Requirements document

SQUID

Helsinki 28th February 2005

Software Engineering Project

UNIVERSITY OF HELSINKI
Department of Computer Science

Course

581260 Software Engineering Project (6 cr)

Project Group

Mikko Jormalainen
Samuli Kaipiainen
Aki Korpua
Esko Luontola
Aki Sysmäläinen

Client

Lauri J. Pesonen

Project Masters

Juha Taina
Jenni Valorinta

Homepage

`http://www.cs.helsinki.fi/group/squid/`

Change Log

Version	Date	Modifications
0.1	9.2.2005	First version (Aki Sysmäläinen)
0.2	18.2.2005	Use case list (Samuli Kaipiainen)
0.3	21.2.2005	Some use cases expanded (Samuli Kaipiainen)
0.4	22.2.2005	User requirements (Mikko Jormalainen)
0.5	23.2.2005	More use cases expanded (Samuli Kaipiainen)
0.6	25.2.2005	All use cases expanded (Samuli Kaipiainen)
0.7	26.2.2005	More user requirements, architecture (Esko Luontola)
		Use case diagram (Samuli Kaipiainen)
0.8	27.2.2005	Use case -> requirements -references (Samuli Kaipiainen)
		UI protos, verified the use cases (Esko Luontola)
0.9	28.2.2005	Validation (Mikko Jormalainen)
		Requirement -> uce cases -references (Mikko Jormalainen)
		Requirement setters (Mikko Jormalainen)
		Overview (Aki Sysmäläinen)
		Requirement summary table (Samuli Kaipiainen)

Contents

1 Introduction **1**

1.1 Glossary 1

2 Overview **1**

3 Use cases **2**

3.1 Measuring 2

3.2 Exporting 6

3.3 Printing 7

3.4 Projects 7

3.5 Sequences 9

4 Requirements **12**

4.1 Functional requirements 12

4.1.1 Magnetometer 14

4.1.2 Project files 14

4.1.3 Measurements 16

4.1.4 Others 17

4.2 Quality requirements 17

4.3 Environment 18

4.4 Maintainability 19

5 User interface **19**

5.1 Goal-derived use cases 19

5.1.1 Goal-derived use case 1: Erkki... 19

5.1.2 Goal-derived use case 2: Fabio... 19

5.1.3 Goal-derived use case 3: Tomas... 19

6 Architecture overview **19**

7 External interfaces **20**

7.1 Existing program 20

7.2 Hardware control protocols 21

8 Validation **21**

Appendices

1 Automated Sample Handler System Protocol

2 Automatic Sample Degaussing System Protocol

3 Superconducting Rock Magnetometer Protocol

1 Introduction

This document describes client requirements and system requirements for a SQUID magnetometer program that will be designed and implemented as a software engineering student project at University of Helsinki at the Computer Science Department. The client is the Department of Geophysics.

This document serves as a contract between client and us about the implemented functions of the program. Particulary the requirements part of this document describes them.

Expected readership of this document here..

1.1 Glossary

Technical terms here..

2 Overview

Department of Geophysics uses a magnetometer that works in SQUID (superconducting quantum interference device) principle to measure magnetization of minerals and rocks. Magnetometer is controlled by a computer software that controls and reads electric components that control the device itself.

The use of the present magnetometer software is complicated and unnecessarily burdens the users work memory with complicated work phases and divergent operation in different use cases. The device is now controlled with two various program. Functions of those two program can be combined to one. Now the programs are hard to learn and teach which restricts the group of users.

User of the device must know clearly what is the state of it and which operation is being performed at the moment. The device is delicate and expensive, so a misfunction of it must be noticed early to prevent damage. For example faulty state of demagnetizer can damage the device if it's not shut down manually in time.

Device and program are used to carry out long, several stages involving measurement sequeances. It's important that these sequences can be performed flexibly and fast. User must be able to follow the measuring procedure and to stop and modify sequence if there is unexpected changes in results.

Further process and analyze of the results occurs in various other programs which use a standard file format. So the new program has to export these formats.

