HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

COMPUTER SCIENCE

TECAN.

.NET Parameterize-Tool for Pipet-Instruments

Term Project

Department of Computer Science
University of Applied Science Rapperswil

Autumn Term 2011

Author:	Andreas Zollinger
Advisor:	Joas Leemann
Project Partner:	Tecan Schweiz AG
External Co-Examiner:	Prof. Dr. Luc Bläser

Index of Contents

TECAN

Part I - Management

Vereinbarung über Urheber- und Nutzungsrechte

Vereinbarung

1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Studienarbeit **Parametrisierungs-Tool** von Andreas Zollinger unter der Betreuung von Prof. Dr. Luc Bläser (HSR, Rapperswil) geregelt.

2. Urheberrecht

Die Urheberrechte stehen dem Student zu.

3. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl vom Student, von der HSR sowie von Tecan Schweiz AG nach Abschluss der Arbeit verwendet und weiter entwickelt werden.

Rapperswil, den 22.12.2011 .....
Andreas Zollinger

Männedorf, den 21.12.2011 .....
Rainer Kerkmann
Head Instruments Software
Tecan Schweiz AG

Rapperswil, den 22.12.2011 .....
Prof. Dr. Luc Bläser
HSR

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Erklärung

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt habe, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.

Rapperswil, den 23.12.2011 .....

Andreas Zollinger

Aufgabenstellung Studienarbeit für Andreas Zollinger:

.NET Parametrisierungs-Tool für Pipettier-Robotermaschinen

1. Auftraggeber und Betreuer

Diese Studienarbeit findet in Zusammenarbeit mit der *Tecan Schweiz AG* statt.

Ansprechpartner Auftraggeber:

- Joas Leemann, Tecan Schweiz AG , Software Architekt, joas.leemann@tecan.com

Betreuer HSR:

- Prof. Dr. Luc Bläser, Institut für Software, lblaeser@hsr.ch

2. Ausgangslage

Die Tecan AG stellt unter anderem Pipettier-Robotermaschinen her, welche über verschiedene Robotermotoren verfügt. Zum Zeitpunkt der Entwicklung einer neuen Maschinenserie müssen die unterschiedlichen Parameter der Motoren so konfiguriert werden, dass die Robotersteuerung im produktiven Betrieb einwandfrei funktioniert.

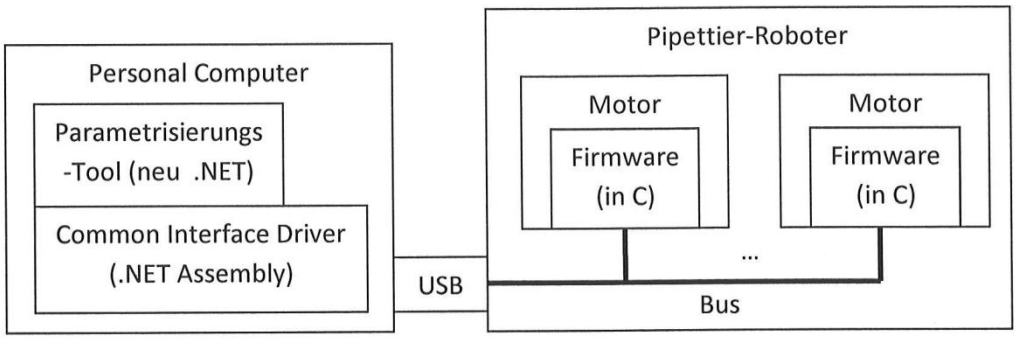
Um die Konfiguration der verschiedenen Parameter (z.B. Kraft, Offsets oder PID-Regler-Parameter) zu bestimmen und zu testen, setzt die Tecan AG bereits ein selbstentwickeltes Parametrisierungs-Tool ein. Das Tool bietet ein Benutzerinterface, in welchem die Parameter eingegeben werden können, die Eingaben an die Maschine übertragen und dort getestet werden und schliesslich die gemessenen Resultate wieder in der Benutzerschnittstelle dargestellt werden.

Das bisherige Parametrisierungs-Tool ist allerdings in verschiedener Hinsicht nicht mehr zufriedenstellend und soll daher überarbeitet werden:

1) Das Tool ist für die Software-Ingenieure schlecht wartungsfähig, weil es in der weniger verbreiteten Programmiersprache LUA geschrieben ist.
2) Die graphische Benutzerschnittstelle benötigt einem kompletten Redesign in Absprache mit den Parametrisierungs-Spezialisten.
3) Das Tool integriert sich schlecht in das existierende .NET Basisframework für den Maschinentreiber von Tecan.

Aus diesem Grund soll im Rahmen dieser Studienarbeit das Parametrisierungs-Tool für Pipettier-Robotermaschinen auf der Technologiebasis von .NET (mit C# und WPF) neu entwickelt werden. Dies umfasst einerseits die Migration der gesamten existierenden Funktionalität in .NET sowie anderseits die vollständige Neugestaltung der Benutzerschnittstelle für die Parametereingabe und die Resultate-Visualisierung. Das Tool verwendet dazu direkt das existierende .NET Treiberframework

(Common Interface Driver) der Firma Tecan, welches direkt als .NET Assembly in das Tool eingebunden werden kann. Die Übergabe der Parameterdaten an den Treiber erfolgt über ein XML-Format.



Situierung des neuen .NET Parametrisierungs-Tool

Auch mit dem neuen Tool bleibt die Parametrisierung eine manuelle Aufgabe, die vom Benutzer in mehreren Versuchsiterationen bestimmt wird. Das Tool könnte jedoch im Rahmen einer weitergehenden Arbeit (ausserhalb dieser Arbeit) dahingehend erweitert werden, dass eine halbautomatische bzw. vollautomatische Iteration zur Parameterbestimmung durchführt.

3. Ziele und Aufgabenstellung

Die Aufgabe dieser Arbeit ist es, dass existierende Parametrisierungs-Tool für die Tecan Pipettier-Robotermaschinen auf Basis von .NET neu zu implementieren sowie die Benutzerschnittstelle in Absprache mit den Benutzern komplett neu zu designen.

Folgende spezifische Ziele werden vorgegeben:

- Aufnahme der genauen Anforderungen für das neue Parametrisierungs-Tool.
- Entwurf der komplett neuen graphischen Benutzerschnittstelle in Absprache mit den Parametrisierungs-Spezialisten (Screenshots, Abläufe, wenn nötig mit Prototyp).
- Neuentwicklung des Parametrisierungs-Tools in C# auf Basis von .NET 4.0, dass die gesamte Funktionalität des existierenden Tools unterstützt sowie das neue GUI in .NET WPF implementiert. Die Funktionalität umfasst insbesondere die Parametereingabe, die Maschinenansteuerung und die Resultate-Darstellung.
- Erstellung eines Online Benutzer-Manuals für das neue Parametrisierungs-Tool.

4. Zur Durchführung

Mit dem HSR-Betreuer finden in der Regel zweiwöchentliche Besprechungen statt. Zusätzliche Besprechungen sind nach Bedarf durch die Studierenden zu veranlassen. Besprechungen mit dem Auftraggeber werden nach Bedarf durchgeführt.

7. Beurteilung

Eine erfolgreiche Studienarbeit erhält 8 ECTS-Punkten (1 ECTS Punkt entspricht einer Arbeitsleistung von ca. 25 bis 30 Stunden). Für die Modulbeschreibung der Studienarbeit siehe https://unterricht.hsr.ch/staticWeb/allModules/19456_M_SAI.html

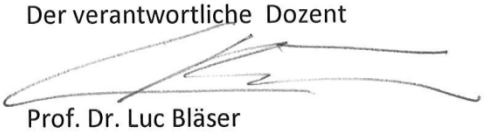
Gesichtspunkt	Gewicht
1. Organisation, Durchführung	1/5
2. Berichte (Abstract, Mgmt Summary, techn. u. persönliche Berichte) sowie Gliederung, Darstellung, Sprache der gesamten Dokumentation	1/5
3. Inhalt *)	3/5

*) Die Unterteilung und Gewichtung von 3. Inhalt wird im Laufe dieser Arbeit festgelegt.

Im Übrigen gelten die Bestimmungen der Abt. Informatik zur Durchführung von Studienarbeiten.

Rapperswil, den 14. September 2011

Der verantwortliche Dozent



Prof. Dr. Luc Bläser
Institut für Software
Hochschule für Technik Rapperswil

Abstract

Tecan Schweiz AG develops various instruments and devices to support the daily tasks in a laboratory. Many of these instruments have motor-driven parts which move items (water tubes, tablets of wells etc.) from one station to another. During engineering, numerous motor control parameters need to be evaluated to program an optimal machine configuration. However, the current machine parameterization tool has two major shortcomings: Firstly, it is not implemented with standard Tecan library, the Tecan Base SDK. Secondly, it is written in the less known scripting language LUA. As this has led to deviations and inconsistencies in the past, Tecan aims to develop a new parameterization tool in C# using the common Tecan Base SDK.

The task of this semester thesis is to determine the requirements of this new parameterization tool and to design and implement a first version of the tool in C# and with the Tecan Base SDK.

In the first phase of the project, all the requirements for the new tool have been analyzed and formally defined. In the second phase, the essential functionality required for the framework has been implemented. During the implementation, it became soon apparent that the primary functionality of the Tecan Base SDK is not yet sufficient to realize the tool. Therefore, the mid-level SDK functionality as well as the low-level SDK communication had to be additionally used to transmit selective firmware commands.

As a result of the project, the first version of the parameterization tool has been completed, supporting the configuration of the essential parameters, the execution of simple moves on the machines and the visualization of performed movements. This shows that the parameterization use cases can be realized and covered with the C# software tool on the top of the Tecan Base SDK. The only - and intended - limitation is that the tool does no longer support ad-hoc adjustments of any scripting code at the engineering time. In the future, the new tool can be extended to cover more functionality and reach the same functional level as the existing LUA program. Moreover, extra supporting tasks may be added in a next version, such as the automatic version control of the different parameter sets.

Management Summary

1 Current Situation

Tecan Schweiz AG develops various instruments to support the daily business in laboratories. Many of these instruments have movable components called devices. For a device to fulfill its task successfully, it has to be configured. These configurations typically contain numerous parameters. Some of the parameters can be easily obtained from the relevant CAD-sheets, whereas other parameters (in particular the motor PID controller values) have to be evaluated on a built-up instrument. Tecan provides a software development kit called Tecan Base SDK, to manage the devices and their configurations. Due to the parallel development of the SDK and the firmware and limited time, the SDK could not create the software to evaluate the different parameters for the devices – despite such a requirement. As a result, the motion control team implemented their own parameterization tool in a language called LUA. This tool has grown in complexity over time and was never replaced by a proper solution using the common Tecan Base SDK. The three relevant disadvantages of this solution are: (1) inconsistencies may arise between this LUA tool and the SDK due to the double engineering, (2) the LUA language is little known in the technical programmer community and is therefore hard to maintain by typical software engineers and (3), this language allows fast ad-hoc adjustments of the code which and can result in dangerous and hard-to-detect mistakes. In order to improve this situation, a new parameterization tool is necessary which establishes a harmonized solution using the same common library and being implemented in a state-of-the-art programming language.

2 Task and Approach

A new parameterization tool shall be developed in the well-known language C# with the use of the Tecan Base SDK to ensure that the same functionality used by the SDK is used to drive the devices. For the members of the motion control team, the change to the new tool should be as easy and as fast as possible. The graphical user interface should look similar to the current firmware team's implementation, such that it is optimized for their needs. Throughout the backend of the software, the Tecan Base SDK should be used as much as possible. The main points in this project are to determine the requirements of the new parameterize tool and to implement a base structure and first version in order to evaluate the feasibility of this new solution.

3 Conclusion

The requirements have been collected and formally defined in the first phase of the project. In the second phase, a base structure of the new tool has been designed and implemented. After a short time in the implementation phase, it became evident that the Tecan Base SDK itself does not provide the entire functionality to fulfill all the requirements of the motion control team. The functionalities and procedures for these special cases had to be additionally implemented. At the end of the project, a first version of the parametrization tool has been completed. Although this version is limited to only cover essential functionality, it proves that the transition to the new functional tool is technically feasible and is a viable approach.

4 Outlook

As for the next steps, the first version of the parameterization tool can be extended to match the full functionality of the current LUA tool. Additional features that could be implemented are:

- user aid to parameterize a motor
- automatic version control for multi users
- the possibility to execute a function on more than one instrument
- the monitoring of more than one device at a time

TECAN

Part II - Documentation

Project Development Plan

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **-**

	Author	Reviewer	Approver
Name	Andreas Zollinger	Luc Bläser	Joas Leemann
Function	Software	Supervisor HSR	Project Leader
Date / Visa			

Table of Contents

1 Introduction

1.1 Purpose of this Plan

This *Product Development Plan* defines the activities, the resources and quality practices for the development of the Parameterize Tool. Between milestones, it is a 'living document' that has to be updated according to the ongoing project work.
The objectives for this product development plan include:

- Defining the development approach and strategy.
- Describing how design control requirements will be met
- Clarifying interacting responsibilities.
- Defining the quality practices ensuring that the product will meet the customers' expectations and that Tecan's quality standards will be met.

To get a total overview, the Product Requirement Document (see Ref. [1]) is needed.

1.2 Scope of this Plan

This Document is written in the Concept Phase and is first released at M2. After first release changes will be documented in the Document Change History.

1.3 Intended Audience

This report is written for the PL, the supervisor from HSR and as general documentation for further usage from the customer.

1.4 Definitions, Acronyms and Abbreviations

Definitions, acronyms and abbreviations can be found in the global table (see Ref. [2])

1.5 References

Ref #	Description
Ref. [1]	Product Requirement Document for Parameterize Tool, 02_ProductRequirementDocument.doc, V1.1
Ref. [2]	Definition, Acronyms and Abbreviations for Parameterize Tool, 90_DefinitionAcronymsAbbreviations.pdf, V1.0
Ref. [3]	Design History File for Parameterize Tool, 92_DesignHistoryFile.xls, V1.0

1.6 Document Change History

Date	*Version*	*Change*	*Author*
2011-10-17	1.0	Release Version 1.0	AnZo
2011-10-31	1.1	• 1.4 Definitions, Acronyms and Abbreviations: Moved into a global list. • 3.8.1 Milestone and Phase Description Added chapter with description of the milestones. • 3.10 Substitution Plan Added chapter about a substitution plan of the current version to the new one.	

1.7 Development Process and Deviations

This Documentation is based according to the following version of the SOP Product Development [1]:

V4.6.

Deviations from that version won't be documented, as the project is a small one and would not be qualified enough in time and complexity for all the needed documents.

1.8 Assumptions and Constraints

Assumptions and Constraints	*Impact if not true*
The project will be further developed after this research project.	Research project would not have an economic output. It would be shift to a feasibility study.
Tecan Base SDK of Tecan Schweiz AG can be used.	A separate communication layer has to be built, make it impossible to maintain the timeline.

2 Product Overview

2.1 Product Description

Existing and newly developed components of the product of this project are indicated in the table below.

Type	*Name*	*Develop / Re-use*
SW	Parameterize tool	New development
SW	Tecan framework for motor drivers	Re-use
SW	Tecan Firmware	Re-use
HW	Existing motors	Re-use
HW	Simulation kit	Re-use

3 Project Overview

3.1 Project Organization

3.1.1 Project Team Definition

The Project Team of this project consists of the members listed in the table below:

Function	*Name*	*Initials*	*Member of Ptm or associated*	*Name of Deputy*
Project Leader (PL)	Joas Leemann	JoLe	N/A	Rainer Kerkmann
Application Software Engineer (SWE)	Andreas Zollinger	AnZo	N/A	N/A
FW Engineer (FWE)	Remo Kälin	ReKa	N/A	Christian Meier
FW Engineer (FWE)	Stefan Ludwig	StLu	N/A	Christian Meier
Supervisor HSR	Luc Bläser	LuBl	N/A	N/A

3.1.2 Trainings

Andreas Zollinger ha to be introduced to the old parameterize tool. The training is organized by Andreas Zollinger himself.

3.1.3 External Development Partners

No external development partners are involved in this project.

3.1.4 VAR Client Projects

No VAR Client exits within this project.

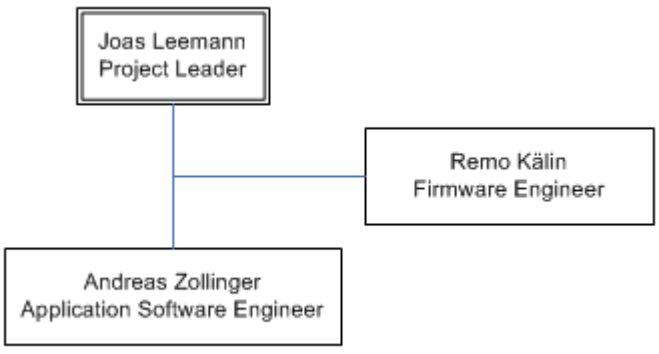
3.1.4.1 Organization at Tecan

No general organization description needed. For internal structure see chapter “3.2 Internal Structure”.

3.1.4.2 Organization at the VAR Client

N/A

3.2 Internal Structure



Joas Leemann fulfills the role as PL. At Tecan Schweiz AG he is working as a software architect.
Andreas Zollinger is the executing engineer, responsible for all the working tasks. Inside Tecan Schweiz AG he is working as a software developer
Remo Kälin provides support considering the firmware. He could be regarded as the costumer. At Tecan Schweiz AG he is the head of the motion control skill group.

3.3 Confidentiality Disclosure Agreements (CDAs)

CDAs concluded for this project are listed in the table below:

Subject of the CDA	*Address of External Party*	*Name and Phone Number of Responsible Person of external Party*	*Name of Responsible Person within this Project at Tecan*	*Signing Date*
N/A	N/A	N/A	N/A	N/A

No CDAs have to be done.

3.4 Additional specific Project Deliverables

The documents listed in the following table are released at the given date in a first version. Further changes are still possible and will be documented inside each document.

Deliverable	*Date available*	*Responsibility*
PRD	2011-10-17	AnZo
SWS	2011-10-31	AnZo
SSD	2011-10-31	AnZo
SDD	2011-10-31	AnZo
SW Configuration Management Plan	2011-10-31	AnZo
SW GUI Design	2011-10-31	AnZo
SW Unit Test Case Plan	2011-12-05	AnZo
SW Unit Test Case Report	2011-12-19	AnZo

3.5 Project Controlling

Control Point	*Method*	*Responsibility*
Requirements Management	Requirements are written at the start of the project in the PRD. Minor changes can be adapted after consulting the PL. Late major changes cannot be considered.	AnZo
Schedule Control	In each week one or more meetings with the PL the progress will be monitored. Within meetings every second week the actual progress will be discussed with the supervisor from HSR.	AnZo
Quality Control and Quality Assurance	Quality checks are made on a regular basis after each milestone together with the PL and the supervisor from HSR.	AnZo

3.6 Reviews and Audits Out of the Development Process

3.6.1 Reviews and additional Assessments

Design Reviews are executed inside the project team by reviewing design documents by experts of the own team and other teams. The review minutes and the respective signatures will document the review.

#	*Item to be reviewed*	*Responsible Person*	*Review Committee*	*Phase*	*Status*
1.	PRD	AnZo	JoLe, LuBl	Concept Phase	V1.1
2.	SWS	AnZo	JoLe, LuBl	Design Input	V1.0
3.	SSD	AnZo	JoLe, LuBl	Design Input	Draft V0.1
4.	CMP	AnZo	JoLe, LuBl	Design Input	V 1.0
5.	SW GUI Design	AnZo	JoLe, LuBl	Design Input	V 1.0
6.	SW Unit Test Case Plan	AnZo	JoLe, LuBl	Validation / Testing	To be done
7.	SW Unit Test Case Report	AnZo	JoLe, LuBl	Validation / Testing	To be done

3.6.2 External Audits

All new external partners must become qualified.

External partner	*Responsible – Project team*	*Responsible – TQM*	*Documentation for qualification*	*Date*
N/A	N/A	N/A	N/A	N/A

No new external partner exists.

3.7 Major Project Risks

Project risks are indicated in the table below for this project:

Risk	*Probability* (low, medium, high)	*Severity (low, medium, high)*	*Mitigation*	*Remarks*	*Identification Date*
loss of AnZo (illness, death or other causes)	very low	high	-	Very unlikely project would be stopped	2011-10-04
HW is not available when needed	medium	medium	Organize early the needed HW and reserve it for the time of the project.	2011-10-10: own simulation kit is organized	2011-10-04
HW has malfunctions during usage time	medium	medium	Clarify who could provide alternate HW and how big the effort would be	Alternate HW should be available 2011-10-10: Members of Motion Control skill group can hand out parts of the simulation kit.	2011-10-04
Project cannot be finished due to lack of time	medium	high	Addition time reserve has to be planed		2011-10-04
Tecan Base SDK has to little functionality to provide all required steps	medium	low	Feasibility study has to be done		2011-10-04

3.8 Planned Milestones and Costs

3.8.1 Milestone and Phase Description

3.8.1.1 Pre M1 Phase

Milestone M1 was before this project. At this point the main idea on the topic was defined. Milestone M1 was also the starting point of this project.

3.8.1.2 Concept Phase (M1 – M2)

The Goal of the concept phase is the creation and definition of product requirements. In addition in this phase initial concepts for the new product are created and the general technical feasibility is estimated.
The Product Requirements Document is the main deliverable of this phase.
A Project Development Plan, this document, is also developed in this phase. The PDP lives on through the whole project.
In the concept phase, the Design History File is opened. The DHF index serves as documentation planning for the software documentation. All additional documents that are generated during the project must be stored in the DHF.
The concept phase is completed with the milestone M2.

