specification	English	Customer	PDF
Final report	English	Customer	PDF
Project Review	English	Customer	PDF
Technical documentation	English	Customer	PDF
Program code	English	Customer	Bundle of text files