3 Use cases

Describes planned use cases for the program. Use cases are derived from user interface prototype and requirements. All use cases are perfomed by any ordinary user and in program main screen. A simplified use case diagram is presented in Figure 1.

Use cases are divided to different sections, grouping logically similar use cases together.

Use case format:

UC0: Use case identifier and title
[POSSIBLE TEMPORARY NOTE HERE FOR PROJECT GROUP; SHALL BE REMOVED FROM FINAL VERSION.]

Scenario a	First scenario for doing the use case.
Scenario b	Second scenario for doing the use case. ...
Precondition	Preconditions for use case.
Postcondition	Postconditions for use case.
Error condition	Error handling, mainly if anything special needs to be done.
Goal-derived	Goal-derived use case(s) in which this use case occurs (if any); see section 5.1.
Requirements	Requirement(s) from which this use case derives; see section 4.

3.1 Measuring

As in any and all measuring action with the squid.

UC1: Single step measuring without demagnetization

Scenario a	Enter as next AF demagnetization step "0" or empty (default for new projects), meaning no demagnetization, and click "Single step".
Precondition	AF project open, sample in sample holder.
Postcondition	Sample measured, results on screen and appended to current project.
Error condition	The program shall let the user know if something went wrong.
Goal-derived	5.1.3, first few steps.
Requirements	R1, R6, R10, R14, R17, R18

