PUSS214200

v. 0.3

TimeMate

Software Development Plan

Group 2

Responsible: Project Management Group

Authors: Project Management Group

2021-02-04

Contents

1 Document History

Version	Date	Responsible	Description
0.1	2021-01-25	PG	Document created.
0.2	2021-02-02	PG	Ready for informal review.
0.3	2021-02-04	PG	Fixed gramatical changes and typos.

2 Introduction

This document describes the development model and the development plan for TimeMate. TimeMate and is a system for time reporting and is based on *Baseblock System* and will be developed by students at LTH for the course *ETSF20 Programvaruutveckling för stora projekt*.

3 Terminology

See table 1.

4 Referenced Documents

- Baseblock system, is the base system used in TimeMate.
- The Java Language Specification, describes the code convetion that is followed in this project.
- Project Instruction (Projekthandledningen), contains all details on how the project is structured.
- CML, consists of all configuration units in the project.

5 Development Model

The development model in this project is the waterfall model. This means that the project is divided into four seperate phases where each phase depends on the previous one in a sequential manner. It is thus required that a phase is completed before the next one begins.

In every phase, several documents must be produced (see table 2) and a phase is considered completed only once all documents required in the phase have reached baseline. To reach baseline, all documents of the phase must first pass an informal review, followed by a formal review. This process and its details is further described in section 9.

6 Staff Organisation

The group consists of 21 members divided into 4 main groups, with 2 extra groups consisting of members from several groups.

SG	System management Group
DG	Developer Group
TG	Test Group
PG	Project Management Group
CML	Configuration unit Management List
ECG	Error Control Group
SDP	Software Development Plan
SRS	Software Requirements Specification
SVVS	Software Verification and Validation Specification
SVVI	Software Verification and Validation Instruction
STLDD	Software Top Level Design Document
SDDD	Software Detailed Design Document
SVVR	Software Verification and Validation Report
SSD	System Specification Document
PFR	Project Final Report
IRP	Informal Review Protocol

Table 1: Terminology

6.1 Project Management Group

Members: (2)

- Assar Orpana
- Victor Krook

Responsibilities:

- Coordinating the group effort.
- Creating a project plan and making sure that it is followed.
- Ensuring that every individual has the information they need.
- Authoring the following documents.
  - SDP

  - SSD
  - PFR

6.2 Software Architecture Group

Members: (4)

- David Vilppu
- Ramtin Mosavi
- Filip Sjövall
- Mustafa Elomeiri

Responsibilities

- Designing the software architecture.
- Delegating work to DG and TG.
- Coordinating and co-authoring of the following documents.
  - SRS
  - STLDD
  - SDDD
  - PFR

6.3 Development Group

Members: (10)

- Alexandra Galonja
- Anna Bergvall
- Annelie Sinander
- Hjalmar Janson
- Oscar Johansson
- Sebastian Forslund
- Abd Salam
- Alaa Wahbah
- Alexander Olofsson
- Johan Wulf

Responsibilities

- Designing the UI for the software.
- Developing the architecture designed by SG.
- Developing the code for the system.
- Co-authoring the following documents.
  - SRS
  - STLDD
  - STDDD
  - PFR

6.4 Test Group

Members: (5)

- Alexander Möhle
- Lazar Trpeski
- Max Palmgren
- Malte Wallander
- Anas Abu Al-Soud

Responsibilities

- Testing and validating the functionality of code written by DG.
- Authoring the following documents.
  - SVVS
  - SVVI
- Co-authoring the following documents.
  - SRS
  - PFR

6.5 Group Leaders

Members: (3)

- Oscar Johansson (DG)
- Alexander Möhle (TG)
- David Vilppu (SG)

Responsibilities:

- Coordinating work efforts between the groups.
- Reporting progress to PG.
- Organizing work shifts for their respective groups.

6.6 Error control group

Members: (7)

- Assar Orpana (PG)
- Victor Krook (PG)
- David Vilppu (SG)
- Ramtin Mosavi (SG)
- Filip Sjövall (SG)
- Mustafa Elomeiri (SG)

Responsibilities:

- Managing error reports.
- Handling configuration management.
- Updating baseline documents.

6.7 Others

The forementioned groups make up the team responsible for developing the software. Other stakeholders include:

- **Head of section:** Oversees the project and has direct contact with the client.
- **Client:** Provides a requirement specification.
- **Experts:** Provides auxiliary knowledge in their respective fields.
- **Examiner:** Performs the formal review.