3.8.1.3 Design Input Phase (M2 – M3)

The goal of this phase is to define all Design Inputs necessary.
The final Product Requirements Document is the main deliverable of this phase.
During this phase SW Specifications and SW Structure Design are also deliverables.
The design input phase is completed with the milestone M3.

3.8.1.4 Design Output Phase (M3 – M4)

In the design output phase the software is developed on the basis of insights from the input phase. The goal is to realize prototypes and to confirm the matching of these with the specifications through the verification.
The design output phase is completed with the milestone M4. The traceability between requirements and specifications have to be done with a design review.

3.8.1.5 Validation & Testing Phase (M4 – M5)

The goal of the validation and testing phase is the completion of the design verification and validation.
The traceability is completed and verified.
The completeness of the implementation of requirements in specifications and their successful verification is checked.
The validation and testing phase is completed with the milestone M5.
M5 is also the end of this project. After M5 only tasks considering the student project from HSR are done.

3.8.2 Milestone Dates

Milestone dates for this project are indicated in the table below:

Milestone	*Expected Date @M2*	*Expected Date @M3*	*Expected Date @M4*	*Expected Date @RfV*	*Expected Date @M5*	*Actual Date*
M1	N/A	N/A	N/A	N/A	N/A	N/A
M2	N/A	N/A	N/A	N/A	N/A	2011-10-17
M3	2011-10-31	N/A	N/A	N/A	N/A	2011-11-01
M4	2011-11-28	2011-11-28	N/A	N/A	N/A	2011-12-23
RfV	N/A	N/A	N/A	N/A	N/A	N/A
M5	2011-12-19	2011-12-19	-	N/A	N/A	-
M6	N/A	N/A	N/A	N/A	N/A	N/A

3.8.3 Milestone Costs

Forecasted and actual costs of this project are indicated in local currency (man-hours) in the table below:

Cost-to-date (Actual Cost) @ Each Milestone:							
	@M1	*@M2*	*@M3*	*@M4*	*@RfV*	*@M5*	*@M6*
Internal:	0	44.5	97.5	300	N/A		N/A
External:	0	0	0	0	N/A		N/A
Material:	0	0	0	0	N/A		N/A
Total:	0	44.5	97.5	300	N/A		N/A
Cost-to-date (Actual Cost from Last to Actual Milestone) @ Each Milestone:							
	M1	*M2*	*M3*	*M4*	*RfV*	*M5*	*M6*
Internal:	N/A	44.5	53	202.5	N/A		N/A
External:	N/A	0	0	0	N/A		N/A
Material:	N/A	0	0	0	N/A		N/A
Total:	N/A	44.5	53	202.5	N/A		N/A
Forecast (Total Project Cost from BOI to M6) Estimated @ Each Milestone:							
	@M1	*@M2*	*@M3*	*@M4*	*@RfV*	*@M5*	*@M6*
Internal:	240	240	260	-	N/A		N/A
External:	0	0	0	-	N/A		N/A
Material:	0	0	0	-	N/A		N/A
Total:	240	240	260	-	N/A		N/A
Forecast (Project Cost to Next Milestone) Estimated @ Each Milestone:							
	@M1	*@M2*	*@M3*	*@M4*	*@RfV*	*@M5*	*@M6*
Internal:	40	40	80	-	N/A		N/A
External:	0	0	0	-	N/A		N/A
Material:	0	0	0	-	N/A		N/A
Total:	40	40	80	-	N/A		N/A

3.9 Production Quantity

The output will be an executable software package with no quantity requirements.

3.9.1 Project History

This chapter is of informal character only. Its intention is to keep track of the most significant events during the project in order to extract from time to time the lessons learned and to improve the situation.
It specifically contains major changes requested, refused or executed during development phase. This document is also a summary of all changes with the responsible person and time of implementation.

Event / Date	*Decision*	*Consequences*

3.10 Substitution Plan

The new developed Parameterize Tool shall sooner or later substitute the current solution. This substitution has also been planned.
This substitution is not part of this project and so will not be documented.

4 Project Documentation

4.1 Documentation Deliverables for VAR Projects

Document Title	*Purpose*	*Intended Audience*	*Responsibility for:*		
			Writing	*Review*	*Approval*
N/A	N/A	N/A	N/A	N/A	N/A

Not required for this project.

4.2 Design History File (DHF) on the Server

Files are saved on the Tecan network server.
A DHF exists in the documentation dictionary of the project. See Ref. [3]

4.3 DHF in Hardcopy

At the end of the project a hardcopy will be delivered to the PL and the instances of HSR who require a hardcopy.

4.4 Traceability

The traceability will be insured in this project by Andreas Zollinger.
The traceability will be saved in an excel sheet.

5 Appendix

N/A

Product Requirements Document

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **-**

	Author	Reviewer	Approver
Name	Andreas Zollinger	Luc Bläser	Joas Leemann
Function	Software Engineer	Supervisor HSR	Project Leader
Date / Visa			

Table of Contents

1 Introduction

1.1 Purpose

The purpose of the PRD is to clarify who the persons in interests are and what they expect from the outcome of this project.

1.2 Scope

This document is created in the Concept Phase of the project timeline. At M2 (see Ref. [1]) the document has a valid status and goes into version 1.0. After M2 the PRD can be modified with entries in the change history.

1.3 Definitions, Acronyms and Abbreviations

Definitions, acronyms and abbreviations can be found in the global table (see Ref. [3])

1.4 References

Ref #	*Description*
Ref. [1]	Project Development Plan for Parameterize Tool, 01_ProjectDevelopmentPlan.pdf, V1.0
Ref. [2]	http://www.lua.org/
Ref. [3]	Definition, Acronyms and Abbreviations for Parameterize Tool, 90_DefinitionAcronymsAbbreviations.pdf, V1.0

1.5 Document Change History

Date	*Version*	*Change*	*Author*
2011-10-17	1.0	First version of document	AnZo
2011-10-31	1.1	• 1.3 Definitions, Acronyms and Abbreviations list was moved in a global list. • 1.4 References references were updated • 3.3.1.1 Motion Control Member Updated description • 3.3.2 User Characteristics Updated description of physical knowledge • 4.12 Special Moves PRD 12 Special Moves removed • 4.13 Extendable Function Added extendable parameters • New PRD: Current Control Tuning	AnZo

2 Intended Use of the Product

2.1 Background

One of the core businesses from Tecan Schweiz AG are the instruments which are built up of many different electrical motors. At the start of the development of a new instrument the motors have to be configured with different parameters. This step has to be done to ensure the correct function of each motor and the task it has to fulfil.
One motor has many different parameters like force values considering gravitation, offsets or the parameters for a PID controller. All these parameters have do be sought by a special team and a separate software tool with a try and error principle: parameters will be modified and then send to the instrument where a move will be executed and an analysable log will be returned.
The old software has several weaknesses:

- The software does not use the existing Tecan Base SDK written in C#.
- The graphical user interface could be optimized.
- The parameters are saved in a separate place outside the basic driver framework. The data structure is different to the one in the basic driver framework. Exporting the parameters from the old parameterize tool into the basic driver framework is complicated and a source for errors.
- The software is not maintainable by the software team because it is written in LUA, see Ref. [2]. Developers of the motor drivers are not capable of maintain an additional tool written in an uncommon programming language.

2.2 New Solution

The new parameterize tool should replace the current one and cover up the weaknesses.

3 Basic Conditions for Definition of Requirements

3.1 Clients, Customers and other Stakeholders

3.1.1 Clients for the Product

Client and customers don't differ. Look at the following chapter.

3.1.2 Customers for the Product

The Motion Control skill group is the customer of the new parameterize tool. Inside this skill group a team of parameterize specialists need the software to determine parameters for each electric motor and make the results available to the motor driver development team.

3.1.3 Other Stakeholders

The Instrument Software skill group is responsible for the Tecan Base SDK and has the role as a consultant concerning implementation of the software.

3.2 RA Strategy

No RA Strategy is needed for this project.

3.3 Users of the Product

3.3.1 User Groups

3.3.1.1 Motion Control Member

The main task of the Motion Control skill group is to improve, extend and supporting the FW. This team has different sub teams responsible for different tasks. Each sub team uses tools to execute their specific task. One of these tools is the Parameterize Tool.

3.3.1.2 Parameterize Specialist

Inside of the Motion Control skill group a special team with the task to parameterize the motors exist. These team members are the Parameterize Specialists. They have an expert knowledge about the procedure of getting all the different values. They also have experienced data and know the outline of each value depending to each special purpose. They are the ones who work the most with the Parameterize Tool.

3.3.1.3 Instrument Software Member

With the output parameters from the Parameterize Specialist Team a member of the Instrument Software skill group can start to use a motor and develop a new driver. On of the member could get into contact with the Parameterize Tool, but this should be an exception to the rule.

3.3.1.4 System Integrator

System Integrators are responsible for new build up instruments to be ready for the software. At the moment the LUA parameterize tool is one of the used tools to verify the functionality of each drive. They don't parameterize a motor. The tool is just used for motion tests.

3.3.2 User Characteristics

The following characteristics are used to describe the user groups:

Name	*Description*	*Range*
Physical Knowledge	The knowledge how different aspects of physics work and how they have an effect to the motors. Already an average level of knowledge can help a lot for understanding problems. This knowledge could be as example how a motor works in general, differences between different motors, how the gravity comes in play, what is to be mentioned about friction, what about wearing down problems and so on…	little – average – expert
Parameter Knowledge	The understanding what parameters are responsible for each task and which changes must be made to gain a better outcome.	little – average – expert
Attitude towards new technology	Describes how easy a person can be motivated to use new software. The changeover to the new Parameterize Tool can be made smoother if the user is willing to learn new things	conservative – neutral – enthusiastic
Abstraction wanted	How much abstraction a person wants to move a drive. Some people want measure the current who power the motor, others just want to press a button and it should move.	deep – neutral – abstract
HW capability	Has the person the allowance and the knowledge to alter the HW. Does the person know how the HW is built?	No permission - expert

Characteristic	*Motion Control*	*Parameter Specialist*	*Instrument SW*	*System Integrator*
Physical Knowledge	expert	average	little	average
Parameter Knowledge	average	expert	little	Little to average
Attitude towards new technology	neutral	neutral	enthusiastic	neutral
Abstraction wanted	neutral	neutral	abstract	deep
HW capability	no permission	no permission	no permission	expert

3.3.3 User Priority

From the table above the priority of the different user groups could be determined:

3.3.3.1 Key User: Parameterize Specialist

A Parameterize Specialist is the one person who really works with the tool. The new software is made especially for him and so all the requirements should be set to satisfy his needs.

3.3.3.2 Secondary Level User: System Integrator

A System Integrator uses the parameterize tool to verify the correct functionality of every axis during integration with executing the same functions as the Parameter Specialists.

3.3.3.3 Third Level User: Motion Control Member

A Motion Control Member is from the same department as the Parameterize Specialist but doesn't have much to do with the Parameterize Tool. They don't work with the new Parameterize Tool directly but are affected by it. So the inputs from this user will be analysed and implemented if reasonable.

3.3.3.4 Third Level User: Instrument Software Member

An Instrument Software Member has to work with the output of the new tool. They don't work with the new Parameterize Tool directly but are affected by it. So the inputs from this user will be analysed and implemented if reasonable.

3.3.4 User Participation

Parameterize Specialists will be questioned about the old Parameterize Tool to identify what exactly the current problems are. From the outcome of these interviews requirements will be generated. Together with the Parameterize Specialists again this requests will be reviewed and altered.

System Integrators will be questioned witch functionality is used from the LUA parameterize tool. Most likely their needs are covered with the basic functionality of the new parameterize tool, but they will be also involved in the review steps.
Motion Control Members can help establishing requirements with additional information to concrete topics.
Instrument Software Members can help with the requirements about the output parameter file.

3.4 Constraints to the Product

3.4.1 Solution Constraints

The Parameterize Tool shall run on Windows XP operating system and on Windows 7 64Bit operating system.
The parameterize Tool shall run using the Tecan Base SDK only, independent of a project where the Tecan Base SDK is used. Special customization for specific projects could be possible but is out of the boundaries of this project.

3.4.2 Implementation Environment

A normal software development workspace is quit sufficient for implementing the new software. For implementation a simulation box with attached motor is needed.

3.4.3 Interface to other Applications

The Parameterize Tool uses the Tecan Base SDK.

3.4.4 Commercial Off-the-Shelf Packages

N/A

3.4.5 Anticipated Workplace Environment

Parameterize specialists work normally with notebooks to be independent from location. The notebook is connected most of the time to an instrument where a specific drive has to be parameterized. As parameterizing drives happens during development phases the specialist is rarely to never in contact with customers.

3.4.6 Anticipated problems

No physical problems are expected.

3.4.7 Critical Dates and Opportunity Windows

23. October 2011 is the delivery date for the project. At 05:00 pm the work has to be finished.

3.4.8 Other Input for generating Requirements

N/A

3.5 Costs

In the context of costs in this project the currency is always man-hour if not specified different.

3.5.1 Total Costs

As a research project is valued with 8 ECTS points the total costs shall lie around 200-250 man-hours.

3.5.2 Material Costs

This project should not raise additional material costs as the infrastructure and development tools are already available.

3.6 Warranty

This project will not provide any warranty. Additional after work has to be done to gain a status where warranty can be ensured.

3.7 Relevant Trends and Assumptions

The LUA parameterize tool will not be maintainable in the far future because of the static growth. As everyone can extend the LUA parameterize tool according to their needs, the code will be shortly greater disarray as it already is. So a migration to a C# solution has to be done as fast as possible.

3.8 Further Development and Extensibility

After this project further development is not only possible, it is also needed.
A short list with possible additional features:

- Use more than one instrument simultaneous.
- Give the possibility to automate some functions.
- Upgrade the GUI frontend for faster working with consultation the specialists.

3.9 Work Context and Workflow

3.9.1 Work Context

Tecan AG develops medicinal instruments which helps a laboratory technician by taking over some of the tasks in an automatically procedure. During development of this instruments each single motor inside has to be parameterized. This because it is utterly depending on what task with what circumstances a motor is used. So for each single motor-“type of use”-combination different parameters have to be determined. After defining a specific set of parameters for each use case these parameters are used by the firmware and the basic driver framework.

3.9.2 Workflow

The workflow is always the same:

1. Take user input.
2. Move a drive.
3. Present the result to the user.

3.9.3 Working Procedure

To parameterize a new drive following steps have to be done:

1. Organize an HW instrument and verify that a connection to the software is available.
2. Choose an existing motor from the library or make a new one.
3. Choose an existing encoder from the library or make a new one.
4. Gather valid parameters for initializing the drive.
    a. Experience
    b. Rule of Thumb
    c. Try- Error-Principle
5. Gather all other parameters needed
    a. Experience
    b. Rule of Thumb
    c. Try- Error-Principle
6. Save the parameters back to the data storage instance.
7. Export the parameters for the basic driver framework

4 Requirements

ID: Unique ID number
Description: Description of the requirement
Priority 1: If the requirement is not met, the product has strongly reduced market potential, key feature.
Priority 2: If the requirement is not met, the product has moderately reduced market potential.
Priority 3: Requirement with little impact on function, usability or market potential.
Source: Name of the issuer of this requirement
Date: Date of issuing the requirement
Supporting Info: References to additional supporting information
Comment: any further comments on the requirement
Proposal: for implementation / specifications

4.1 Use Tecan Base SDK

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 1*
Description							
The Tecan Base SDK shall be used by the new product. It supports many functions needed by the new parameterize tool. If it does not support a functionality an own implementation can be done.							
Supporting Info							
The old product implements its own functionalities and so a parallel implementation was made to the basic drive framework. Statements from users like: “In LUA it works but with the Tecan Base SDK not” or vice versa were the results.							
Comments							
-							
Proposals							
-							

4.2 Data Storage

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 2*
Description							
A new possibility to storage the data shall be designed. The database should support the possibility to save axes, motors and encoders. Each of this group shall have the possibility to be tagged with hardware versions.							
Supporting Info							
The old LUA implementation does save its data directly in LUA script files. These were imported directly in the running code.							
Comments							
It is possible to use LUA script classes in .NET. Open libraries exist to support this functionality. But sooner or later the migration to a .NET solution has to be done anyway.							
Proposals							
-							

4.3 Read from Data Storage

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 3*
Description							
The tool shall read the parameters from the data storage.							
Supporting Info							
-							
Comments							
-							
Proposals							
-							

4.4 Alter Parameters

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 4*
Description							
Parameter which shall be presented to the user shall have the possibility to be altered by a user. Parameters which are just additional information and cannot or should not be altered should be recognizable as such.							
Supporting Info							
-							
Comments							
Data from axes, motors and encoders shall be able to alter.							
Proposals							
-							

4.5 Write to Data Storage

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 5*
Description							
The tool shall have to write parameters to the data storage.							
Supporting Info							
-							
Comments							
-							
Proposals							
-							

4.6 Initialize Axis

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 6*
Description							
The product shall have the possibility to initialize an axis.							
Supporting Info							
-							
Comments							
Initializing an axis is the first procedure what has to be done. Otherwise no other move can be done.							
Proposals							
-							

4.7 Perform Move

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 7*
Description							
The product shall have the possibility to let the user execute a simple move. Different parameters describing a move shall be selectable by the user. The time for preparing to execute a move shall be short, no complex parameter have to be selected.							
Supporting Info							
-							
Comments							
-							
Proposals							
-							

4.8 Configure FW Log

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 8*
Description							
The product shall be able to configure the firmware logging function. The different parameters for logging shall be selectable by the user.							
Supporting Info							
The FW let the user selected up to three different logging statistics out of about 30. These logging values are recorded during the next move. 100 samples are saved for each selected logging and are returned to the controlling instance.							
Comments							
Different moves, particularly the special moves, need different logging statistics. As the firmware is limited the wished logging statistics have to be chosen beforehand.							
Proposals							
-							

4.9 Present Log

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 9*
Description							
The product shall provide a way to present the executed move and the collected logging data to the user.							
Supporting Info							
-							
Comments							
-							
Proposals							
-							

4.10 Read Encoder Position

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 10*
Description							
The Product shall have the functionality to read the current encoder position. This information shall also be presented to the user.							
Supporting Info							
-							
Comments							
-							
Proposals							
-							

4.11 Send Single FW Command

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 11*
Description							
The product shall have the possibility to send single firmware commands to an axis. The command shall be selectable by the user. Also the axis where the command should be sent to shall be selectable.							
Supporting Info							
-							
Comments							
-							
Proposals							
-							

4.12 Extendable Functionalities and Parameters

Priority:	*2*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 13*
Description							
The product shall have the possibility to be extended by functions and parameters from individual users.							
Supporting Info							
-							
Comments							
One of the biggest advantages of LUA is the extensibility every single user could implement. A part of this flexibility shall be tried to keep.							
Proposals							
-							

4.13 Compare Moves

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*AnZo*	*ID:*	*PRD 14*
Description							
The product shall be able to let the user compare moves.							
Supporting Info							
The log from the firmware shall be saved temporally as long as the user needs the information.							
Comments							
-							
Proposals							
-							

4.14 Optimize Run Cycle

Priority:	*2*	*Date:*	*2011-10-16*	*Source:*	*StLu*	*ID:*	*PRD 15*
Description							
The time needed from changing a parameter, executing the move and getting the results shall be as short as possible.							
Supporting Info							
-							
Comments							
As a big part of the work with this tool will be try and error a fast changeability is essential.							
Proposals							
-							

4.15 Manage Grouping Axis

Priority:	*3*	*Date:*	*2011-10-16*	*Source:*	*StLu*	*ID:*	*PRD 16*
Description							
The new product shall have the possibility group up single axis to devices and one or more devices to one instrument.							
Supporting Info							
-							
Comments							
-							
Proposals							
-							

4.16 Handling Errors

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*StLu*	*ID:*	*PRD 17*
Description							
The product shall support a robust error handling.							
Supporting Info							
Many tests provoke firmware errors. These errors have to be cached and presented to the user on an intelligent way.							
Comments							
-							
Proposals							
-							

4.17 Current Control Tuning

Priority:	*1*	*Date:*	*2011-10-16*	*Source:*	*ReKa*	*ID:*	*PRD 18*
Description							
The product shall be able to configure current control and so a possibility to tune these parameters has to be given.							
Supporting Info							
-							
Comments							
-							
Proposals							
-							

Software Specification

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **-**

	Author	Reviewer	Approver
Name	Andreas Zollinger	Luc Bläser	Joas Leemann
Function	Software Engineer	Supervisor HSR	Project Leader
Date / Visa			

Table of Contents

1 Introduction

1.1 Purpose of the Document

The SWS describes the external behavior of the application or subsystem identified. It also describes nonfunctional specifications, design constraints and other factors necessary to provide a complete and comprehensive description of the specification for the software.

1.2 Scope

This document is generated during the “Design Input” phase and is first released at M3. Changes after the first release will be documented in the document history.