UC2: Single step measuring with (AF) demagnetization

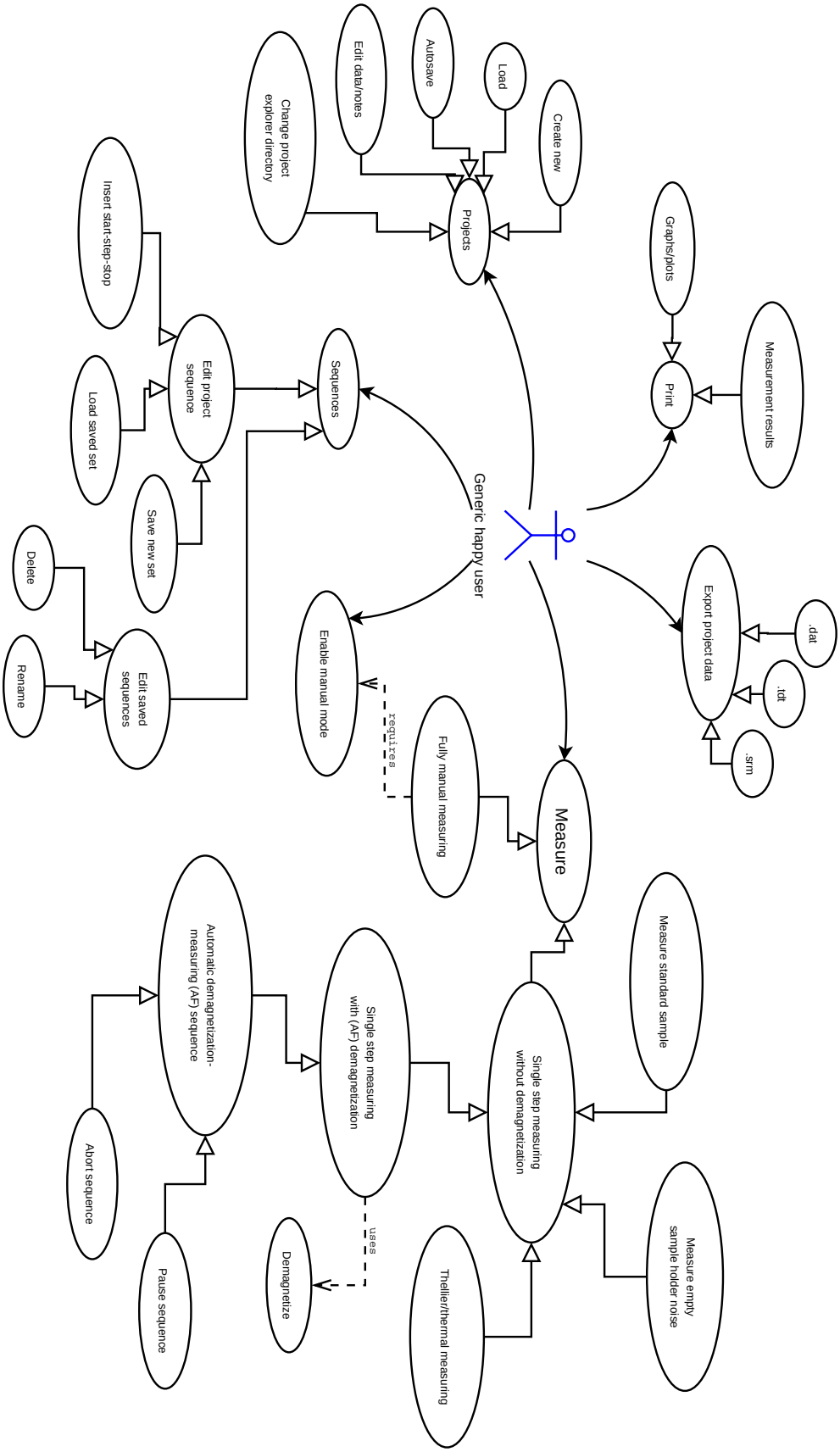

Figure 1: Use case diagram

Scenario a	Enter as next AF demag step anything greater than zero, and click "Single step".
Precondition	AF project open, sample in sample holder.
Postcondition	Sample demagnetized (possibly ruined) and measured, results on screen and appended to current project.
Error condition	The program shall let the user know if something went wrong, and, should the demagnetization field not be coming down, warn user with an alarm sound :)
Requirements	R1, R4, R6, R10, R14, R17, R18

UC3: Automatic demagnetization-measuring sequence (AF sequence)

Scenario a	Enter the AF sequence (see section 3.5 for ways to enter it) and click "Measure".
Precondition	AF project open, sample in sample holder.
Postcondition	Sample demagnetized according to entered AF sequence (possibly ruined) and measured after each demagnetization, results on screen and appended to current project.
Error condition	The program shall let the user know if something went wrong, and, should the demagnetization field be uncalm, warn user with an alarm sound x)
Goal-derived	5.1.1 whole sequence, 5.1.3 sequence after few single steps at the beginning.
Requirements	R1, R4, R6, R10, R13, R17, R18

UC4: Pause automatic measuring sequence

Scenario a	While measure sequence is running, click "Pause".
Precondition	Ongoing measuring sequence (UC3).
Postcondition	Measure sequence halts after current step is done, results on screen.
Error condition	Program tells if sequence can’t be paused (and something has gone terribly wrong).
Goal-derived	5.1.3, Tomas pauses the sequence as the meteor demagnetizes too rapidly.
Requirements	R1, R15

UC5: Abort automatic measuring sequence

Scenario a	While measure sequence is running or paused, click "Stop now!".
Precondition	Ongoing or paused measuring sequence (UC3).
Postcondition	Measure sequence halts immediately [AND PROGRAM ENTERS "FULLY MANUAL" MODE?]
Error condition	Program tells if sequence can’t be aborted (and something has gone terribly wrong).
Requirements	R1, R16

UC6: Thellier measuring

Scenario a	Click "Single step". (Temperature can be entered later, as it won't affect measuring.)
Precondition	TH project open, sample in sample holder.
Postcondition	Sample measured, results on screen and appended to current project.
Error condition	As usual.
Goal-derived	5.1.2, temperatures already entered, one step at 380 °C.
Requirements	R1, R6, R10, R14, R17, R18

UC7: Thermal measuring

[EXACTLY THE SAME AS THELLIER?]

Scenario a	Click "Single step". (Temperature can be entered later, as it won't affect measuring.)
Precondition	TH project open, sample in sample holder.
Postcondition	Sample measured, results on screen and appended to current project.
Error condition	As usual.
Requirements	R1, R6, R10, R14, R17, R18

UC8: Measure magnetometer ground noise

[2005-02-23 NOT IN CURRENT UI PROTO, NOR PLANNED FOR IMPLEMENTATION.]