7 Schedule

The project starts in week 3 and ends in week 12, which means there are 9 weeks available. In every phase, several documents shall be produced and table 2 illustrates which documents should be produced for each phase. Figure 1 illustrates the estimated time for each phase and the estimated days scheduled for meetings, informal reviews and formal reviews (see section 9.1).

Phase	**Document**
1	• SRS • SVVS • SDP
2	• SVVI • STLDD
3	• SDDD
4	• SVVR • SSD • PFR

Table 2: Documents to be produced for each phase

Project Group Meetings will be held on Tuesdays and these meetings will be mandatory for each member in the group. These meetings will be used to discuss the project and take general decisions. This is an important step in the development model, to ensure that everyone is on the same page and that the time plan is being kept.

7.1 Estimated Work Load

PG holds a one hour long meeting every week and has scheduled for a two hour time slot every Wednesday during which every group works on their respective task. To supplement this, the groups are expected to manage their own planning meetings and work sessions. Since the workload of each group varies between phases, it seems more fitting to allow groups the flexibility of managing their own time. Group leaders are tasked with making sure that coordination between groups is constant throughout the project. PG also estimates that approximately an hour every week will be spent on general discussions in the discord channel.

Table 3 illustrates the estimated start and end dates for every phase, as well as estimated hours spent per week, per person.

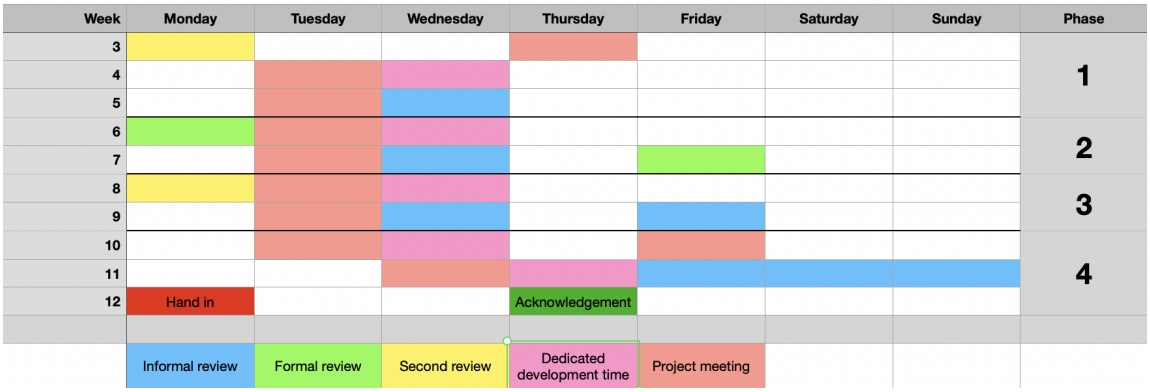

Figure 1: Estimated project schedule

Phase	Start	End	Work days	Estimated hours/week
1	18/1	5/2	15	10
2	08/2	19/2	10	10
3	22/2	5/3	10	10
4	8/3	19/3	10	15

Table 3: Estimated start dates and end for dates the phases

Table 4 illustrates the estimated time spent per activity as well as how often the activity is expected to occur.

8 Standards & Tools

In order to make the development process as easy and straight forward as possible, the project group has agreed on several standards and tools to use.

The standards should be followed by every group member and consists of the following:

- The source code should follow Oracles "The Java Language Specification".
- All comments, commits and pull-requests should be in english.

8.1 Discord

Discord is used as the main communcation tool in the group. A server will be set up and is used for messaging as well as working together and having project meetings. The server consiststs of several voice channels and several text channels, each with its specific purpose, eg *Meetings* or *Developer Group*.

Activity	**Frequency/week**	**Duration (h)**
Project group meeting	1	1
Project group work	1	2
Subgroup work session	2	2*2
Self studies	1	1
Discussions	1	1
Reviews/Expert meetings	1	1
Total hours		**10**

Table 4: Estimated time for each activity per person, per week

8.2 Github & Git

Git is used for collaboration between documents and the project library (see 10.1) and is hosted on Github. Git provides abilities that make certain actions very easy, such as pushing and pulling updates to the working repository and creating pull-requests for merging.

8.3 Eclipse

Eclipse is the primary IDE that is used due to it's huge span of functionality and the groups familarity to it.

8.3.1 Egit

Egit is a plugin for Eclipse provides the tools needed for a git workflow in Eclipse IDE.