1.3 Definitions, Acronyms and Abbreviations

Definitions, acronyms and abbreviations can be found in the global table (see Ref. [3])

1.4 Referenced Documents

Ref #	*Description*
Ref. [1]	http://www.lua.org/
Ref. [2]	Use Case Specification for Parameterize Tool, 04_UseCaseSpecification.pdf, V1.0
Ref. [3]	Definition, Acronyms and Abbreviations for Parameterize Tool, 90_DefinitionAcronymsAbbreviations.pdf, V1.0
Ref. [4]	Traceability Matrix for Parameterize Tool, 91_TraceabilityMatrix.pdf, V1.0
Ref. [5]	06_SW Graphical User Interface Design for Parameterize Tool, 06_SWGUIDesign.pdf, V1.0

1.5 Document Change History

Date	*Version*	*Change*	*Author*
2011-12-18	1.0	Initial Version	AnZo

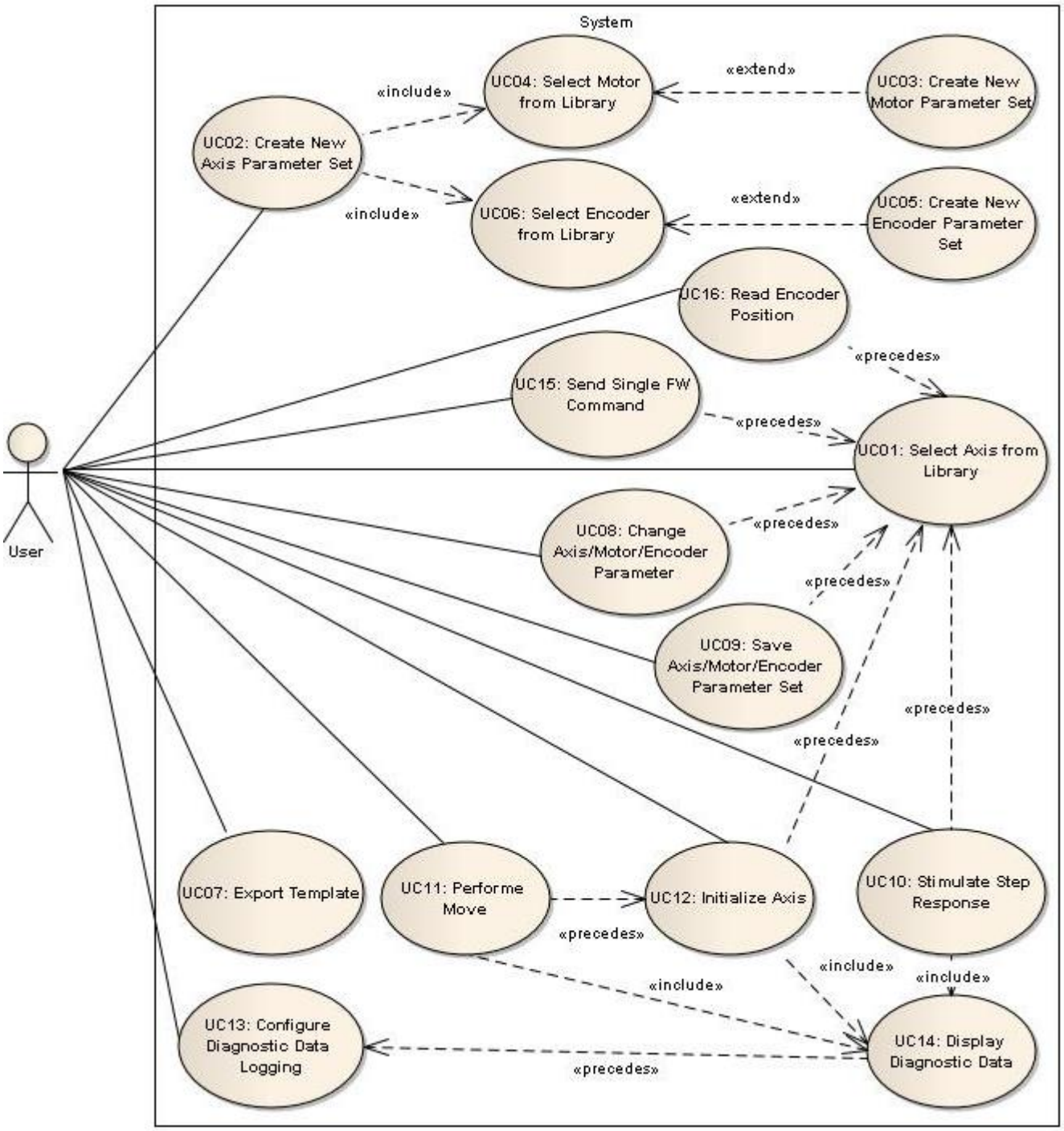
2 Overall Description

One of the core businesses from Tecan Schweiz AG are the instruments which are built up of many different electrical motors. At the start of the development of a new instrument the motors have to be configured with different parameters. This step has to be done to ensure the correct function of each motor and the task it has to fulfill.
One motor has many different parameters like force values considering gravitation, offsets or the parameters for a PID controller. All these parameters have do be sought by a special team and a separate software tool with a try and error principle: parameters will be modified and then send to the instrument where a move will be executed and an analyzable log will be returned.
The old software has several weaknesses:

- The software does not use the existing Tecan Base SDK written in C#.
- The graphical user interface could be optimized.
- The parameters are saved in a separate place outside the Tecan Base SDK. The data structure is different to the one in the basic driver framework. Exporting the parameters from the old parameterize tool into the basic driver framework is complicated and a source for errors.
- The software is not maintainable by the software team because it is written in LUA, see Ref. [1]. Developers of the motor drivers are not capable of maintain an additional tool written in an uncommon programming language.

2.1 Use Case Model Overview

In this chapter a short overview of the use cases is shown. Fully described use cases are defined in the UCS (see Ref. [2]) document.



2.1.1 Use Cases

ID: Unique ID number
Name: Name of the use case
Description: A short description what the use case does in general.

ID	*Name*	*Description*
UC01	Select Axis from Library	Let the user select a predefined axis parameter set from a library.
UC02	Create New Axis Parameter Set	Let the user create a new axis parameter set. This includes selecting a motor parameter set and encoder parameter set.
UC03	Create New Motor Parameter Set	Let the user create a new motor parameter set.
UC04	Select Motor from Library	Let the user select a predefined motor parameter set.
UC05	Create New Encoder Parameter Set	Let the user create a new motor parameter set.
UC06	Select Encoder from Library	Let the user select a predefined encoder parameter set.
UC07	Export Template	The user can select multiple axes. These axes will be exported into one single template file who can be directly used by the Tecan Base SDK.
UC08	Change Axis/Motor/Encoder Parameter	The user can change the parameters of axes, motors and encoders on the selected axis.
UC09	Save Axis/Motor/Encoder Parameter Set	The user can save his changes in the parameter sets on the selected axis back to the data storage.
UC10	Stimulate Step Response	The user can stimulate a step response on the selected axis with selectable parameters.
UC11	Perform Move	The user can execute a move with on the selected axis selectable parameters.
UC12	Initialize Axis	The user can initialize the selected axis.
UC13	Configure Diagnostic Data Logging	The user can configure the diagnostic data parameters used in the next logging that will be executed.
UC14	Display Diagnostic Data	The diagnostic data will be presented to the user after movements of a motor.
UC15	Send Single FW Command	The user can send a single FW command to the selected axis.
UC16	Read Encoder Position	The user can read the encoder position of the selected encoder.

2.1.2 Actors

Name: Name of the actor.
Description: Description of the actor.

Name	*Description*
User	As the only user of the system he is the single operator and has the full accessibility to all direct usable use cases.

2.2 Physical Characteristics

Name	*Specification*
Code language	.net 4.0, C# 4.0
Data file storage	XML
Operating system	Windows XP & Windows 7 (32 & 64 Bit)

3 Specifications

ID: Unique ID number
Name: Name of specification
Description: Detailed description about the specification
Priority: 1 – Must have
2 – Nice to have
3 – Additional nice to have, not in this project

3.1 Functionality

3.1.1 Organization of Data Entities

Name	Organisation of Axis	*Priority*	1	*ID*	SWS 1
Description					
An axis contains one motor and one or two encoders.					

Name	Organisation of Device	*Priority*	1	*ID*	SWS 2
Description					
One or more axis can be grouped to a device.					

Name	Organisation of Instrument	*Priority*	1	*ID*	SWS 3
Description					
One or more devices can be grouped to an instrument					

3.1.2 Select

Name	Select Axis	*Priority*	1	*ID*	SWS 4
Description					
The user has the possibility to open a single axis and its parameter set.					

Name	Select Encoder	*Priority*	1	*ID*	SWS 5
Description					
The user can assign one to three encoders to an axis.					

Name	Select Motor	*Priority*	1	*ID*	SWS 6
Description					
The user can assign one motor to an axis.					

3.1.3 Create

Name	Create New Motor	*Priority*	1	*ID*	SWS 7
Description					
The user can create a new motor parameter set.					

Name	Create New Encoder	*Priority*	1	*ID*	SWS 8
Description					
The user can create a new encoder.					

Name	Create New Axis	*Priority*	1	*ID*	SWS 9
Description					
The user can create a new axis.					

Name	Create New Instrument	*Priority*	1	*ID*	SWS 10
Description					
The user can create a new Instrument. Inside the instrument the user has to define at least one device.					

3.1.4 Open

Name	Open Axis	*Priority*	1	*ID*	SWS 11
Description					
The user can open an axis parameter set.					

Name	Open Instrument	*Priority*	1	*ID*	SWS 12
Description					
The user can open an instrument parameter set.					

3.1.5 Alter

Name	Alter Encoder Parameter	*Priority*	1	*ID*	SWS 13
Description					
The user can change encoder parameters.					

Name	Alter Motor Parameter	*Priority*	1	*ID*	SWS 14
Description					
The user can change motor parameters.					

Name	Alter Axis Parameter	*Priority*	1	*ID*	SWS 15
Description					
The user can change axis parameters.					

3.1.6 Save

Name	Save Encoder Parameter	*Priority*	1	*ID*	SWS 16
Description					
The user can save all changed encoder parameters.					

Name	Save Other Encoder Version	*Priority*	1	*ID*	SWS 17
Description					
The user can save an encoder under another version.					

Name	Save Motor Parameter	*Priority*	1	*ID*	SWS 18
Description					
The user can save all changed motor parameters.					

Name	Save Other Motor Version	*Priority*	1	*ID*	SWS 19
Description					
The user can save a motor under another version.					

Name	Save Axis Parameter	*Priority*	1	*ID*	SWS 20
Description					
The user can save all changed axis parameters.					

Name	Save Other Axis Version	*Priority*	1	*ID*	SWS 21
Description					
The user can save an axis under another version.					

3.1.7 Movement

3.1.7.1 Move

Name	Perform Move	*Priority*	1	*ID*	SWS 22
Description					
The user can perform a move. Selectable parameters are distance, speed, acceleration and deceleration.					

3.1.7.2 Step Response

Name	Stimulate Step Response	*Priority*	1	*ID*	SWS 23
Description					
The user can trigger a step response. Different step types are selectable. Amplitude and duration are selectable.					

3.1.7.3 Initialization

Name	Hard Reset an Axis	*Priority*	1	*ID*	SWS 24
Description					
The user can hard reset an axis.					

Name	Initialize Motion Control	*Priority*	1	*ID*	SWS 25
Description					
The user can initialize the motion control firmware elements.					

Name	Search Encoder Index	*Priority*	1	*ID*	SWS 26
Description					
The user can start the procedure of searching the encoder index.					

Name	Search Mechanical Stop	*Priority*	1	*ID*	SWS 27
Description					
The user can start a procedure to search a mechanical stop of an axis.					

Name	Initialize a Device	*Priority*	3	*ID*	SWS 28
Description					
The user can initialize all axes assigned to selected device.					

Name	Initialize an Instrument	*Priority*	3	*ID*	SWS 29
Description					
The user can initialize all devices assigned to a selected.					

3.1.8 Diagnostic Data Log

Name	Log Points	*Priority*	1	*ID*	SWS 30
Description					
All three Logging Points per axis of the FW are usable. Also the Log points of an axis which will not be moved can be set and logged.					

Name	Configure Diagnostic Data Logging	*Priority*	1	*ID*	SWS 31
Description					
The user can configure the diagnostic logging functionality of the FW.					

Name	Memorize Diagnostic Data Log	*Priority*	1	*ID*	SWS 32
Description					
The system memorizes the diagnostic data log.					

Name	Display Memorized Diagnostic Data Log	*Priority*	1	*ID*	SWS 33
Description					
The system presents the log to the user. Output can be a graphical plot or a text plot.					

Name	Select Plot for Log	*Priority*	2	*ID*	SWS 34
Description					
The user can select to display a memorized log on a specific display.					

Name	More than 3 Log Display	*Priority*	3	*ID*	SWS 35
Description					
The user can select to compare memorized logs on an extra display.					

3.1.9 Others

Name	Send Single FW Command	*Priority*	1	*ID*	SWS 36
Description					
The user can send single FW commands. Module ID is selectable by the user.					

Name	Read Encoder Position	*Priority*	1	*ID*	SWS 37
Description					
The current position of the encoder can be read.					

Name	Torque Offset	*Priority*	3	*ID*	SWS 38
Description					
The torque offset of an axis can be parameterized.					

Name	Ripple Moment Compensation	*Priority*	3	*ID*	SWS 39
Description					
Ripple moment compensation can be parameterized.					

Name	Read FW Version	*Priority*	2	*ID*	SWS 40
Description					
The user can read the FW version of a connected axis.					

Name	Hold Switch Delay	*Priority*	3	*ID*	SWS 41
Description					
The hold switch delay can be parameterized.					

Name	Move Axis Several Times	*Priority*	2	*ID*	SWS 42
Description					
The user can move an axis the defined distance several times on bidirectional ways.					

Name	Sweeping Moves	*Priority*	3	*ID*	SWS 43
Description					
The user can define sweeping moves and execute them.					

3.2 Safety

No specifications concerning safety exist in this project.

3.3 Software driven Alarms and Warnings

Name	Reliability	*Priority*	1	*ID*	SWS 44
Description					
As one point of the software is to provoke FW errors, the software has to be resistant to exceptions.					

3.4 Operator Messages

No operator messages have be specified.

3.5 Security

No specifications concerning security exist in this project.

3.6 Usability

Name	Usability	*Priority*	2	*ID*	SWS 45
Description					
The software is design for an expert. New beginners have to be trained in the software. The training should not take longer than two hours direct teaching. This presumes that the beginner is not new the overall topic but also already knew what his goal with this tool will be. The usability for beginners can be neglected to a maintainable degree if the productivity of an expert user can be increased.					

3.7 Reliability

No reliability specifications have to be done for this project.

3.8 Performance

Name	Time to Create New Axis	*Priority*	2	*ID*	SWS 46
Description					
The Time to create a new axis is less than 1 minute.					

Name	Time to Handle Parameters	*Priority*	2	*ID*	SWS 47
Description					
After getting the last diagnostic data log, the time to change one parameter and execute a move again does not exceed 10 seconds for a trained user.					

3.9 Installation, Methods of Operation and Maintenance

Name	Installation	*Priority*	2	*ID*	SWS 48
Description					
The software is runnable inside the Tecan Base SDK without extra time costs.					

3.10 Attributes

No attributes have to be specified for this project.

3.11 Inputs and Outputs

Name	Export Template from Instrument	*Priority*	1	*ID*	SWS 49
Description					
The user can select an axis to generate a Tecan Base SDK template file.					

Name	Export Templates from Axis	*Priority*	1	*ID*	SWS 50
Description					
The user can select an axis to generate a Tecan Base SDK template file.					

3.11.1 Data definition and Database specifications

Name	Data Storage	*Priority*	1	*ID*	SWS 51
Description					
The data is stored in a file system. Used format is XML.					

3.12 Design Constraints

The new product uses the Tecan Base SDK. So some design constrains are given:

- SW language: .net 4.0 C# 4.0
- WPF
- Development Tool: Microsoft Visual Studio 2010

3.13 Online User Documentation and Help System

The base of an online user documentation will be done.
A detailed documentation will not be part in this project.

3.14 Applicable Hardware

The software should be usable by all hardware that can be used with the Tecan Base SDK.

3.15 Purchased Components (SOUP Components)

Non SOUP components need to be specified for this project.

3.16 Interfaces

3.16.1 User Interfaces

Name	User Interface	*Priority*	1	*ID*	SWS 52
Description					
A user interface will be implemented using WPF.					

3.16.2 Hardware Interfaces

The HW interfaces are handled by the Tecan Base SDK.

3.16.3 Software Interfaces

Name	Tecan Base SDK	*Priority*	1	*ID*	SWS 53
Description					
The software uses the Tecan Base SDK and its functionality.					

3.16.4 Communications Interfaces

Communication to the FW will be handled by the Tecan Base SDK.

3.17 Licensing

No licensing specifications have to be done for this project.

3.18 Legal, Copyright and Other Notices

The developed product is property of Tecan Schweiz AG.

3.19 Applicable Standards

Applicable standards concerning the GUI are handled in the Graphical User Interface Design. See Ref. [5]

4 Traceability

The traceability is handled inside the global traceability matrix file. (See Ref. [4])

5 Appendix

n/a

Use Case Specification

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **-**

	Author	Reviewer	Approver
Name	Andreas Zollinger	Luc Bläser	Joas Leemann
Function	Software Engineer	Supervisor HSR	Project Leader
Date / Visa			

Table of Contents

1 Introduction

1.1 Purpose

This document describes the software specification if the Parameterize. It defines the ways how the tool should be realized.

1.2 Scope

This document belongs to the Parameterize Tool. This document is generated during the “Design Input” phase and is first released at M3. Changes after the first release will be documented in the document history.

1.3 Definitions, Acronyms and Abbreviations

Definitions, acronyms and abbreviations can be found in the global table (see Ref. [1])

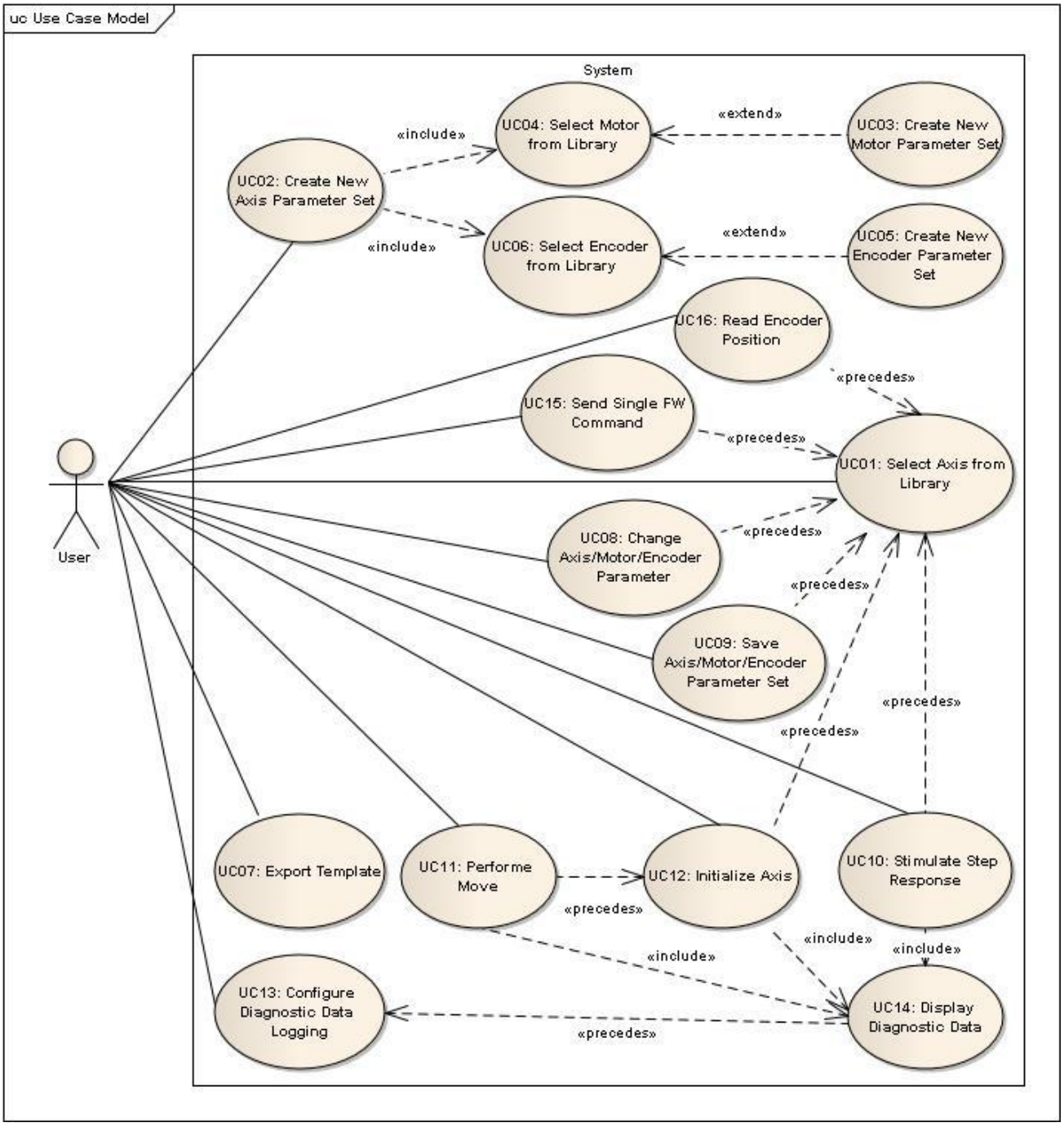
1.4 References

Ref #	*Description*
Ref. [1]	Definition, Acronyms and Abbreviations for Parameterize Tool, 90_DefinitionAcronymsAbbreviations.pdf, V1.0
Ref. [2]	Traceability Matrix for Parameterize Tool, 91_TraceabilityMatrix.pdf, V1.0

1.5 Document Change History

Date	*Version*	*Change*	*Author*
2011-12-18	1.0	Initial version	AnZo

2 Use Cases Overview



2.1 Use Cases

ID: Unique ID number
Name: Name of the use case
Description: A short description what the use case does in general.