8.3.2 Texmaker

Texmaker is a latex editor providing the tools needed to compile and preview latex files.

9 Follow up and Quality Evaluation

To keep the quality of all documents as high as possible during the project, two reviews will be carried out at the end of every phase. One *informal review* followed by one *formal review*

9.1 Informal Review

An informal review is performed within the project group and is meant to catch errors, bugs and mistakes in the documents. This is to ensure as high quality as possible for the formal review. The result of the review is *not* pass or not pass, but its purpose is rather to improve the documents.

An informal review is carried out at least 3 days before the following formal review and the documents that are up for review be reviewed should ready no later than 17:00 the day before the review. PG is responsible for coordinating the meetings as well as documenting them. There are at least three reviewers assigned for each document to be reviewed, and these are assigned by PG. During the review, the reviewers will mention and discuss what they have found and then the reviewees will get an opportunity to reply and have a discussion if needed. A detailed guide for the informal reviews are found in the IRP.

To emphasize, the informal review is *not* ment to criticize or judge, but rather to make the documents as good as possible.

9.2 Formal Review

After the documents have gone through the informal review, they must pass the formal review to reach baseline. During the formal review, an external reviewer is given the documents at least 48 hours in advance so that he or she can prepare. During the review, the entire project group shall be present. The formal review can result in one of the following scenarios:

1. The documents are approved.
2. The documents are approved after certain modification.
3. Modifications must be made followed by a re-review.
4. The documents are *not* approved and must be re-written, followed by a new review.

9.3 Re-Review

In case the documents fail the formal review, it might be necessary to to do a new review after certain modifications have been made.

9.4 Following the Timeplan

If the project is unable to follow timeplan for some reason, the following steps should be:

1. **Identify the problem** - This is the most critical step of the procedure because the cause could be anything from a bad timeplan to members of the team being sick and unable to work.
2. **Explore possible solutions** - Depending on what kind of problem has occured, there might be several different solutions and therefore it important to identify and weigh the different ones.
3. **Pick a solution**.
4. **Take an action**.
5. **Prevent the problem from happening again** - To prevent the problem from happening again, there must be some type of follow-up routine. Depending on what the problem scenario

is, this follow-up could be a personal meeting between PG and the persons involved, or re-creating a timeplan using a different technique. The take-away here is to note the problem, the action and the result of the action.

10 Configuration Management

All changes to units in the CML must follow a certain procedure once the documents have reached baseline. Git and Github are the main tools used to handle configuration management in this project.

10.1 Project Library

The project library consists of two seperate libraries: Document libray and Work library.

10.1.1 Document Library

The document library consists of the following:

- All configuration units that have reached baseline.
- All documents that invloves version control and change management (see section 10.2).
- Protocols from informal reviews (see section 9.1).
- Protocols from formal reviews (see section 9.2).
- Protocols and agendas from Project Group Meetings (see section **??**).

The purpose of this library is that the customer or a reviewer at any point should be able to access the forementioned. This means that this library is initially almost empty, and documents are added as the project proceeds. In the end of phase 4, all units listed in CML can be found in this library.

10.1.2 Work Library

The work library contains all the files that are required during the project. The library is divided into three different branches where each branch has its unique purpose.

- `development`

  This branch is where the all development is done and it is used as a placement for all files related to the project, regardless of their status. Every member in the group has free push access to this branch.

- `review`

  Once the documents in the development branch are ready for an informal review, they are moved into the *review* branch. This requires a pull-request to be made, which must be reviewed and accepted by ECG before it is merged into the branch.

- `master`

  Once the documents pass the formal reviews, they are merged into *master* branch, which consists only of files that have reached baseline. Once new documents are added to this branch, they are also placed in the document library (see section 10.1.1). Note that not all files are copied to the document library, but only the documents specified in the CML.