ID	*Name*	*Description*
UC01	Select Axis from Library	Let the user select a predefined axis parameter set from a library.
UC02	Create New Axis Parameter Set	Let the user create a new axis parameter set. This includes selecting a motor parameter set and encoder parameter set.
UC03	Create New Motor Parameter Set	Let the user create a new motor parameter set.
UC04	Select Motor from Library	Let the user select a predefined motor parameter set.
UC05	Create New Encoder Parameter Set	Let the user create a new motor parameter set.
UC06	Select Encoder from Library	Let the user select a predefined encoder parameter set.
UC07	Export Template	The user can select multiple axes. These axes will be exported into one single template file who can be directly used by the Tecan Base SDK.
UC08	Change Axis/Motor/Encoder Parameter	The user can change the parameters of axes, motors and encoders on the selected axis.
UC09	Save Axis/Motor/Encoder Parameter Set	The user can save his changes in the parameter sets on the selected axis back to the data storage.
UC10	Stimulate Step Response	The user can stimulate a step response on the selected axis with selectable parameters.
UC11	Perform Move	The user can execute a move with on the selected axis selectable parameters.
UC12	Initialize Axis	The user can initialize the selected axis.
UC13	Configure Diagnostic Data Logging	The user can configure the diagnostic data parameters used in the next logging that will be executed.
UC14	Display Diagnostic Data	The diagnostic data will be presented to the user after movements of a motor.
UC15	Send Single FW Command	The user can send a single FW command to the selected axis.
UC16	Read Encoder Position	The user can read the encoder position of the selected encoder.

2.2 Actors

Name: Name of the actor
Description: Description of the actor.

Name	*Description*
User	He is the only user of the system and the single operator and has so the full accessibility to all direct usable use cases.

3 Detail Use Case

3.1 Description of the Following Descriptions

This section contains the explanations of the different points the following chapters are about.

3.1.1 Characteristic Information

This chapter defines information that pertains to this particular use case. Each piece of information is important in understanding the purpose behind the Use Case.

3.1.2 Main Success Scenario

This Scenario describes the steps that are taken from trigger event to goal completion when everything works without failure. It also describes any required cleanup that is done after the goal has been reached. The steps are listed in a table.

3.1.3 Scenario Extensions

This is a listing of how each step in the Main Success Scenario can be extended. Another way to think of this is how can things go wrong. The extensions are followed until either the Main Success Scenario is rejoined or the Failed End Condition is met. The Step refers to the Failed Step in the Main Success Scenario and has a letter associated with it. I.E. if Step 3 fails the Extension Step is 3_A.

3.1.4 Scenario Variations

If a variation can occur in how a step is performed it will be listed here.

3.1.5 Related Information

The following table gives the information that is related to the Use Case.

3.2 UC01: Select Axis from Library

3.2.1 Characteristic Information

Goal In Context:	The user has to select an axis from the library. The parameter set from the selected axis is loaded.
Scope:	System
Level:	Task
Pre-Condition:	-
Success End Condition:	All parameters from the selected axis are loaded to the software.
Failed End Condition:	No parameters could be loaded.
Primary Actor:	User
Trigger Event:	User action

3.2.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User wants to select an axis parameter set.
2	System	System presents all available axis parameter set.
3	User	User selects an axis parameter set.
4	System	System presents the axis parameter set to the user.

3.2.3 Scenario Extensions

Step	*Condition*	*Action Description*
2_A	No available axes exist	2_A1: Systems informs the user that no axis parameter sets are available.
		2_A2_A1: User cancels 2_A2_A2: No parameters could be loaded.
		2_A2_B1: User selects to create a new axis parameter set. 2_A2_B2: “UC02: Create New Axis Parameter Set” will be executed. 2_A2_B3: Scenario continuous normally with step 4.
3_A	The axis the user wants to select is not available	3_A1_A1: Users stops the selecting procedure. 3_A1_A2: No axis parameter set could be loaded.
		3_A1_B1: User selects to create a new axis parameter set. 3_A1_B2: “UC02: Create New Axis Parameter Set” will be executed. 3_A1_B3: Scenario continuous normally with step 4.

3.2.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.2.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	On every startup and during run multiple times.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.3 UC02: Create New Axis Parameter Set

3.3.1 Characteristic Information

Goal In Context:	The user can create a new axis parameter set that also contains the link to a motor and one or two encoder.
Scope:	System
Level:	Task
Pre-Condition:	-
Success End Condition:	A new axis parameter set could be generated.
Failed End Condition:	No new axis parameter set could be generated.
Primary Actor:	User
Trigger Event:	User action

3.3.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User wants to create an axis parameter set.
2	-	UC04: Select Motor from Library
3	-	UC06: Select Encoder from Library
4	System	System creates a new axis parameter set with the selected encoder and motor.
5	System	System presents the new created axis parameter set to the user.

3.3.3 Scenario Extensions

Step	*Condition*	*Action Description*
-	-	-

3.3.4 Scenario Variations

Step	*Variable*	*Possible Variations*
3_A	User wants to select two encoders	3_A1: UC06: Select Encoder from Library 3_A2: UC06: Select Encoder from Library 3_A3: Scenario continuous normally with step 4.
3_B	User wants to select three encoders	3_B1: UC06: Select Encoder from Library 3_B2: UC06: Select Encoder from Library 3_B3: UC06: Select Encoder from Library 3_B4: Scenario continuous normally with step 4.

3.3.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Variable. From several times a day to once in a fortnight.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.4 UC03: Create New Motor Parameter Set

3.4.1 Characteristic Information

Goal In Context:	The user can create a new motor parameter set.
Scope:	System
Level:	Task
Pre-Condition:	-
Success End Condition:	A new motor parameter set could be generated.
Failed End Condition:	No new motor parameter set could be generated.
Primary Actor:	User
Trigger Event:	User action triggered inside UC04: Select Motor from Library.

3.4.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User wants to create a motor parameter set.
2	System	System creates a motor parameter set.

3.4.3 Scenario Extensions

Step	*Condition*	*Action Description*
-	-	-

3.4.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.4.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Variable. From several times a day to once in a month.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Through UC04: Select Motor from Library
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.5 UC04: Select Motor from Library

3.5.1 Characteristic Information

Goal In Context:	The user can create a new motor parameter set.
Scope:	System
Level:	Task
Pre-Condition:	-
Success End Condition:	A new motor parameter set could be generated.
Failed End Condition:	No new motor parameter set could be generated.
Primary Actor:	User
Trigger Event:	User action triggered inside UC02: Create New Axis Parameter Set.

3.5.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User wants to select a new motor parameter set.
2	System	System presents all available motor parameter set.
3	User	User selects a motor parameter set.
4	System	System returns the selected parameter set.

3.5.3 Scenario Extensions

Step	*Condition*	*Action Description*
2_A	No available motor parameter sets exist	2_A1: Systems informs the user that no motor parameter sets are available.
		2_A2_A1: User cancels. 2_A2_A2: No parameter set could be created.
		2_A2_B1: User selects to create a new motor parameter set. 2_A2_B2: “UC03: Create New Motor Parameter Set” will be executed. 2_A2_B3: New created motor parameter set will be loaded. 2_A2_B4: Scenario continuous normally with step 4.
3_A	The motor parameter set the user wants to select is not available	3_A1_A1: Users stops the selecting procedure. 3_A1_A2: No motor parameter set could be created.
		3_A1_B1: User selects to create a new motor parameter set. 3_A1_B2: “UC03: Create New Motor Parameter Set” will be executed. 3_A1_B3: New created motor parameter set will be loaded. 3_A1_B4: Scenario continuous normally with step 4.

3.5.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.5.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Variable. From several times a day to once in a month.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Through UC02: Create New Axis Parameter Set
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.6 UC05: Create New Encoder Parameter Set

3.6.1 Characteristic Information

Goal In Context:	The user can create a new encoder parameter set.
Scope:	System
Level:	Task
Pre-Condition:	-
Success End Condition:	A new encoder parameter set could be generated.
Failed End Condition:	No new encoder parameter set could be generated.
Primary Actor:	User
Trigger Event:	User action triggered inside UC06: Select Encoder from Library.

3.6.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User wants to create a new encoder parameter set.
2	System	System creates an encoder parameter set.

3.6.3 Scenario Extensions

Step	*Condition*	*Action Description*
-	-	-

3.6.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.6.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Variable. From several times a day to once in a month.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Through UC06: Select Encoder from Library
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.7 UC06: Select Encoder from Library

3.7.1 Characteristic Information

Goal In Context:	The user can create a new encoder parameter set.
Scope:	System
Level:	Task
Pre-Condition:	-
Success End Condition:	A new encoder parameter set could be generated.
Failed End Condition:	No new encoder parameter set could be generated.
Primary Actor:	User
Trigger Event:	User action triggered inside UC02: Create New Axis Parameter Set.

3.7.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User wants to select a new encoder parameter set.
2	System	System presents all available encoder parameter set.
3	User	User selects an encoder parameter set.
4	System	System returns the selected parameter set.

3.7.3 Scenario Extensions

Step	*Condition*	*Action Description*
2_A	No available encoder parameter sets exist	2_A1: Systems informs the user that no encoder parameter sets are available.
		2_A2_A1: User cancels. 2_A2_A2: No parameter set could be created.
		2_A2_B1: User selects to create a new encoder parameter set. 2_A2_B2: "UC05: Create New Encoder Parameter Set" will be executed. 2_A2_B3: New created encoder parameter set will be loaded. 2_A2_B4: Scenario continuous normally with step 4.
3_A	The encoder parameter set the user wants to select is not available	3_A1_A1: Users stops the selecting procedure. 3_A1_A2: No encoder parameter set could be created.
		3_A1_B1: User selects to create a new encoder parameter set. 3_A1_B2: "UC05: Create New Encoder Parameter Set" will be executed. 3_A1_B3: New created encoder parameter set will be loaded. 3_A1_B4: Scenario continuous normally with step 4.

3.7.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.7.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Variable. From several times a day to once in a month.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Through UC02: Create New Axis Parameter Set
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.8 UC07: Export Template

3.8.1 Characteristic Information

Goal In Context:	The user can export a list from selected axis in a template file used by the Tecan Base SDK for driver configuration.
Scope:	System
Level:	Task
Pre-Condition:	-
Success End Condition:	All parameters from the selected axes could be exported
Failed End Condition:	Export had failed.
Primary Actor:	User
Trigger Event:	User action

3.8.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User tells the system to start the export procedure.
2	System	System presents all available axis parameter sets.
3	User	User selects axis parameter sets.
5	System	System exports the selected axis parameter sets to a template file.

3.8.3 Scenario Extensions

Step	*Condition*	*Action Description*
2_A	No available axes exist	2_A1: Systems informs the user that no axis parameter sets are available. 2_A2: Scenario stops, export could not be done.
3_A	An axis the user wants to select is not available	3_A1: Users stops the selecting procedure. 3_A2: Scenario stops, export could not be done.

3.8.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.8.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Variable. From several times a day to once a month.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.9 UC08: Change Axis/Motor/Encoder Parameter

3.9.1 Characteristic Information

Goal In Context:	The user can change the parameters of an axis, an encoder or motor parameter set.
Scope:	System
Level:	Task
Pre-Condition:	UC01: Select Axis from Library
Success End Condition:	Parameter is changed
Failed End Condition:	Parameter is not changed
Primary Actor:	User
Trigger Event:	User action

3.9.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User changes a parameter.

3.9.3 Scenario Extensions

Step	*Condition*	*Action Description*
-	-	-

3.9.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.9.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Extremely often.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.10 UC09: Save Axis/Motor/Encoder Parameter Set

3.10.1 Characteristic Information

Goal In Context:	The user can save the parameters of an axis, an encoder or motor parameter set.
Scope:	System
Level:	Task
Pre-Condition:	UC01: Select Axis from Library
Success End Condition:	Parameters are saved
Failed End Condition:	Save procedure has failed.
Primary Actor:	User
Trigger Event:	User action

3.10.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User tells the system to save an axis parameter set.
2	System	System save the parameter set.

3.10.3 Scenario Extensions

Step	*Condition*	*Action Description*
-	-	-

3.10.4 Scenario Variations

Step	*Variable*	*Possible Variations*
1_A	User wants to save an encoder parameter set.	1_A1: User tells the system to save an encoder parameter set.
1_B	User wants to save a motor parameter set.	1_B1: User tells the system to save a motor parameter set.

3.10.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Often.
Super Use Case:	
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.11 UC10: Stimulate Step Response

3.11.1 Characteristic Information

Goal In Context:	The user can stimulate a step response to determine the right motor current parameters.
Scope:	System
Level:	Task
Pre-Condition:	UC01: Select Axis from Library
Success End Condition:	Step response was executed and diagnostic data is presented to the user.
Failed End Condition:	Step response could not be executed. Diagnostic data could not be presented to the user.
Primary Actor:	User
Trigger Event:	User action

3.11.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User tells the system to stimulate a step response.
2	System	System stimulates a step response.
3	System	System waits for axis to finish.
4	System	UC14: Display Diagnostic Data

3.11.3 Scenario Extensions

Step	*Condition*	*Action Description*
3_A	Tecan Base SDK informs of a timeout during waiting	3_A1: The axis does notify a timeout. 3_A2: Step response could not be executed.
3_B	Tecan Base SDK informs of an error from FW	3_B1: The axis does notify an error. 3_B2: Step response could not be executed.

3.11.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.11.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Normal frequency.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.12 UC11: Perform Move

3.12.1 Characteristic Information

Goal In Context:	The user can move an axis to determine the correctness of the parameters.
Scope:	System
Level:	Task
Pre-Condition:	UC01: Select Axis from Library UC12: Initialize Axis
Success End Condition:	Move was executed and diagnostic data is presented to the user.
Failed End Condition:	Move could not be executed. Diagnostic data could not be presented to the user.
Primary Actor:	User
Trigger Event:	User action

3.12.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User tells the system to move an axis.
2	System	System moves an axis.
3	System	System waits for axis to finish.
4	System	UC14: Display Diagnostic Data

3.12.3 Scenario Extensions

Step	*Condition*	*Action Description*
3_A	Tecan Base SDK informs of a timeout during waiting	3_A1: The axis does notify a timeout. 3_A2: Move could not be executed.
3_B	Tecan Base SDK informs of an error from FW	3_B1: The axis does notify an error. 3_B2: Move could not be executed.

3.12.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.12.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Normal frequency.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.13 UC12: Initialize Axis

3.13.1 Characteristic Information

Goal In Context:	The user can initialize an axis to determine the correctness of the parameters and to be able to move with the axis afterwards.
Scope:	System
Level:	Task
Pre-Condition:	UC01: Select Axis from Library
Success End Condition:	Move was executed and diagnostic data is presented to the user.
Failed End Condition:	Move could not be executed. Diagnostic data could not be presented to the user.
Primary Actor:	User
Trigger Event:	User action

3.13.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User tells the system to initialize an axis.
2	System	System initialize an axis.
3	System	System waits for axis to finish.
4	System	UC14: Display Diagnostic Data

3.13.3 Scenario Extensions

Step	*Condition*	*Action Description*
3_A	Tecan Base SDK informs of a timeout during waiting	3_A1: The axis does notify a timeout. 3_A2: Move could not be executed.
3_B	Tecan Base SDK informs of an error from FW	3_B1: The axis does notify an error. 3_B2: Move could not be executed.

3.13.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.13.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Normal frequency.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.14 UC13: Configure Diagnostic Data Logging

3.14.1 Characteristic Information

Goal In Context:	The user can configure the diagnostic data logger to ensure the logging procedure for the next move is optimized. Different types of moves need different types of logging options.
Scope:	System
Level:	Task
Pre-Condition:	-
Success End Condition:	Configuration of the diagnostic data logger is made.
Failed End Condition:	User could not configure the diagnostic data logger.
Primary Actor:	User
Trigger Event:	User action

3.14.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User changes a configuration parameter.

3.14.3 Scenario Extensions

Step	*Condition*	*Action Description*
-	-	-

3.14.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.14.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Once to fife time a day.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.15 UC14: Display Diagnostic Data

3.15.1 Characteristic Information

Goal In Context:	The system reads the diagnostic data log and presents the outcome to the user.
Scope:	System
Level:	Task
Pre-Condition:	-
Success End Condition:	Diagnostic data log is presented to the user successfully.
Failed End Condition:	Diagnostic data log could not present to the user.
Primary Actor:	System
Trigger Event:	System triggers inside UC10: Stimulate Step Response, UC11: Perform Move and UC12: Initialize Axis.

3.15.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	System	System is told to display the diagnostic data.
2	System	System reads the diagnostic data log.
3	System	System interprets the diagnostic data log.
4	System	System presents the interpreted data log to the user.

3.15.3 Scenario Extensions

Step	*Condition*	*Action Description*
2_A	No log was made.	2_A1: System informs the user that no diagnostic data log was made. 2_A2: Diagnostic data log could not present to the user.
3_A	Log could not be interpreted	3_A1: System informs that the diagnostic data log was not valid and so not could be interpreted. 3_A2: Diagnostic data log could not present to the user.

3.15.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.15.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Extremely often.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Through: UC10: Stimulate Step Response, UC11: Perform Move and UC12: Initialize Axis.
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.16 UC15: Send Single FW Command

3.16.1 Characteristic Information

Goal In Context:	The user can send a single command to the FW.
Scope:	System
Level:	Task
Pre-Condition:	UC01: Select Axis from Library
Success End Condition:	Single FW command could be send.
Failed End Condition:	Single FW command could not be send.
Primary Actor:	User
Trigger Event:	User action

3.16.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User defines a single FW command.
2	User	User sends the single FW command.
3	System	System sends the single FW command to the FW.

3.16.3 Scenario Extensions

Step	*Condition*	*Action Description*
2_A	User did not defines a single FW command.	2_A1: System informs the user to define a single FW command.

3.16.4 Scenario Variations

Step	*Variable*	*Possible Variations*
1_A	User wants to send to another axis	1_A1: User chooses another axis number. 1_A2: user defines a single FW command. 1_A3: Scenario continuous normally with step 2.

3.16.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Several times during a day.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling.
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

3.17 UC16: Read Encoder Position

3.17.1 Characteristic Information

Goal In Context:	The user wants to see the current encoder position to ensure that the encoder and the communication line to the encoder works probably.
Scope:	System
Level:	Task
Pre-Condition:	UC01: Select Axis from Library
Success End Condition:	Encoder position could be presented to the user.
Failed End Condition:	Encoder position could not be presented to the user.
Primary Actor:	User

Trigger Event:	User action

3.17.2 Main Success Scenario

Step	*Actor*	*Action Description*
1	User	User tells the system to read the encoder position.
2	System	System reads the encoder position.
3	System	System presents the encoder position to the user.

3.17.3 Scenario Extensions

Step	*Condition*	*Action Description*
2_A	Encoder position could not be read.	2_A1: The system informs the user that the encoder position could not be read.

3.17.4 Scenario Variations

Step	*Variable*	*Possible Variations*
-	-	-

3.17.5 Related Information

Schedule:	M4
Priority:	Must
Performance Target:	-
Frequency:	Several times during a day.
Super Use Case:	-
Sub Use Case(s):	-
Channel To Primary Actor:	Direct calling.
Secondary Actor(s):	-
Channel(s) To Secondary Actor(s):	-

4 Remarks

The following table provides insight to additional comments and remarks.

RemarkID	Remark
-	-

5 Open Issues

The following table provides insight to any unresolved problems or questions. These are the things that seem to apply but could not be fit into this use case on this pass.

Issue ID	Issue Description
-	-

6 Traceability

The traceability is handled inside the global traceability matrix file. (See Ref. [2])

Software Structure Design (Architecture)

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **-**

	Author	Reviewer	Approver
Name	Andreas Zollinger	Luc Bläser	Joas Leemann
Function	Software Engineer	Supervisor HSR	Project Leader
Date / Visa			

Table of Contents

1 Introduction

1.1 Purpose

This document describes the software architecture of the Parameterize Tool. It comprises the structural design of the SW components and design decisions and conventions.

1.2 Scope

This document belongs to the Parameterize Tool. This tool is needed to determine various parameters for moving motors used by medicinal instruments.
This document is generated during the “Design Input” phase and is first released at M3. Changes after the first release will be documented in the document history.

1.3 Definitions, Acronyms and Abbreviations

Definitions, acronyms and abbreviation can be found in the global table (see Ref. [1])

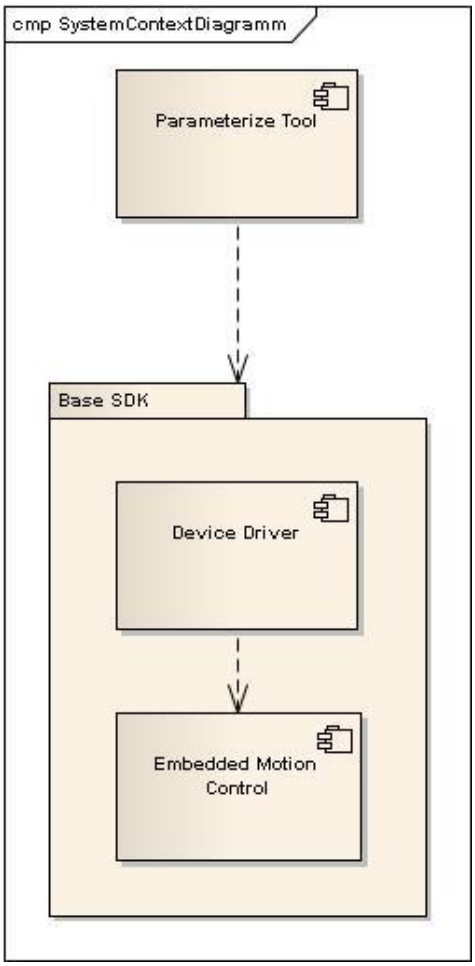
1.4 References

Ref #	*Description*
Ref. [1]	Definition, Acronyms and Abbreviations for Parameterize Tool, 90_DefinitionAcronymsAbbreviations.pdf, V1.0
Ref. [2]	Use Case Specification for Parameterize Tool, 04_UseCaseSpecification.pdf, V1.0
Ref. [3]	00143_05238 C# Programming Guidelines Version 1.0
Ref. [4]	00143_05028 SSD Tecan Base SDK Version 1.0

1.5 Document Change History

Date	*Version*	*Change*	*Author*
2011-12-19	1.0	Initial Version	AnZo

2 System Context



2.1 Parameterize Tool

The Parameterize Tool component provides the functionality to manage motor parameters. It is responsible for the different functionalities needed for determine these parameters.

2.2 Base SDK

The Tecan Base SDK provides drivers and hosts a driver manager used to implement application software to control one or more Tecan instruments. To do that, these drivers communicate to embedded firmware modules. This document does not describe this package in detail, because it is mentioned just for giving context to the Parameterize Tool.

2.2.1 Device Driver

The device driver controls devices which are specific to the application. This document does not describe this package in detail, because it is mentioned just for giving context to the Parameterize Tool.

2.2.2 Embedded Motion Control

The Motion Control is a firmware module which handles the communication between the PC and the Tecan instrument on the embedded site. It further contains a firmware module which controls motion drives. This document does not describe this package in detail, because it is mentioned just for giving context to the Parameterize Tool.

3 Functional View

3.1 Functional Context

The Parameterize Tool provides the functionality to manage and determine parameters for motors.

3.2 Functional Areas

3.2.1 Managing Parameter Sets

The Parameterize Tool can handle parameter sets. Axes can be saved and loaded from a file system.

3.2.2 Diagnostic Data Logging

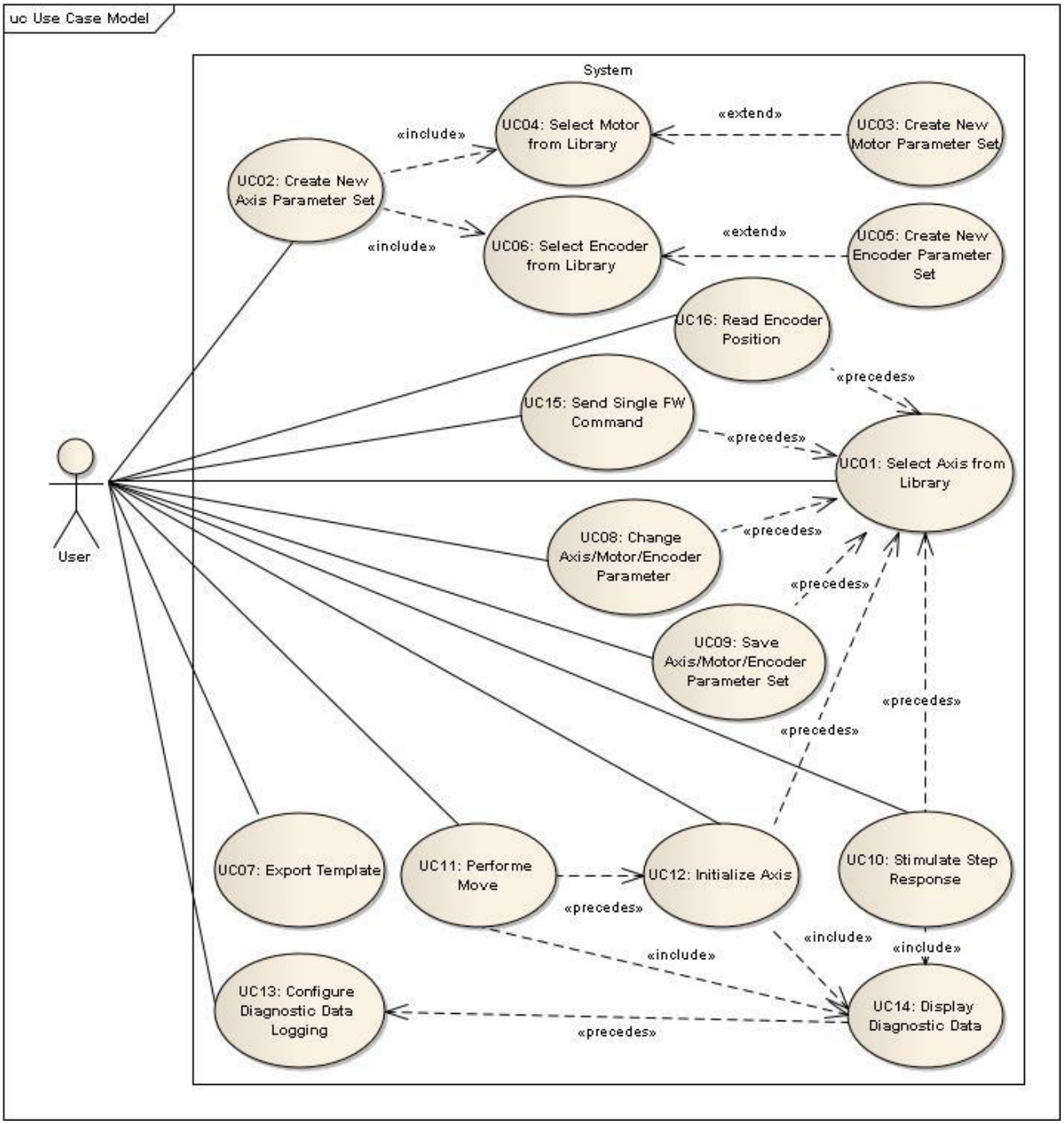
It provides the functionality to control the FW logging function and display the logging data to the user.

3.2.3 Determine Parameters

It manages the determine process of finding good working parameters for each single axis.

3.3 Use Cases Overview

In this chapter a short overview of the use cases is shown. Fully described use cases are defined in the UCS (see Ref. [2]) document.



3.3.1 Use Cases

ID: Unique ID number
Name: Name of the use case
Description: A short description what the use case does in general.

ID	*Name*	*Description*
UC01	Select Axis from Library	Let the user select a predefined axis parameter set from a library.
UC02	Create New Axis Parameter Set	Let the user create a new axis parameter set. This includes selecting a motor parameter set and encoder parameter set.
UC03	Create New Motor Parameter Set	Let the user create a new motor parameter set.
UC04	Select Motor from Library	Let the user select a predefined motor parameter set.
UC05	Create New Encoder Parameter Set	Let the user create a new motor parameter set.
UC06	Select Encoder from Library	Let the user select a predefined encoder parameter set.
UC07	Export Template	The user can select multiple axes. These axes will be exported into one single template file that can be directly used by the Tecan Base SDK.
UC08	Change Axis/Motor/Encoder Parameter	The user can change the parameters of axes, motors and encoders on the selected axis.
UC09	Save Axis/Motor/Encoder Parameter Set	The user can save his changes in the parameter sets on the selected axis back to the data storage.
UC10	Stimulate Step Response	The user can stimulate a step response on the selected axis with selectable parameters.
UC11	Perform Move	The user can execute a move with on the selected axis selectable parameters.
UC12	Initialize Axis	The user can initialize the selected axis.
UC13	Configure Diagnostic Data Logging	The user can configure the diagnostic data parameters used in the next logging that will be executed.
UC14	Display Diagnostic Data	The diagnostic data will be presented to the user after movements of a motor.
UC15	Send Single FW Command	The user can send a single FW command to the selected axis.
UC16	Read Encoder Position	The user can read the encoder position of the selected encoder.

3.3.2 Actors

Name: Name of the actor.
Description: Description of the actor.

Name	*Description*
User	As the only user of the system he is the single operator and has the full accessibility to all direct usable use cases.

4 Process View

The Parameterize Tool as a whole doesn't represent or implement a process. There are processes in functionalities of the tool described in the following chapters.

4.1 Any Type of Move

Independent what type of move should be executed, move and motor parameter have to be sent to the embedded motion controller. Then the diagnostic data logger configuration has to be sent and the logger has to be activated. Afterwards the move can be started. The tool has to wait for the move to be completed. Afterwards it can read the log data and present it to the user.

4.2 Export Parameters to a Template

The user can select an axis that should be exported to a template file usable by the Tecan Base SDK. After the user has selected the wished axis, the tool searches for the motor parameter set referenced in the axis parameter set. Also one or two encoder parameter sets will be get by the tool from the data storage. With this combined parameter sets a template file will be serialized. This file can further be used from the Tecan Base SDK to read in all parameters and write them down to the embedded motion controller.

5 Non-functional View

5.1 Runtime non-functional requirements

5.1.1 Monitoring

Monitoring is not one a big feature of the Parameterize Tool. The log function from the Tecan Base SDK will be used as this library supports all the needed functionalities concerning monitoring.

5.1.2 Auditability

The object instance saving format is xml. With a simple but strong base structure the auditability is ensured.

5.1.3 Scalability

The architecture of the program is designed that additional functionality can added on flexible way. The program can also use the wide range of scalability functions from the Tecan Base SDK,

5.1.4 Management

One or more axes can be grouped up to a device. Axes and devices can have different versions.

5.2 Non-runtime non-functional requirements

5.2.1 Flexibility

The program uses a fine granularity of assemblies and interfaces to be as flexible as possible. Functions can be modified or enhanced by providing changed or newly created modules.
The Tecan Base SDK supports this functionality innately.

5.2.2 Localization

No localization is done. The default language is English. The tool and all outputs are in English.

5.2.3 Extensibility

The Parameterize Tool is designed to give the possibility to modify and extend the functionality provided by it. This is important as new parameters and so new functionality will come up during the lifetime of the tool.

6 Architectural Constrains

6.1 IDE

The Parameterize Tool is developed with Microsoft Visual Studio 2010.

6.2 Coding-Languages

C# / .NET 4.0 for PC based SW will be used.

6.3 Coding-style-guide

The coding-style-guide for C# is specified in an external document. See Ref. [3].

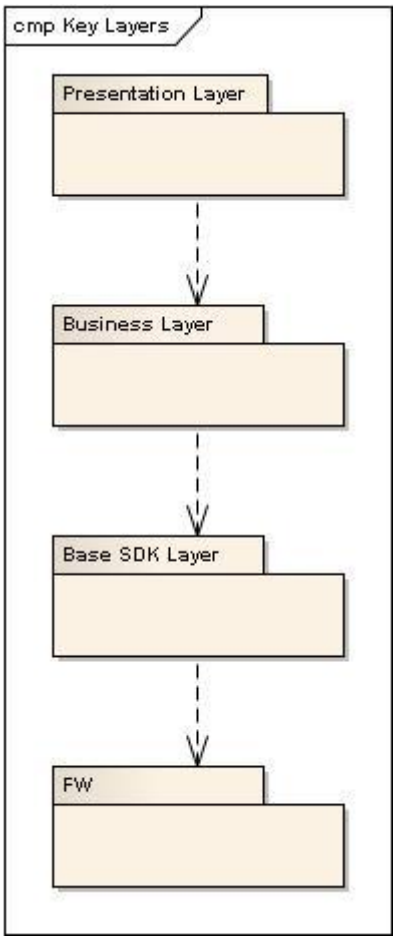
6.4 WPF/GUI

WPF is used for GUI.

7 Architectural Principles

7.1 Key Layers

The whole system the Parameterize Tool uses can be divided into following layers:



Generally the flow of logic goes from top to down. This means that lower layers have no knowledge of or do not reference to layers above them except if the higher layers register themselves to lower layers. The abstraction goes from bottom to top. The lower layers implement smaller components and the higher layers combine these components to modules.

7.1.1 Presentation Layer

The presentation layer implements the GUI suitable for the needs of determine parameters. Data is refined to be presented to the user.

7.1.2 Business Layer

The business layer contains all the logic how to handle the data structure, the logging function and the different functionalities. The layer further handles the communication with the Base SDK Layer.

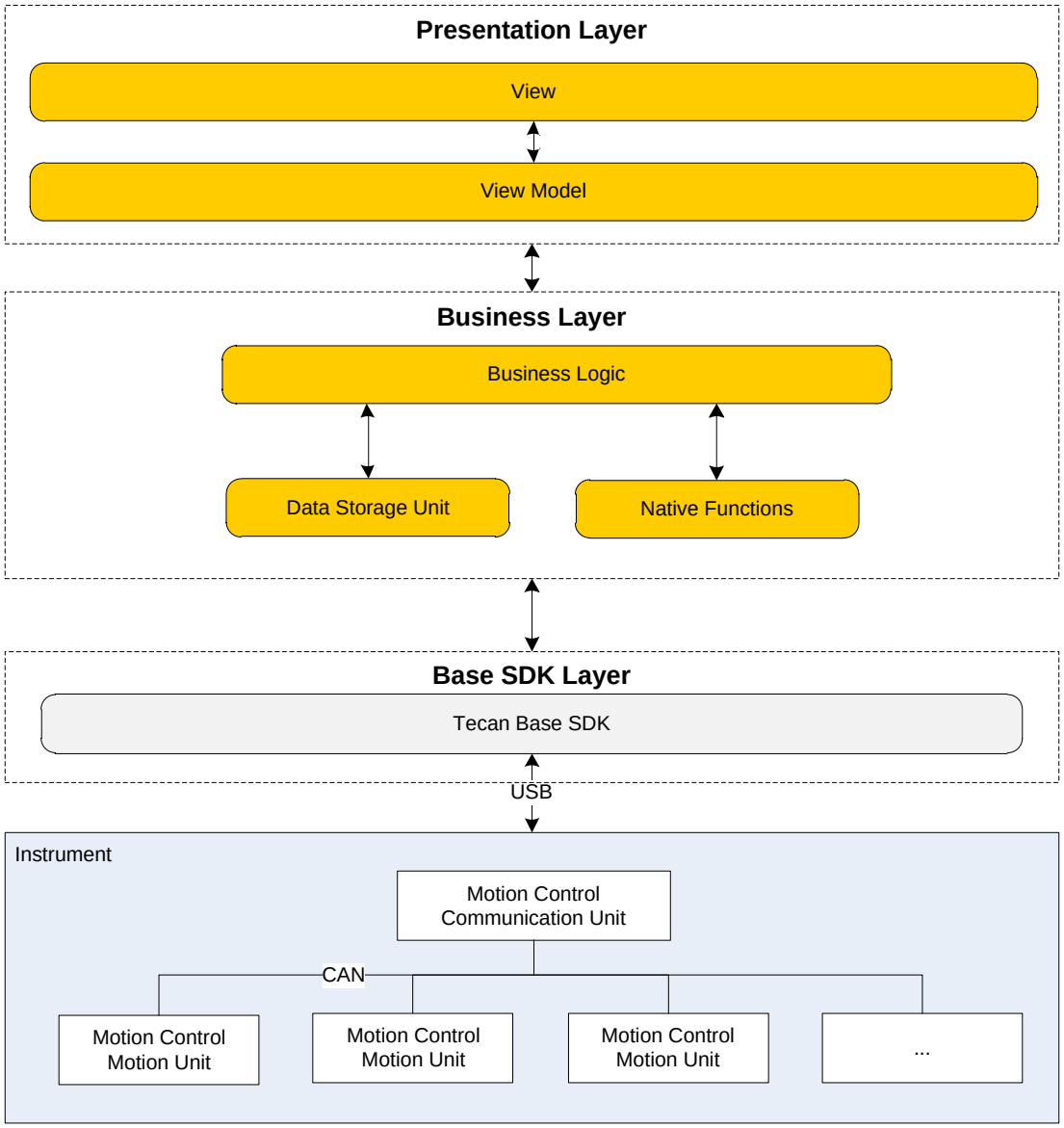
7.1.3 Base SDK

The layer of the Tecan Base SDK contains drivers and services which are used to implement specific applications. This document does not describe this layer in detail, because it is mentioned just for giving context to this tool.

7.1.4 FW

The firmware layer contains the software modules which run on the embedded processors in the instrument. This document does not describe this layer in detail, because it is mentioned just for giving context to this tool.

7.2 Layer Details



7.2.1 Presentation Layer

7.2.1.1 View

View Package contains all the GUI logic and presentation data.

7.2.1.2 View Model

View Model knows how to convert data from the business layer that the view layer can present them. The view model also interprets user actions and forwards the input to the business layer.

7.2.2 Business Layer

Business layer is responsible for the business logic. It is the knowledge holder and knows what function under what circumstances have to be executed.
Every component of the Business Layer can access the classes and so the functionalities of the Tecan Base SDK.

7.2.2.1 Business Logic

The Business logic unit provides the View Model the needed resources and interprets the user input. It further has the main knowledge on the processes in the tool.

7.2.2.2 Data Storage Unit

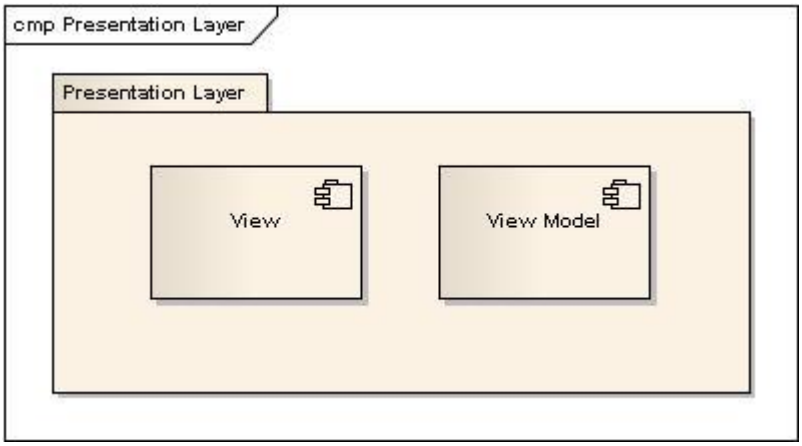
This unit is responsible for handling the storage of the parameter sets. The business layer gets and sets parameters through the functionalities from this layer.
Everywhere where it is reasonable the Tecan Base SDK data classes should be used.

7.2.2.3 Native Functions

The native functions unit provides the business logic the functionalities which the Tecan Base SDK cannot provide.

8 Logical View

8.1 Presentation Layer



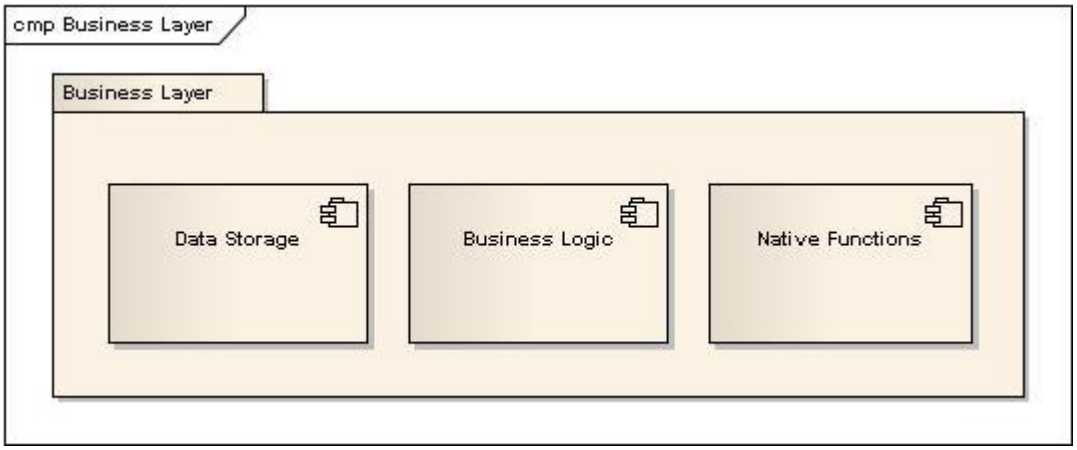
8.1.1 View

This component package contains all views, templates and user controls. The view is connected to the view model.

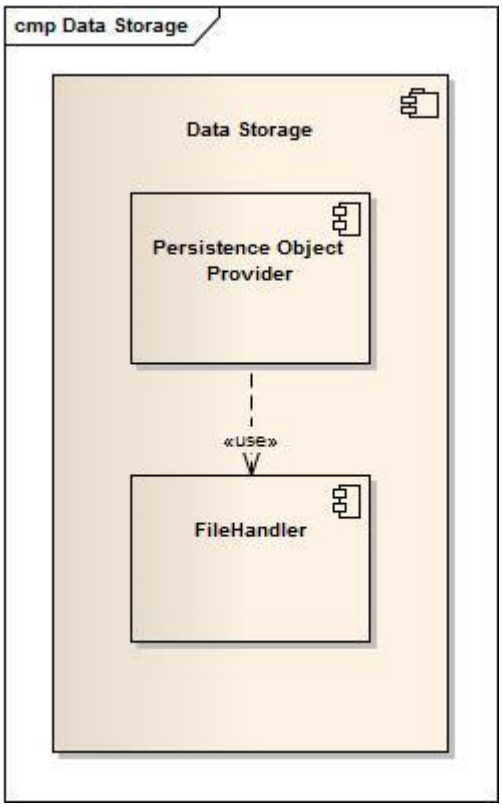
8.1.2 View Model

The view model connects the view with the business logic. It provides the transmit the user input to the business logic and take the data objects and convert them to small pieces representable in the view.