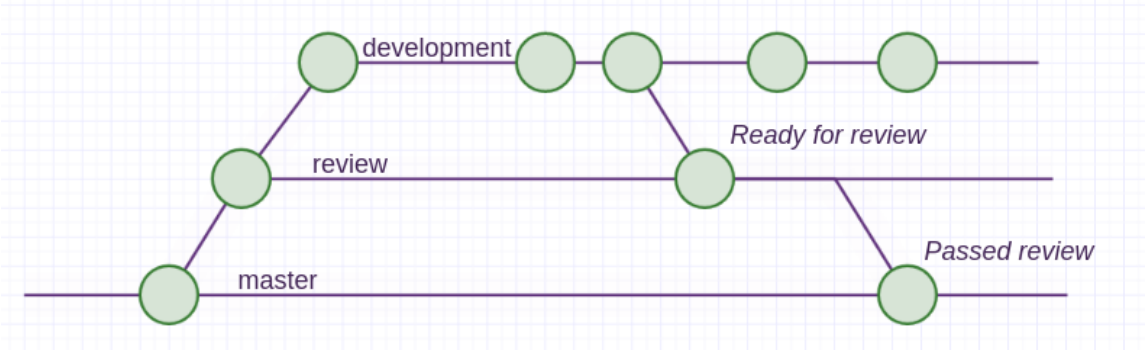

Figure 2: Workflow of the three branches.

10.2 Version Control

Before a configuration unit has reached baseline, changes to the unit may occur freely, without any restrictions. However, once a configuration unit has reached baseline there are several steps that must be taken in order to make changes to it. These steps consist of the following:

1. Creating an error report that states what the error is with the unit, in its current state.
2. The error report is handed to the ECG who then decides if the error is legitimate.
   (a) If the error is deemed legitimate, then they propose a solution and decides who shall be responsible to fix the problem. Once the error is corrected, ECG must approve it and then the version number for the unit must be updated.
   (b) If ECG decides that the error is to be discarded.

The exact details of this procedure can be found in the Project Instruction (see section 4).

10.3 Version Naming & Update

The version number is in the format of $X.Y$ where X refers to the baseline of the unit (in this project, X will be either 0 or 1). Y refers to which version at the given baseline.

How the version number is decided depends on if the unit has reached baseline or not.

Baseline	**How Y is decided.**
Before	Once a unit has been created, the Y starts at 1, which results in version 0.1. When the unit is ready for informal review, the Y is incremented to 2. Then, the Y is incremented once for every change to the unit until it passes a formal review. This means that before reaching baseline, a unit will have a version of at least 0.2.
After	If and only if the procedure described in 10.2, when a document has reached baseline, results in a modification of the configuration unit, the Y is incremented. This means that it is significantly harder to modify a configration unit once is has reached baseline. This should make sense since units that have reached baseline have passed formal reviews, which means that they can be seen as valid and reliable.

Table 5: How the Y in $X.Y$ is decided in the version numbering

11 Rules and Guidelines

To ensure clear communcation and efficiency PG has set up a series of rules and guidelines. If these rules are not followed, there will be a meeting with the involved parties to determine cause and future actions.

- **React to information** - Discord has a feature that allows users to react to messages with an emoji. This is a good way to let the sender know that their message has been acknowledged.
- **Mandatory attendance for group meetings and work sessions** - If a group member is unable to attend a meeting they are expected to contact PG and get the information shared during the meeting another way.
- **Weekly time report in epuss every Sunday 17:00**
- **Official project language is English**
- **Be on time**

12 Risk Analysis

Below are two tables describing the risks, their probability and impact as well as warning signs and solutions

Risk	**Probability**	**Impact**
Poor Communication	High	High
Poor work distribution	High	Medium
Problems with the document library	Medium	Medium
Conflicts within the group	Low	Medium

Table 6: Risks and impacts

Risk	**Warning sign**	**Solution**
Communicationa issues	• Information loss. • People missing meetings. • Confusion around deadlines. • Multiple people working on the same thing.	• Be very clear *where* we communicate, ask questions. • Everyone will pay extra attention to all communication channels in use. • Ensure that the other part of the communcation has recieved and understood the information. • Practice active listening, ask for confirmation.
Poor work distribution	• Some members experience stress whilst others think the workload is light. • Missed deadlines. • Varying quality of work.	• Take resposibility for your own work. • Communicate with your group leader. • If you finish early, ask if anybody needs help. • Ask for help if you are falling behind.
Poor planning	• Low attendance at meetings. • Work is completed long before or after dead lines.	• Plan around deadlines • Consider individual schedules
Problems with the document library	• Work being done twice due to unupdated documents. • Reocurring questions about how it works.	• Hold lectures breaking down the steps • Use cheat sheats and provide additional resources
Conflicts within the group	• Tension during meetings. • Group members are withdrawn out of fear of judgement.	• Put your pride aside and focus on a solution. • Have an understanding for others. • Tell a leader if you feel like you are being treated unfairly. • Ask somebody to mediate in a conflict

Table 7: Warning signs and solutions