8.2 Business Layer



8.2.1 Data Storage



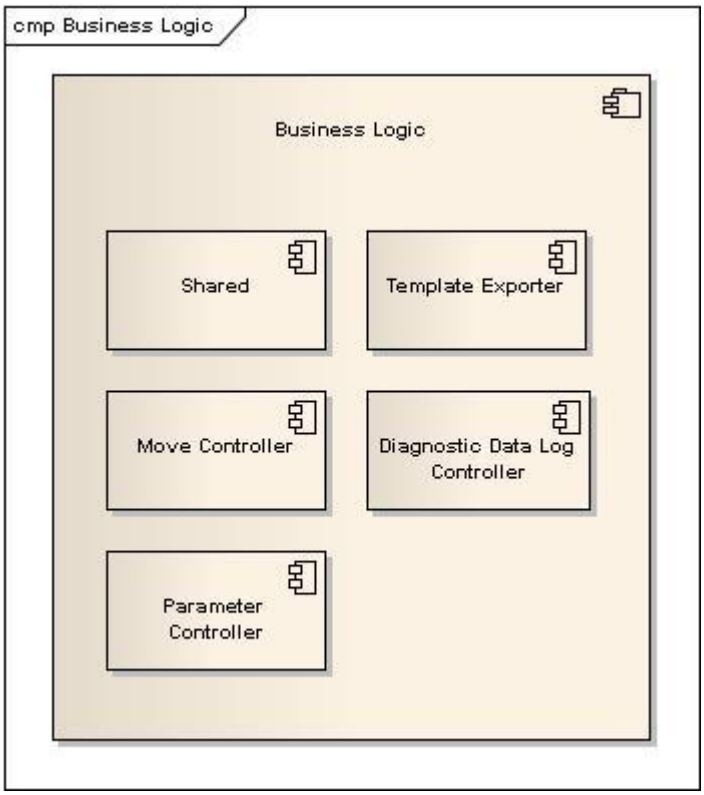
8.2.1.1 File Handler

The file handler makes all the communication with the persistence files. It is responsible for read and writes the data from and to the files located on the running system. The file handler knows how to serialize each individual file. The file handler should only be used by the persistence object provider.

8.2.1.2 Persistence Object Provider

The persistence object provider makes the data object available to other packages. So not every action with data objects have to be done with the files.

8.2.2 Business Logic



8.2.2.1 Shared

Contains shared functionalities other layers can use like interfaces.

8.2.2.2 Template Exporter

The template exporter is responsible for generating a new Tecan Base SDK template.
See also chapter 4.2 Export Parameters to a Template.

8.2.2.3 Move Controller

All functions which are supported by the Tecan Base SDK and are concerned with moving an axis are handled inside the move controller.

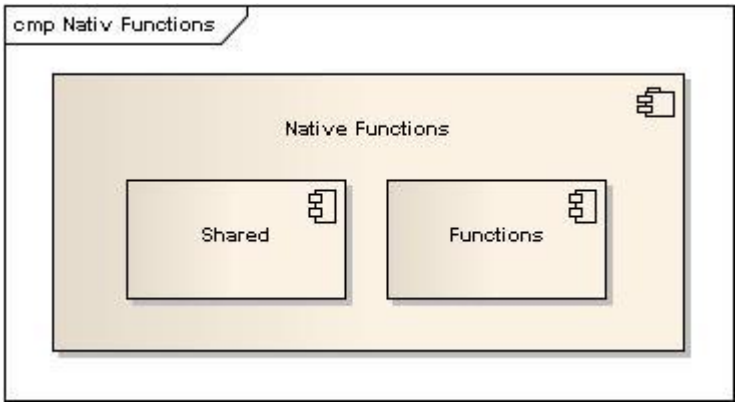
8.2.2.4 Diagnostic Data Log Controller

Configuration, starting and evaluating the embedded movement controller diagnostic data log is the task of this component.

8.2.2.5 Parameter Controller

The parameter controller supports the other components with the correct parameters and is designed to do calculations for parameters if needed.

8.2.3 Native Functions



8.2.3.1 Shared

Used for shared functionalities other layers can use like interfaces.

8.2.3.2 Functions

A library of functions the Tecan Base SDK does not support, who have to be implemented additionally to assure the fully extend of the tool.

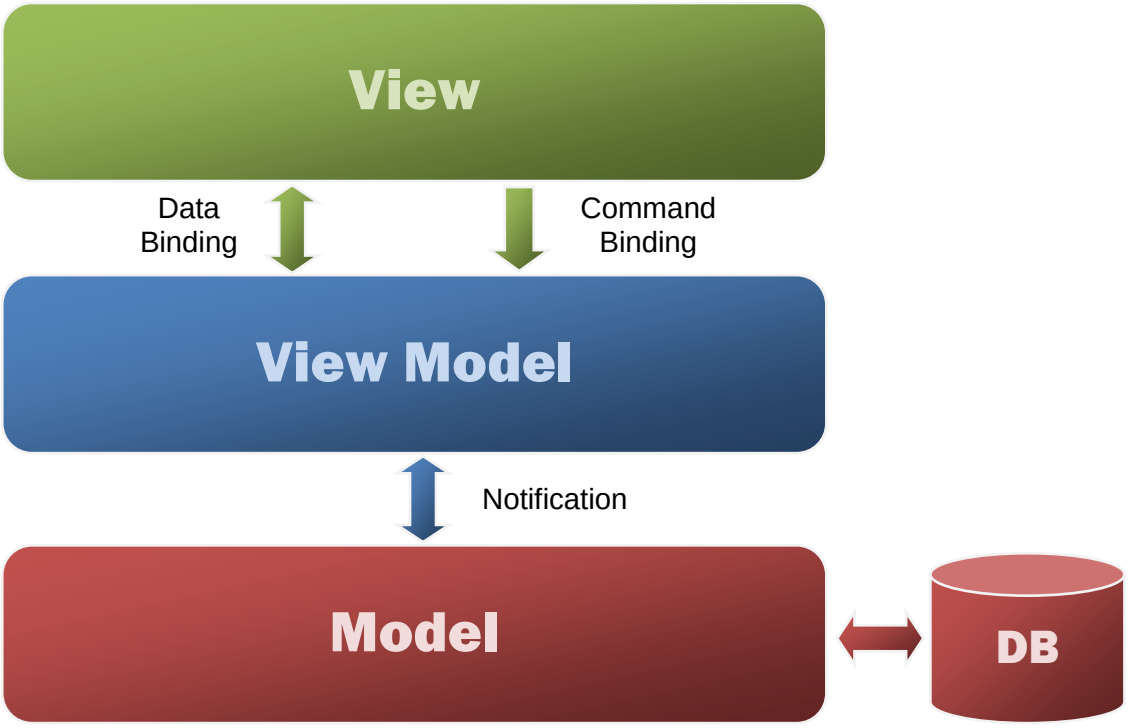
9 Interface View

The Tecan Base SDK remoting services will be used to.

For further information can be obtained from the Tecan Base SDK documentation. (See Ref. [4])

10 Design View

10.1 MVVM



The MVVM (Model – View – View Model) is a design pattern mostly used in WPF applications. The main purpose of the pattern is to isolate the business logic from the view.
Other patterns with similar functionalities are MVC (Model – View – Controller) or MVP (Model – View – Presenter).

10.1.1 Model

As in the MVC and MVP pattern the model has the task to organize the data access and business logic.

10.1.2 View

The view refers to all elements displayed in the GUI such as buttons, windows, graphics and other controls.

10.1.3 View Model

The new part of the MVVM pattern is the View Model. It is “a model of the view”, an abstraction of the view that also serves as in data binding between the model and the view. It converts the information provided by the model to a data structure used in the view and vice versa. It also interprets user actions and passes them on to the model.

10.1.4 Data Binding

Data binding is one or two way. The GUI control will always show the data the bounded model view property holds. If the user changes a value from a two way bounded property the setter from the model view property will be executed. This way changes can be forwarded from the Model View to the Model.

10.1.5 Command Binding

Commands are also direct bounded to a method in the View Model. They will be executed of the user uses the controls that are bounded.

10.1.6 Notification

The Notifications between the Model View and the Model are synchronal method calls or event based calls. View Models can subscribe them to a publisher class inside the Model and update their properties if anything changes.

10.2 Data Log Notification

As an example for the notification process the data log process will be described.

10.2.1 Classes

10.2.1.1 Plot View

Plot views are view classes who present data on a line chart. Three instances will be built for the three definable test points. Theoretically more instances are possible.

10.2.1.2 Plot Model View

It is a model view class. Holds the series of line chart data. One instance per plot view exists.

10.2.1.3 Data Log Handler

It is places inside model. It provides an event which is fired if new data is available. The data log handler also has a method to add new data.

10.2.1.4 Tecan Base SDK Handlers

Various classes handling the communication with the Tecan Base SDK or the motion control unit. They can generate the chart data.

10.2.2 Process

1. During the construction of the plot view model they register an event handler to the data log handler that shall be executed if new data is added.
2. The user commands to execute a move. After the axis has finished the move a part of the Tecan Base SDK handler read out the diagnostic data log from the motion control and decode the input.
3. The decoded data will be sent to the data log handler together on which axis on what time the data was generated.
4. The data log handler fires an event with the new data.
5. Each plot model view checks if the new data has to be considered.
6. If yes the plot data property will be set with the new data.
7. The plot view will update itself thanks to the one way binding.

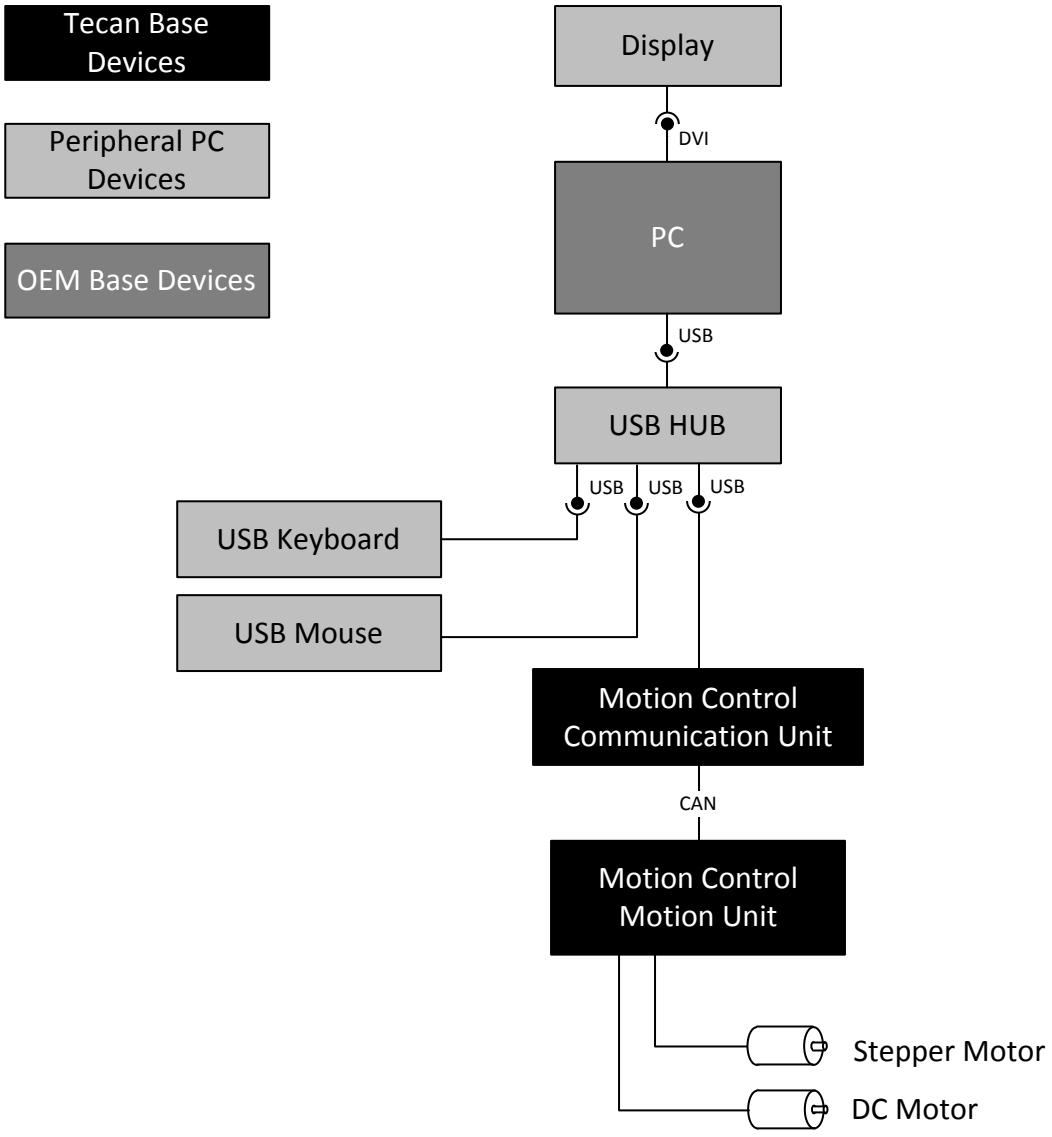
10.3 Threading Design

General the number of threads should be as few as possible. Most of the processes are handled by the startup GUI thread. Processes who have to wait, for example waiting till a move has finished or all diagnostic data could be received from the firmware, have to be dispatch so the GUI wont freeze. The dispatching will be done as close as possible to the Tecan Base SDK.

10.4 Update on Use

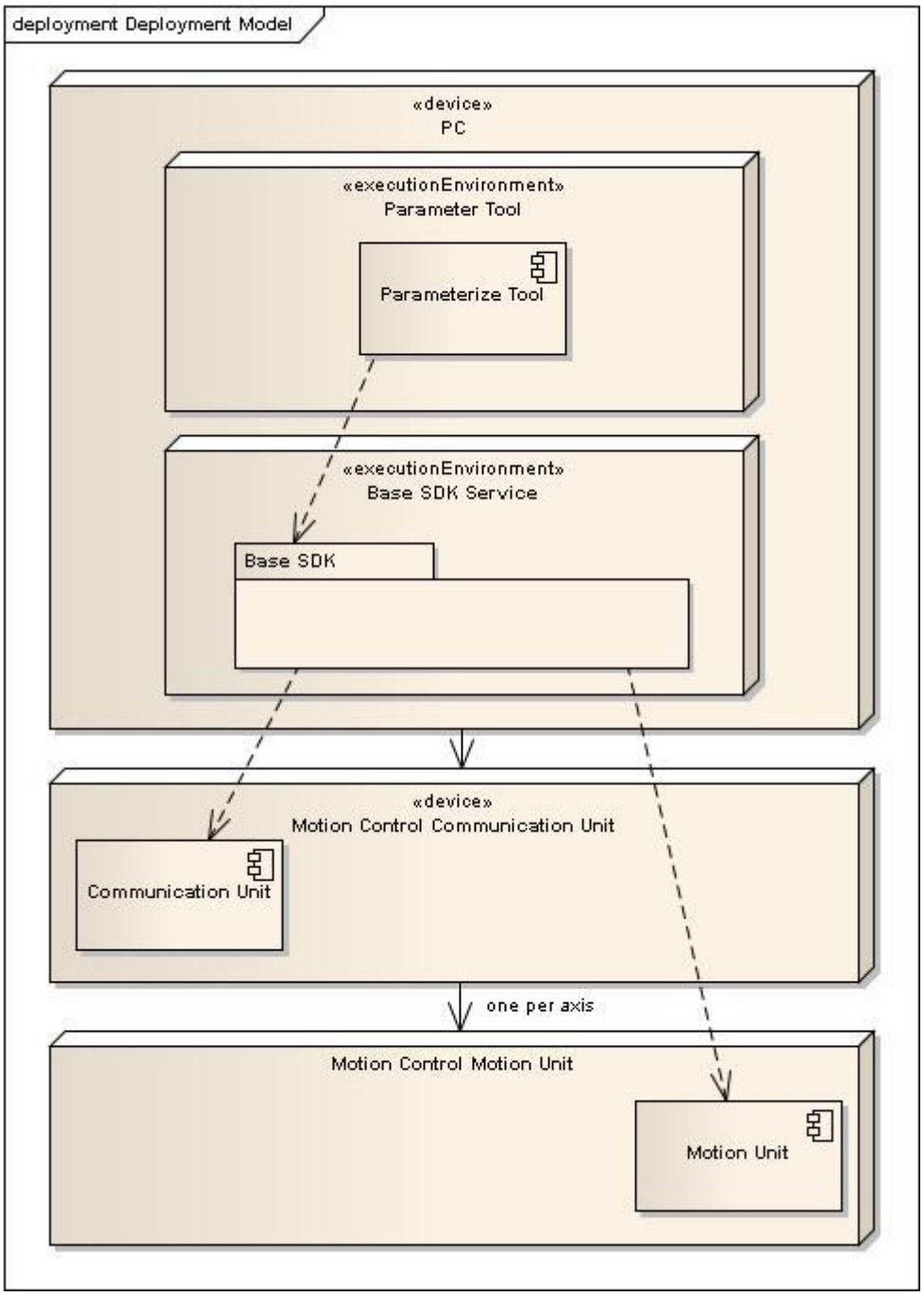
All properties the user can change will have no effect until a new move has to be executed. Because of this fact all parameters will be sent every time to the instrument right before a move.

11 Infrastructure View



The Parameterize Tool is running on the PC. Normally a display is connected to that PC using a DVI or VGA port driven by the OS. To connect a Tecan instrument to the PC, it has to provide an USB port. Normally the PC hosts a USB hub and provides more than one USB port. To them may some other peripheries like a keyboard or a mouse is connected. The module controlled by an embedded motion control communication unit transfers commands from the USB port to the CAN bus and back. To the CAN bus devices can be connected. These devices are controlled by the embedded motion control motion unit. For those modules the Tecan Base SDK provides device drivers. Normally to the module controlled by the embedded motion control motion unit DC or stepper motors are connected.

12 Deployment View



The Parameterize Tool runs in its own process on the PC. The Tecan Base SDK also runs in tis own process. The embedded components (Communication Unit and Motion Unit) are executed on an embedded processor on dedicated PCBs. To install (download firmware binaries) and configure the firmware the Tecan Base SDK provides drivers.
Processes which take a longer time should not block the GUI and are dispatched between the Param Tool layer and the Tecan Base SDK layer.

13 Operational View

13.1 Logging

The logging system of Tecan Base SDK will be used.

13.2 Location of Base Binaries

The binaries of the Parameterize Tool are located in:

```
Program Files\Tecan\ParameterizeTool
```

14 Security View

No specific security aspects exist for the Parameterize Tool.

15 Data View

15.1 Parameter Set Data

Files containing the different parameter set of motors, encoder and axes.
XML is used, stored on the file system of the computer.
The files shall be editable by the user with a normal text editor. So no strong signature or a CRC is used to validate the files.

15.1.1 Versioning

The parameter set data should not change the schema. If it has to change because of firmware or Tecan Base SDK changes an updater has to be written that export all data to the new schema.

15.1.2 XML File Structure

For each type of storing module a separate file exists. These files are:

- axes.xml
- motors.xml
- encoders.xml
- grouping.xml

15.1.3 XML Structure

15.1.3.1 Instances

In the axes.xml, motors.xml and the encoders.xml file directly under the root node a list of the respective instances are saved. Their individual main node contains an id attribute working as a unique identifier.

```
<Motors>
      …
      <Motor id =”motorOne”>
            …
      </Motor>
      …
</Motors>
```

```
<Encoders>
      …
      <Encoder id =”encoderOne”>
            …
      </Encoder>
      <Encoder id =”encoderTwo”>
            …
      </Encoder>
      …
</Encoders>
```

```
<Axes>
      …
      <Axis id =”axisOne”>
            …
            <RefEncoder1 id=”EncoderOne” />
            <RefEncoder2 id=”EncoderTwo” />
```

```
            <RefMotor id=”MotorOne” />
            …
      </Axis>
      …
</Axes>
```

15.1.3.2 Grouping

The grouping.xml builds up one or more axis to axis groups, called devices. One or more devices have to be in a device group node. An attribute works as a unique identifier. The axes are referenced with their unique id.

```
<Grouping>
      <DeviceGroup id =”groupOne”>
            <Device id =”deviceOne”>
                  <RefAxis id=”axisOne” />
            </Device>
      </DeviceGroup>
</Grouping>
```

15.2 Data Model

15.2.1 Mirror Classes

Concerning the data model separate classes are implemented to mirror the Tecan Base SDK configuration containers. Where ever it is reasonable the Tecan Base SDK classes are used, like enums for example. Mirror classes for the axis, the motor and the encoder exist. The axis mirror class contains the reference to one to three encoder mirror class and one reference to the motor mirror class.

15.2.2 Extensibility

One advantage towards a direct Tecan Base SDK classes using approach is the extensibility. Every parameter in the new mirror class can be implemented with a generic class. This generic class supports the possibility to buffer new values the user changes and save or drop the new values. It also can enhance the parameter with additional information like the unit of the value.

16 Technology Section

For the Parameterize Tool C# and the.Net framework V4.0 was chosen because:

- The Tecan Base SDK shall be used.
- The target operating systems are Windows XP and Window 7 and C# and the .Net framework are supported for them both in a 32 and a 64 bit version.
- The delivered binaries are CPU unspecific, the operating system cause if the binaries are executed in a 32 or 64 bit environment.
- C# is the latest development of the C language family.
- C# has features like garbage collection avoiding memory leaks and thus enhancing software stability.
- C# allows reusing legacy C and C++ code.
- C# allows interfacing to legacy C and C++ components.
- .Net framework V4.0 is a well establish library.
- .Net framework V4.0 contains feature rich technologies like WPF.

WPF is used for all GUI components, preferably by creating controls, allowing a state-of-the-art user interface.

17 Architectural Justification

17.1 General

The Tecan Base SDK has to be used and so many architectural decisions are already made, like using the Tecan Base SDK remoting services. One further goal is to replace the current LUA implementation entirely.

17.2 Presentation Layer

The MVVM patter is common used and for simply little tools like this one a simple but powerful support.

17.3 Simple Architecture

The tool shall be used by a team not trained in .net C#. They may have experience with the language, but this is no pre condition. Nevertheless they should be able to debug the tool if as example an error occurs. Also the further development is not secured to be done by one of the software team members.

Graphical User Interface Design

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **-**

	Author	Reviewer	Approver
Name	Andreas Zollinger	Luc Bläser	Joas Leemann
Function	Software Engineer	Supervisor HSR	Project Leader
Date / Visa			

Table of Contents

1 Introduction

1.1 Purpose

The GUI design describes the look and behavior of the windows and screen. It further describes the connection between the user controls and the functions required in the SWS.

1.2 Scope

This document belongs to the Parameterize Tool. This document is generated during the “Design Input” phase and is first released at M3 in version 1.0. Changes after the first release will be documented in the document history.

1.3 Definitions, Acronyms and Abbreviations

Definitions, acronyms and abbreviation can be found in the global table (see Ref. [1])

1.4 References

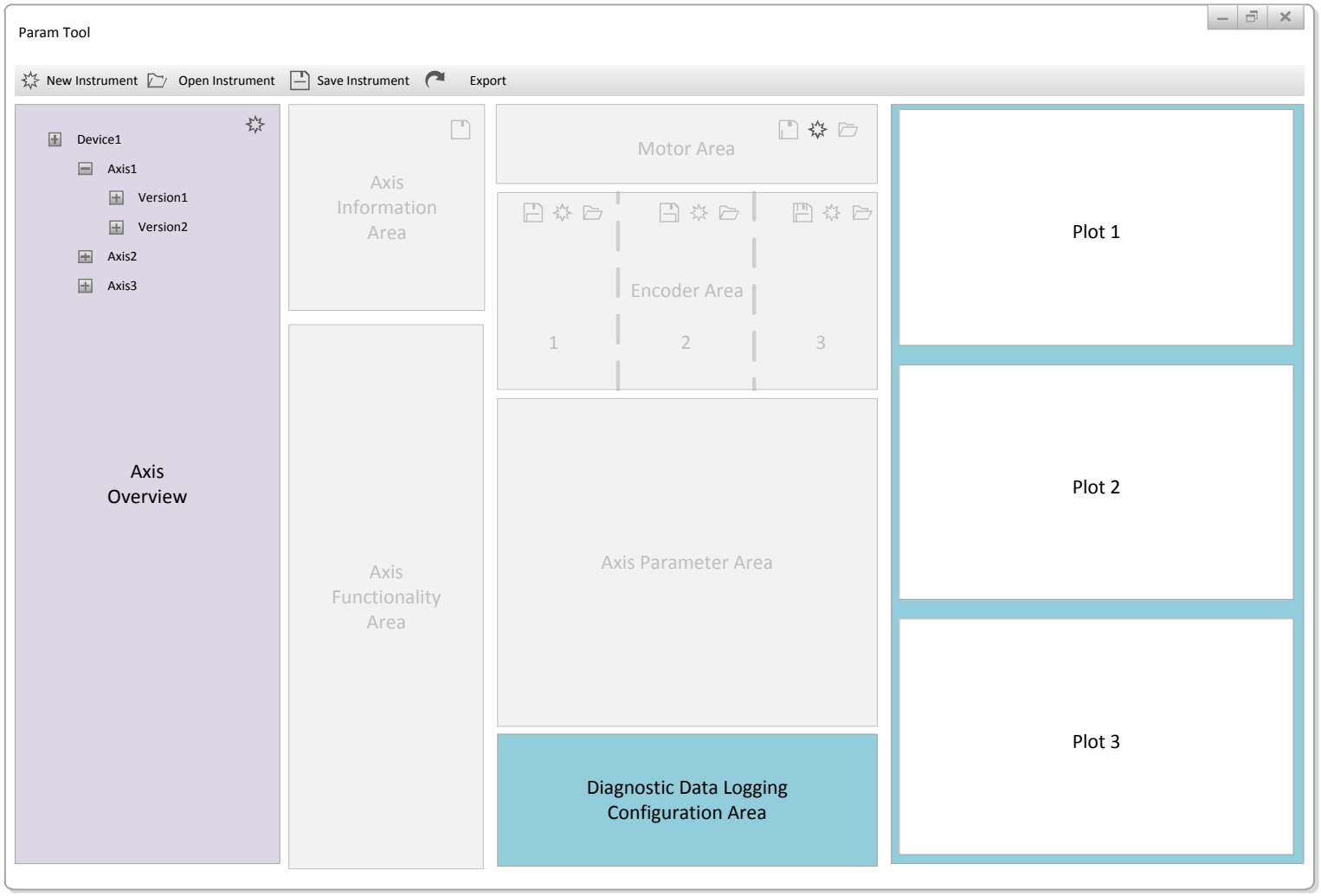
Ref #	*Description*
Ref. [1]	Definition, Acronyms and Abbreviations for Parameterize Tool, 90_DefinitionAcronymsAbbreviations.pdf, V1.0
Ref. [2]	Traceability Matrix for Parameterize Tool, 91_TraceabilityMatrix.pdf, V1.0

1.5 Document Change History

Date	*Version*	*Change*	*Author*
2011-12-18	1.0	Initial version	AnZo

2 Prototype Layout

2.1 Empty Window

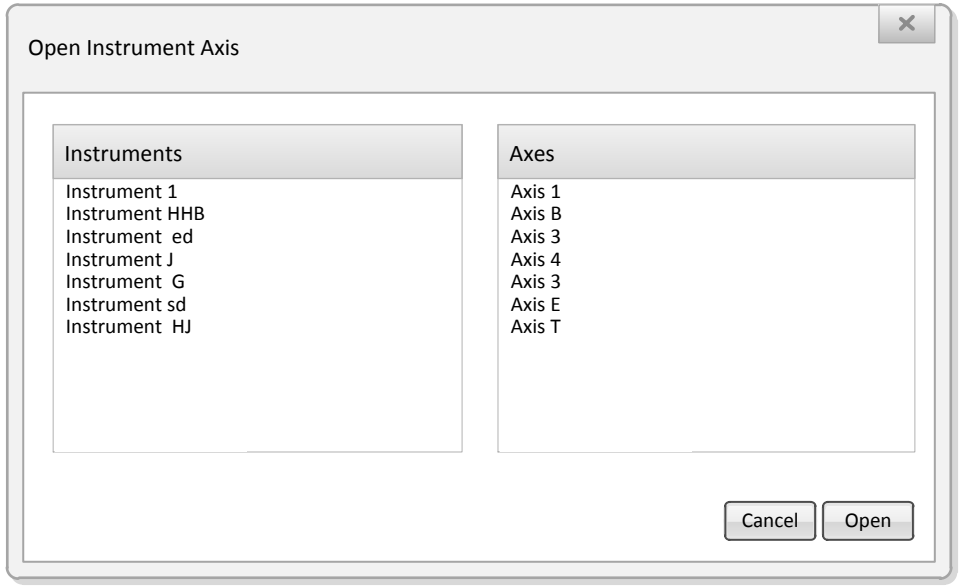


In the empty screen all axis, motor and encoder fields are visible but not enabled.

2.1.1 Open Axis / Instrument

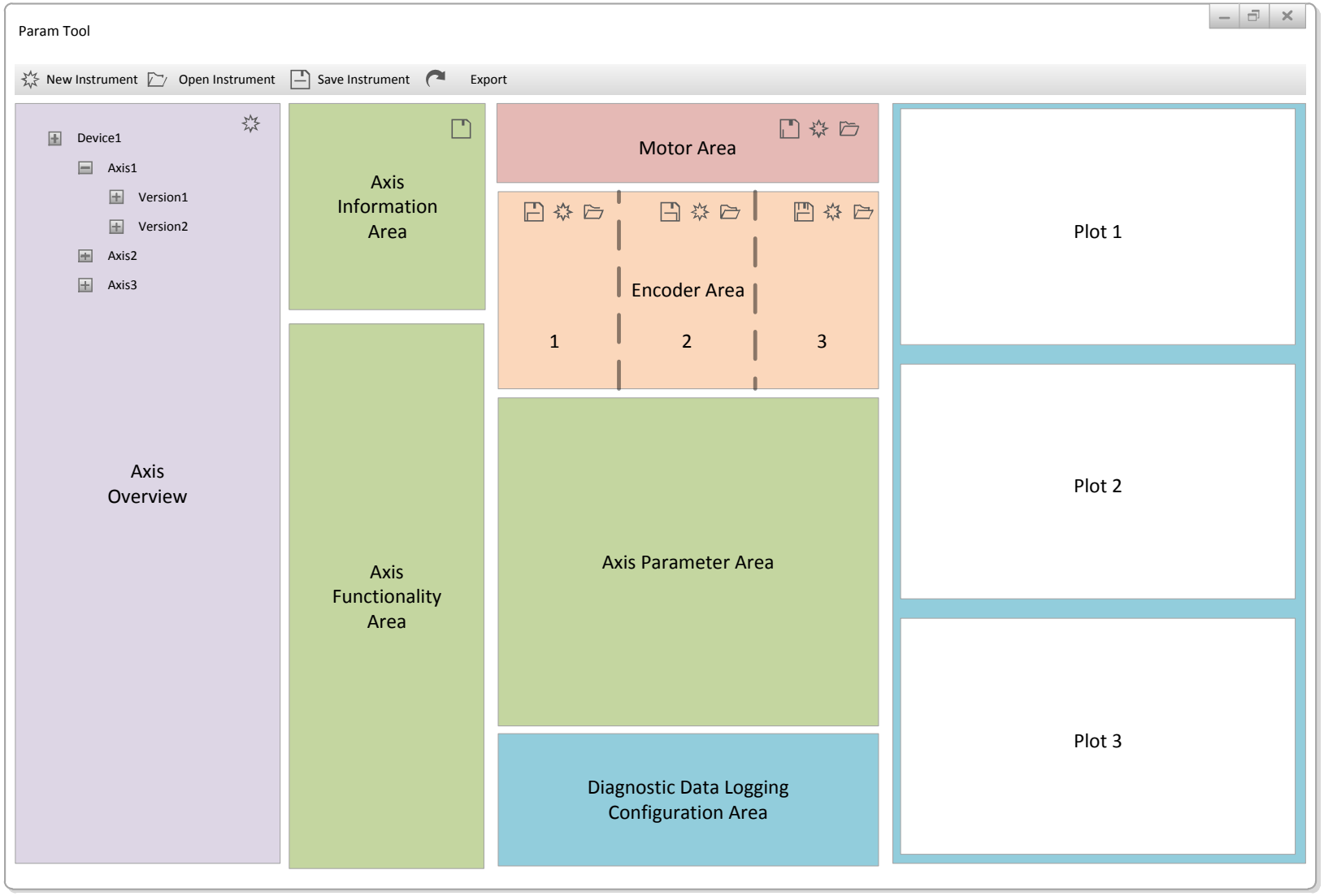
The user has the possibility to open an axis or an instrument.

2.2 Axis / Instrument Loading Screen



All available instruments and axes will be presented. The user can select one instrument or several axes and confirm the selection with an open click.

2.3 Loaded Window



2.3.1 Open Instrument

The user has the possibility to open an instrument.

2.3.2 Change View to Devices and Axes

The user can freely change the view from different axes.

2.3.3 Add Device / Axis

With a click on the *-button in the tree view area a new axis or device can be newly generated.

2.3.4 Save Axis / Instrument

The user can save the current shown axis, all axes or the whole instrument.

2.3.5 Export Axis / Instrument

The user can start the export functionality from inside the tool.

2.3.6 Axis Information Area

In this area the different information’s about the axis are prompted.

2.3.7 Axis Parameters Area

In this area all parameters for the axis are displayed.

2.3.8 Motor Parameters Area

In this area all motor parameters are displayed.
The user has the possibility to change the motor parameter set or create a new one.

2.3.9 Encoder Parameters Areas

In this area one or two parameters are displayed.
The user has the possibility to change the motor parameter set or create a new one.

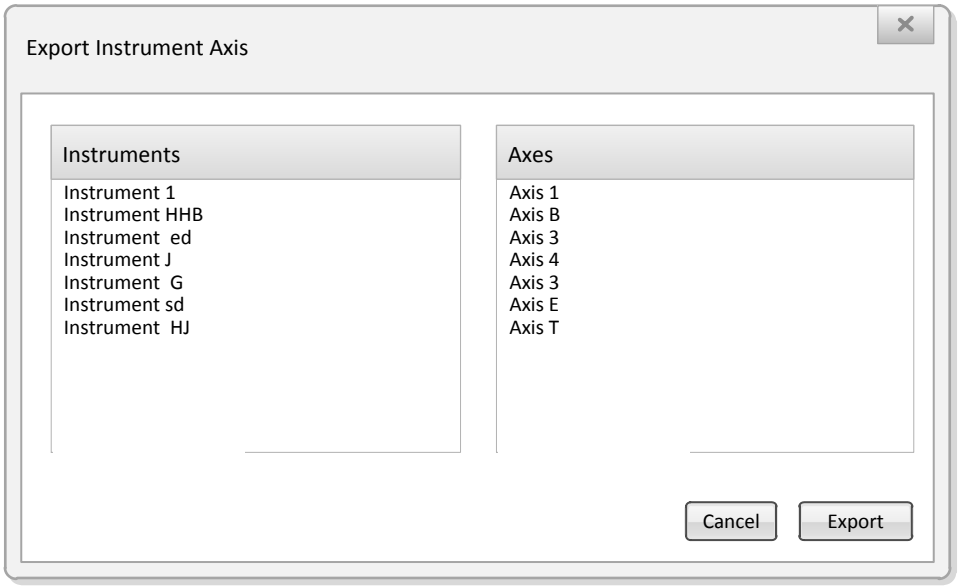
2.3.10 Axis Functionality Area

In this area all the different configurations and executing commands are placed.

2.3.11 Diagnostic Data Area

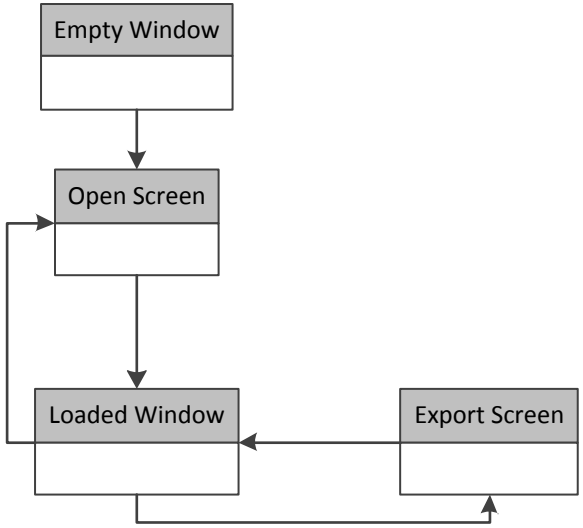
In this area all the different configurations and displays for logging purposes are placed.

2.4 Export Screen



The user can select one instrument or one axis to export and confirms the selection with an OK click.

3 Prototype Screen Flow



The startup screen is the empty window. From there the user can open an axis or an instrument through the open screen. The loaded window shows all necessary information about an axis and the parameters. All functions can be executed inside this window. For exporting data an export screen will be opened.

4 Applicable Standards

4.1 Implementation Technology

For implementation of the GUI .net WPF is used. With this framework the common used pattern MVVM will be used.

4.2 GUI Behavior

- The GUI provides the user immediate feedback, there is no “freeze screen”
- The GUI presents values as editable to the user only if they really are editable.
- The GUI only presents area of values to the user if they really exist. (Example: an axis has no hard stop, so the “find hard stop” functionality will not be shown)
- The GUI uses simply validation and gives to the user (Example: the GUI validate if the input is a number on a number field but does not validate if the number is applicable.)
- Command inputs (like button or menu) are grayed out if not applicable or even hidden.
- Icons and labels should be chosen like other typical Microsoft products and have the expected behavior.

5 Traceability

The traceability is handled inside the global traceability matrix file. (See Ref. [2])

Software Configuration Management Plan

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **-**

	Author	Reviewer	Approver
Name	Andreas Zollinger	Luc Bläser	Joas Leemann
Function	Software Engineer	Supervisor HSR	Project Leader
Date / Visa			

Table of Contents Page

1 Introduction

1.1 Purpose

The purpose of this document is to describe the configuration management during the different phases of the SW Product Life Cycle of the Parameterize Tool. It defines what has to be managed.

1.2 Scope

The scope of this document is to help and guide software developers and testers through the configuration management focusing on the software tools.

1.3 Definitions, Acronyms and Abbreviations

Definitions, acronyms and abbreviation can be found in the global table (see Ref. [2])

1.4 References

Ref #	*Description*
Ref. [1]	Project Development Plan, 01_ProjectDevelopmentPlan.pdf, V1.1
Ref. [1]	Definition, Acronyms and Abbreviations for Parameterize Tool, 90_DefinitionAcronymsAbbreviations.pdf, V1.0

1.5 Document Change History

Date	*Version*	*Change*	*Author*
2011-20-31	1.0	Initial Version	AnZo

2 Description

2.1 Organization, Responsibilities and Interfaces

The responsibilities and roles within the project are described in the Project Development Plan of this Project (see Ref. [1]).

2.2 Tools, Environment and Infrastructure

2.2.1 Tools used during Development

Name	*Manufacturer*	*Unique Identifier (eg. Version or Release Date)*
Team Foundation Server (TFS)	Microsoft	Version 2010
Visual Studio	Microsoft	Version 2010
Enterprise Architect	Sparx Systems	Version 7.5

2.2.2 Development Environment

Windows XP 32 bit, Windows 7 32 bit and Windows 7 64 bit
.Net Framework 4.0

2.2.3 Target Environment

Windows XP 32 bit, Windows 7 32 bit and Windows 7 64 bit
.Net Framework 4.0

2.3 Control of Tools, Environment and Infrastructure

During development the tool is compiled not on a specific computer.
After this project when the tool will be released, compiling will be on a PC which has the relevant tools installed, the access is limited and of which an image is created, if relevant changes to the environment are performed.

2.4 Configuration Identification

2.4.1 Software Versions

Name of Configuration Item	*Version*	*Used in SW Version*
Tecan Base SDK	tbd	-

2.4.2 SOUP Components

Name	*Manufacturer*	*Unique Identifier (eg. Version or Release Date)*
-	-	-

No SOUP components used.

2.4.3 Build Mechanism

Complete source code and all additional files have to be in proper place.
The compiling will be done inside Visual Studio 2010.
The executable program is placed in the standard Tecan Base SDK place.

2.4.4 Versioning

No versioning will be done during this project.
At the end of the project the current software state will be labeled as draft version 0.1.

TECAN

Part III - Appendix

Review Protocol PRD

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **-**

1 Table of Contents

2 Review Summary

2.1 Review Assessment

☐ Accepted, no changes required

☐ Accepted, small changes required

☒ Not accepted, important changes required

2.2 Summary of findings

Major (important)	1
minor (small)	6
? (Question)	1
Total	8

2.3 Objects under Review

PRD for Parameterize Tool

2.4 Reference Material used for the Review

n/a

2.5 Review Date

Review Date: 2011-10-24

Review Time: 10:00 – 11:00

2.6 Review Team

Name	*Initials*	*Review Function*	*Date*	*Signature*
Andreas Zollinger	AnZo	Author	2011-10-24	
Joas Leemann	JoLe	Moderator	2011-10-24	
Remo Kälin	ReKa	Reviewer	2011-10-24	
Luc Bläser	LuBl	Independent Reviewer	2011-10-24	

2.7 Review Completion

Name	*Initials*	*Review Function*	*Date*	*Signature*
Joas Leemann	JoLe	Recorder		

Id	*Initials*	*Area*	*Description of Finding*	*M*	*m*	*?*	*Who*	*When*	*Implementation*
01	ReKa	General	In the PDP or the SSD a section about the migration from the current tool has to be added.	☐	☒	☐	AnZo		
02	ReKa	Chapter 3.3.1.1	The description of the Motion Control Group has to be described more precise.	☐	☒	☐	AnZo		
03	ReKa	Chapter 3.3.1.1	Add a reference to the SSD to explain the overall architecture.	☐	☒	☐	AnZo		
04	ReKa	Chapter 3.3.2	User criteria “physical knowledge” must contain more detailed information.	☐	☒	☐	AnZo		
05	LuBi	PRD 4	Is “usability” part of this PRD?	☐	☐	☒	JoLe		No, it is part of the SSD.
06	ReKa	PRD 12	The “special moves” are not needed for parameterization. Remove this requirement.	☐	☒	☐	AnZo		
07	LuBi	PRD 13	The SW should also allow extending the parameters.	☐	☒	☐	AnZo		
08	ReKa	New PRD	The requirement for “Current Control Tuning” is missing.	☒	☐	☐	AnZo		

Review Protocol

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **SWS v1.0**

1 Table of Contents

2 Review Summary

2.1 Review Assessment

☐ Accepted, no changes required

☒ Accepted, small changes required

☐ Not accepted, important changes required

2.2 Summary of findings

Major (important)	0
minor (small)	3
? (Question)	0
Total	3

2.3 Objects under Review

SWS v1.0 for Parameterize Tool

2.4 Reference Material used for the Review

n/a

2.5 Review Date

Review Date: 2011-12-12

Review Time: 13:00 – 14:30

2.6 Review Team

Name	Initials	Review Function	Date	Signature
Andreas Zollinger	AnZo	Author	2011-12-12	A. Zollinger
Joas Leemann	JoLe	Moderator	2011-12-12	
Claudio Leibacher	ClLe	Reviewer	2011-12-12	C. Leibacher
~~Remo Kälin~~	~~ReKa~~	~~Reviewer~~		
Lub Bläser	LuBl	Independent Reviewer	2011-12-12	

2.7 Review Completion

Name	Initials	Review Function	Date	Signature
Joas Leemann	JoLe	Recorder	2011-12-21	

Id	*Initials*	*Area*	*Description of Finding*	*M*	*m*	*?*	*Who*	*When*	*Implementation*
1	JoLe	Chapter 1.5	Remove empty line in table	☐	☒	☐	AnZo	2011-12-19	Done
2	ClLe	SWS 5	It shall be possible to handle 3 encoders	☐	☒	☐	AnZo	2011-12-19	Done
3	ClLe	Chapter 3.2	It shall be possible to move an axis while another axis is monitored (axis where the log data comes from).	☐	☒	☐	AnZo	2011-12-19	Done

Review Protocol

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **UCS v1.0**

1 Table of Contents

2 Review Summary

2.1 Review Assessment

☐ Accepted, no changes required

☒ Accepted, small changes required

☐ Not accepted, important changes required

2.2 Summary of findings

Major (important)	0
minor (small)	4
? (Question)	2
Total	6

2.3 Objects under Review

UCS v1.0 for Parameterize Tool

2.4 Reference Material used for the Review

n/a

2.5 Review Date

Review Date: 2011-12-12

Review Time: 13:00 – 14:30

2.6 Review Team

Name	Initials	Review Function	Date	Signature
Andreas Zollinger	AnZo	Author	2011-12-12	
Joas Leemann	JoLe	Moderator	2011-12-12	
Claudio Leibacher	ClLe	Reviewer	2011-12-12	
~~Remo Kälin~~	~~ReKa~~	~~Reviewer~~		
Lub Bläser	LuBl	Independent Reviewer	2011-12-12	

2.7 Review Completion

Name	Initials	Review Function	Date	Signature
Joas Leemann	JoLe	Recorder	2011-12-21	

TECAN.

Review Protocol for Parameterize Tool UCS

Id	*Initials*	*Area*	*Description of Finding*	*M*	*m*	*?*	*Who*	*When*	*Implementation*
1	JoLe	Chapter 1.5	Remove empty line in table	☐	☒	☐	AnZo	2011-12-19	Done
2	ClLe	Chapter 2	Is it possible to build encoder and motor w/o an axis?	☐	☐	☒	AnZo		Yes, an ad-hoc axis will be used.
3	ClLe	Chapter 3.9.1	Wrong text: “Failed End Condition”	☐	☒	☐	AnZo	2011-12-19	Done
4	ClLe	Page 15	Remove the single letter	☐	☒	☐	AnZo	2011-12-19	Done
5	ClLe	Chapter 3.16.1	Is it possible to send a single-command to an axis which is not selected?	☐	☐	☒	AnZo		Yes, the device-id can be changed ad-hoc.
6	AnZo	Chapter 3.16.4	Sending single-commands to a not selected axis shall be describe as an alternative scenario	☐	☒	☐	AnZo	2011-12-19	Done

Review Protocol

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **SSD v1.0**

1 Table of Contents

2 Review Summary

2.1 Review Assessment

☐ Accepted, no changes required

☒ Accepted, small changes required

☐ Not accepted, important changes required

2.2 Summary of findings

Major (important)	0
minor (small)	9
? (Question)	1
Total	10

2.3 Objects under Review

SSD v1.0 for Parameterize Tool

2.4 Reference Material used for the Review

n/a

2.5 Review Date

Review Date: 2011-12-12

Review Time: 13:00 – 14:30

2.6 Review Team

Name	Initials	Review Function	Date	Signature
Andreas Zollinger	AnZo	Author	2011-12-12	A. Zollinger
Joas Leemann	JoLe	Moderator	2011-12-12	
Claudio Leibacher	ClLe	Reviewer	2011-12-12	C. Leibacher
~~Remo Kälin~~	~~ReKa~~	~~Reviewer~~		
Lub Bläser	LuBl	Independent Reviewer	2011-12-12	

2.7 Review Completion

Name	Initials	Review Function	Date	Signature
Joas Leemann	JoLe	Recorder	2011-12-21	

Id	Initials	Area	Description of Finding	M	m	?	Who	When	Implementation
1	LuBu	Chapter 7.2.2.2	State that the data-classes are used from the Tecan Base SDK.	☐	☒	☐	AnZo	2011-12-19	Done
2	LuBl	Chapter 7.2	State that all components of the business layer accesses the Base SDK layer.	☐	☒	☐	AnZo	2011-12-19	Done
3	LuBl	Chapter 8.2.1.1	State that the file handler is only used from the persistence object provider.	☐	☒	☐	AnZo	2011-12-19	Done
4	LuBl	Chapter 15	Describe the internal data structure. State what the data model is.	☐	☒	☐	AnZo	2011-12-19	Done
5	JoLe	Chapter 9	Add a reference to the Tecan Base SDK documentation	☐	☒	☐	AnZo	2011-12-19	Done
6	LuBl	Chapter 10	Describe the threading design	☐	☒	☐	AnZo	2011-12-19	Done
7	LuBl	Chapter 10	Describe the data interaction with the instrument (are they explicit or synchronous?)	☐	☒	☐	AnZo	2011-12-19	Done
8	LuBl	Chapter 10	Describe the event-binding with the log-data. Add details to the notification.	☐	☒	☐	AnZo	2011-12-19	Done
9	LuBl	Chapter 12	The GUI shall no be blocked while the instrument is moving. Maybe a receive thread with event dispatching is necessary.	☐	☒	☐	AnZo	2011-12-19	Done
10	LuBl	Chapter 15.1	Shall versioning be supported?	☐	☐	☒	AnZo	2011-12-19	Done

Review Protocol

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **GUID v1.0**

1 Table of Contents

2 Review Summary

2.1 Review Assessment

☐ Accepted, no changes required

☐ Accepted, small changes required

☒ Not accepted, important changes required

2.2 Summary of findings

Major (important)	2
minor (small)	1
? (Question)	0
Total	3

2.3 Objects under Review

GUID v1.0 for Parameterize Tool

2.4 Reference Material used for the Review

n/a

2.5 Review Date

Review Date: 2011-12-12

Review Time: 13:00 – 14:30

2.6 Review Team

Name	Initials	Review Function	Date	Signature
Andreas Zollinger	AnZo	Author	2011-12-12	A Zollinger
Joas Leemann	JoLe	Moderator	2011-12-12	
Claudio Leibacher	ClLe	Reviewer	2011-12-12	C. Leibacher
~~Remo Kälin~~	~~ReKa~~	~~Reviewer~~		
Lub Bläser	LuBl	Independent Reviewer	2011-12-12	

2.7 Review Completion

Name	Initials	Review Function	Date	Signature
Joas Leemann	JoLe	Recorder	2011-12-21	

Id	*Initials*	*Area*	*Description of Finding*	*M*	*m*	*?*	*Who*	*When*	*Implementation*
1	JoLe	Chapter 1.5	Remove empty Lines in table, correct date	☐	☒	☐	AnZo	2011-12-19	Done
2	ClLe	Chapter 2.1	Plots have to be enlarged	☒	☐	☐	AnZo	New version	
3	JoLe	Chapter 3	GUI design with tree-control instead of a tab-control is missing	☒	☐	☐	AnZo	New version	

Definition, Acronyms and Abbreviations

Project-Name: **Parameterize Tool**

Project Number: **-**

Subject: **-**

Table of Contents

1 Introduction

1.1 References

Ref #	Description
Ref.[1]	http://en.wikipedia.org/w/index.php?title=Non-disclosure_agreement&oldid=455105337

1.2 Document Change History

Date	*Version*	*Change*	*Author*
2011-11-01	1.0	Initial version	AnZo

2 Definitions, Acronyms and Abbreviations

BU	Business Unit
BP	BioPharma, BU of Tecan
CAN	Controller Area Network. Specification for a network.
CD	Clinical Diagnostics, BU of Tecan
CRC	Cyclic Redundancy Check; It is an error-detecting code designed to detect accidental changes to raw computer data, and is commonly used in digital networks and storage devices
CDA	Confidentiality Disclosure Agreements, legal contract between at least two parties that outlines confidential material, knowledge, or information that the parties wish to share with one another for certain purposes, but wish to restrict access to by third parties. See Ref.[1]
CFR	Code of Federal Regulations. Codification of the general and permanent rules and regulations of the United States of America.
COM	Common Object Model. Interprocess communication mechanism.
DHF	Design History File, compilation of documentation that describes the design history.
DLL	Dynamic Link Library
DVI	Digital Video Interface. Electric interface to transfer video data.
ECTS	European Credit Transfer System
FDA	Food and Drug Administration, agency of the HHS
FuMu	Funktionsmuster, Breadboard
FW	Firmware
GUI	Graphical User Interface
HHS	United States Department of Health and Human Services, cabinet department with the goal to protecting the health of all Americans.
HSR	Hochschule Rapperswil
I2C	I^2C: Inter-Integrated Circuit, multi-master serial single-ended computer bus
ICP	Instrument Communication Protocol. Communication protocol for data transfer to Tecan instruments.
IQ	Installation Qualification
IVD	In-vitro Diagnostic
IVDD	In-vitro diagnostics directive
LH	Liquid Handling
LUA	Portuguese for moon, scripting language
M	Milestone
NA	Not Applicable
NPV	Net Present Value
OEM	Original Equipment Manufacturer
OS	Operating System. SW to manage HW resources and executing application SW.
PC:	Personal Computer.
PCB	Printed Circuit Board. Mechanically supports and electrically connects electronic components.
PID	PID Controller: proportional–integral–derivative controller
PL	Project Leader
PRD	Product Requirements Document
PT	Project Team
QM	Quality Management
RA	Regulatory Affairs
R&D	Research and Development
RfV	Ready for Validation
RS232:	Recommended Standard 232. Specifies a serial communication protocol.

SCR	Software Change Request
SDK:	Software Development Kit. A set of development tools that allows for the creation of applications for a certain HW platform.
SOP	Standard Operating Procedure
SOUP	Software Of Unknown Provenance
S&S	Setup and Service Software
ST	System Test Team
SW	Software
TBD	To be defined later
UCS	Use Case Specification
USB:	Universal Serial Bus. Specifies a serial communication protocol.
VAR	Value Added Reseller
VGA:	Video Graphics Array. Electric interface to transfer video data.
WPF:	Windows Presentation Foundation. .NET library for GUI coding.
XML:	Extensible Markup Language. Text-based markup language for structuring hierarchical data.

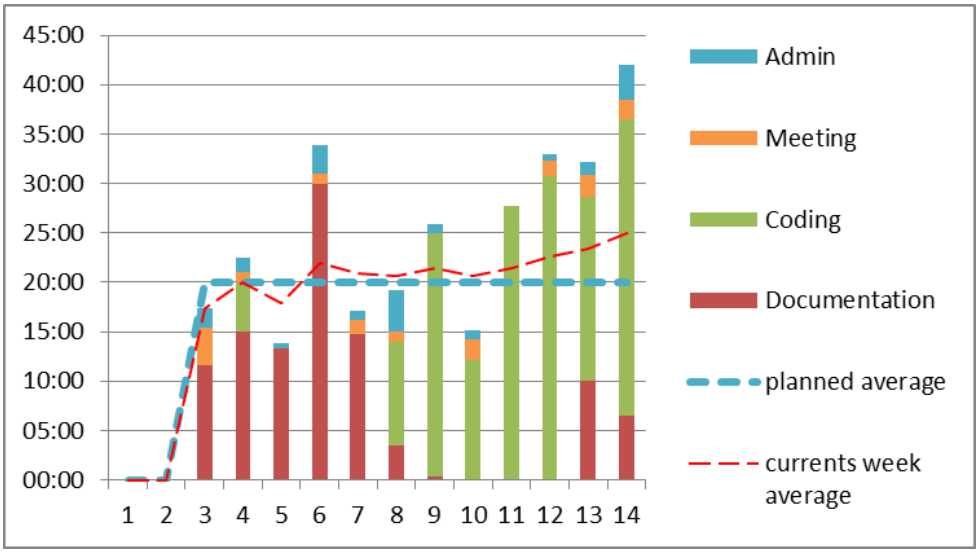
Project Time Overview

1 Used Time

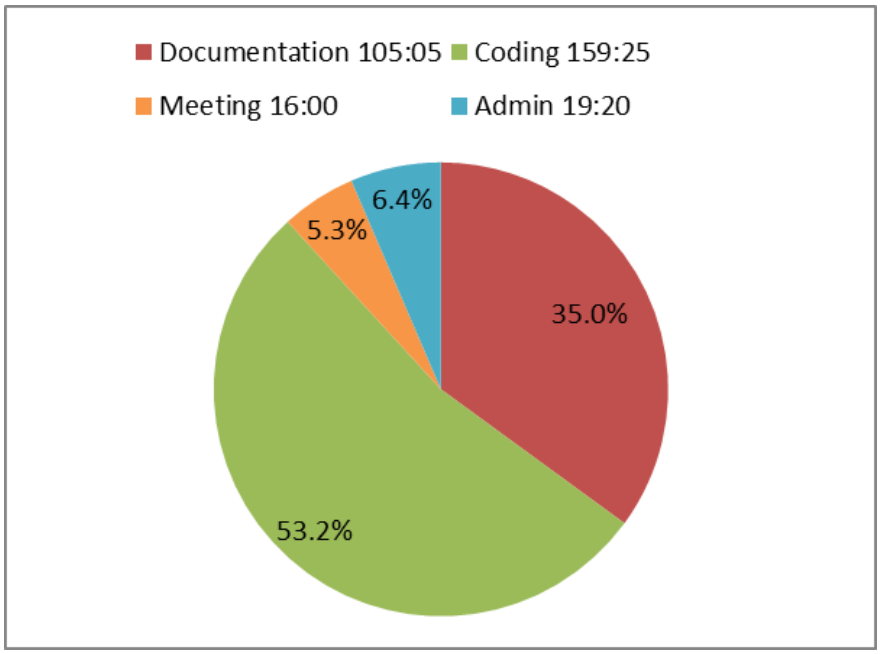
This The Term Project HSR module has a value of 8 ECTS points. One ECTS equals a time effort around 30 hours. Thus, this module should have the outline of about 240 hours.

Estimated Time: 240 hours
Time Worked: 300 hours
Additional Work: 60 hours

1.1 Time per Week



1.2 Time per Topic



1.3 Planned Project Time Line

	1							2							3							4							5							6							7						
	19.09	20.09	21.09	22.09	23.09	24.09	25.09	26.09	27.09	28.09	29.09	30.09	01.10	02.10	03.10	04.10	05.10	06.10	07.10	08.10	09.10	10.10	11.10	12.10	13.10	14.10	15.10	16.10	17.10	18.10	19.10	20.10	21.10	22.10	23.10	24.10	25.10	26.10	27.10	28.10	29.10	30.10	31.10	01.11	02.11	03.11	04.11	05.11	06.11
	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO
nes																													M2															M3						
															Concept Phase														Design Input																					
Detail Work	Military							Military							Requirements														Specification															Architecture						
Deliverables															Project Development Plan SW PRD (Product Requirements)														SWS (Specification) SSD (Structure Design) SDD (Detail Design) SW Configuration Management Plan SW GUI Design																																				

TECAN.

Project Time Overview for Parameterize Tool

	8							9							10							11							12							13							14						
	07.11	08.11	09.11	10.11	11.11	12.11	13.11	14.11	15.11	16.11	17.11	18.11	19.11	20.11	21.11	22.11	23.11	24.11	25.11	26.11	27.11	28.11	29.11	30.11	01.12	02.12	03.12	04.12	05.12	06.12	07.12	08.12	09.12	10.12	11.12	12.12	13.12	14.12	15.12	16.12	17.12	18.12	19.12	20.12	21.12	22.12	23.12	24.12	25.12
	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO	MO	DI	MI	DO	FR	SA	SO
nes															GUI Demo														M4														M5						
	Design Output Prototyp																												Validation / Testing																						
Detail Work	Basis Tool																												Test & Bugfixing																						
	GUI							DataStorage														GUI							Write SW Unit Test Case Plan							Implement and Execute Unit Tests							Documentation Finishing						
Deliverables																													SW Unit Test Case Report SW Unit Test Case Plan																						

Lesson Learned

1 Summary

The project has reached a state where the requirements where formally written down and a basic tool has been implemented. So the general goal was reached. On the other side many functions could not be handled because lack of time.

2 What Went Well

- The GUI elements were done in time. As my first bigger WPF project I had many things to learn and I did. Thanks to the World Wide Web questions could be answered really fast.
- The communication and help of the Team of the Tecan Base SDK went really good. Whenever I had a question, and somebody had time, I was given useful and realizable input.
- After the Tecan Base SDK was embedded properly the work with the firmware made fun. It just worked.

3 What Went Badly

- It is absolutely necessary to use enough time to get the requirements and specifications. Due to the challenging timeline not much time was planned for these documents.
- It is not easy to fit in the business process of Tecan into a 14 week project timeline. The different person's calendars are full of other appointments. It is difficult to organize a meeting.
- I had the approach to start the implementation with the requested granularity of the requirements. At the beginning all went right. But after 3 weeks I went busy because of conquered situations which came up more and more. They were not easy to handle and so I (1) wasted much time, (2) just leaved them out for the moment or (3) stuck me completely. After 3 weeks I used another approach and did not try to gain the little granularity but used the Tecan Base SDK. With this the general functions worked and I could go on. But from these 3 weeks about 2 were lost. The help from the Tecan Base SDK members should be used earlier to prevent such a situation.

Personal Statement

To make my student project at the same place I'm employed is an advantage. I was and I'm still very grateful for this opportunity.

The documentation part at the beginning was a rocket start, from zero to full speed at the very first day. The Tecan SOP has many documents on the list who wanted to be filled out. I did take the most important ones and started to write the content. But the SOP would have many more documentation forms and one working man alone has to have near infinite time to write all these stuff. But even the small subset I used was still much to do in a short amount of time. For example the requirements: I could write time in time and they were reviewed and approved by the customer, but during the implementation phase more and more other requirements popped up from other members of the customer's team or even from my investment of the current code. Some of these new requirements were just "complaints" that it has to be as it is now. With a good explanation such a statement could be "rejected". But others were very important. The time for gathering the right level of requirements with more interviews and questioning, the involvement of more (or better all) members of the customers team would have led to a better and more accurate result.

I could use the tools and libraries I had worked with in the last year. In my daily work at Tecan I use the Base SDK from one or more layers above and had a good general understanding how things work. But that does not mean that it was an easy task. I had to use these helpers on other ways as I did use them before and tried to execute other steps than the ones I used till now. At the end I learned much more about the tools and libraries from Tecan as I had known at the beginning and this learning will also help me a lot with my daily work. The understanding of single simple steps the Base SDK does (or does not) helps me prevent false behavior one the layers on top.

In the developing phase I made one mistake that used much time and gained me more or less nothing. One of the requirements is to be able to control the processes in very granular steps. I started to write down these small procedures. After I had done some of them I started to put them together and tried to use them on an instrument. But that didn't work out very well. Something was missing every time and at the end was had spent much time on "micro" debugging and bug fixing and it still didn't work properly. So I made a step back and used the Tecan Base SDK in these places. The steps are not granular enough, but it works for now. Later on these functionalities have to be split on the reach the required small procedure steps.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

COMPUTER SCIENCE

.NET Parameterize-Tool for Pipet-Instruments

Andreas Zollinger

Advisor: Joas Leemann

Co-Examiner: Prof. Dr. Luc Bläser

Project Partner: Tecan Schweiz AG

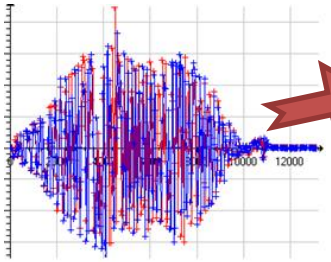
Project Thesis Autumn Term 2011

Subject Area: Software Engineering

Project Goal

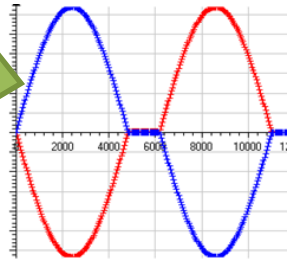
Design and implementation of a new parameterization tool for pipet instruments , replacing the current LUA script-based solution.

- Detailed requirements analysis
- Using C#, WPF and Tecan Base SDK
- Provide additional low-level functionality beyond the SDK



LUA Script Param Tool

- Has No Valid State
- Is Not Well Known
- Other Behavior Than Tecan Base SDK

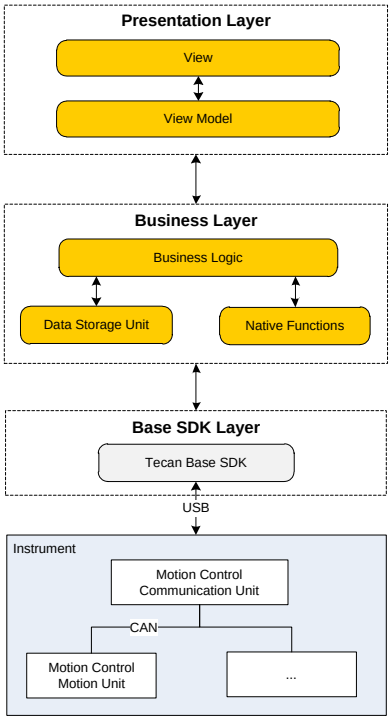


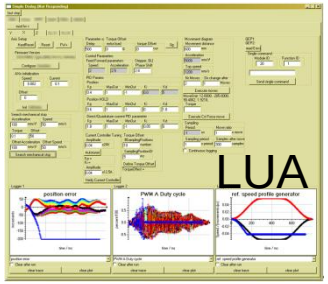
Tecan Base SDK Param Tool

- Provides Functionality to an Application SW
- Holding Configuration
- Communication with Firmware

Implemented Features

- Management of devices, axes, motors, encoders
- Movement of axes
- Visualization of diagnostic data
- Export to Tecan Base SDK configuration templates





Achieved Results

- Analysis and formal definition of detailed requirements
- First functional version of the new parameterization tool
- Verification and validation of the tool

Conclusions

- Parameterization enabled by a proper compiled C# application instead of a LUA script solution
- Consistent integration using the common Tecan Base SDK
- Flexible GUI maintenance in WPF

Future Work

- Increasing functionality coverage (special moves etc.)
- Automatic parameter versioning
- Multi-instrument support