Scenario a	Click "Ground noise" and "Calibrate".
Precondition	No ongoing measurements.
Postcondition	Ground noise measured, results on screen and appended to "Ground noise" project.
Requirements	R1, R3, R6, R17, R18

UC9: Measure empty sample holder noise

Scenario a	Click "Holder noise" and "Calibrate".
Precondition	No ongoing measurements, empty sample holder.
Postcondition	Holder noise measured, results on screen and appended to "Holder noise" project.
Goal-derived	5.1.1, calibration at the beginning.
Requirements	R1, R3, R6, R17, R18

UC10: Measure standard sample

Scenario a	Click "Standard sample" and "Calibrate".
Precondition	No ongoing measurements, standard sample in sample holder.
Postcondition	Standard sample measured, results on screen and appended to "Standard sample" project, and, predefined .std file.
Postcondition	Holder noise measured, results on screen and appended to "Standard sample" project.
Requirements	R1, R3, R6, R17, R18

UC11: Fully manual measuring

[2005-02-25 STILL NOT CLEAR HOW FULLY MANUAL IS SUPPOSED TO WORK; TOMAS WILL BE BACK ON TUESDAY 1.3. AND MIGHT TELL US.]

Scenario a	Click any of the manual control components.
Precondition	Manual mode enabled.
Postcondition	Manual action done, result on screen [AND APPENDED TO CURRENT PROJECT? DRAW GRAPHPLOTS?].
Requirements	R1, R2, R6, R10, R17, R18

UC12: Enable manual mode

Scenario a	Click "Manual" checkbox above the manual control components.
Precondition	No ongoing measurement.
Postcondition	Manual mode enabled.
Requirements	R2

3.2 Exporting

As in exporting measurement results to another file format for othrer programs to use.

UC13: Export project data into .dat file

Scenario a	In project explorer file list, right-click on desired project file, click "Export .dat in current directory".
Scenario b	In project explorer file list, right-click on desired project file, click "Export .dat to disk drive A:".
Scenario c	In project explorer file list, right-click on desired project file, click "Export .dat...", chooce directory and filename to export.
Scenario d	In project explorer file list, select multiple files with shift-click and ctrl-click, and make any of above actions.
Precondition	At least 1 project file in current (selected) directory.
Postcondition	Project data exported to .dat file.
Error condition	Notify if file error occurs (such as no disk in A: drive).
Requirements	R11

UC14: Export (thellier) project data into .tdt file

Scenario a	In project explorer file list, right-click on desired project file, click "Export .tdt in current directory".
Scenario b	In project explorer file list, right-click on desired project file, click "Export .tdt to disk drive A:".
Scenario c	In project explorer file list, right-click on desired project file, click "Export .tdt...", chooce directory and filename to export.
Scenario d	In project explorer file list, select multiple files with shift-click and ctrl-click, and make any of above actions.
Precondition	At least 1 project file in current (selected) directory.
Postcondition	Project data exported to .tdt file.
Error condition	Notify if file error occurs (such as no disk in A: drive).
Requirements	R11

UC15: Export single measurement details into .srm file

Scenario a	In measurement result table, right-click on desired measurement line(s), click "Export .srm in current directory".
Scenario b	In measurement result table, right-click on desired measurement line(s), click "Export .srm to disk drive A:".
Scenario c	In measurement result table, right-click on desired measurement line(s), click "Export...", chooce directory and filename to export.
Scenario d	In measurement result table, select multiple lines with shift-click and ctrl-click, and make any of above actions.
Precondition	At least 1 measurement result in current project.
Postcondition	Measurement details exported to .srm file.
Error condition	Notify if file error occurs (such as no disk in A: drive).
Requirements	R11

3.3 Printing

As in printing measurement results as text or graphs.
[2005-02-25 NOT IN CURRENT UI PROTOTYPE, IMPLEMENTATION PRIORITY LOW.]

UC16: Print measurement results

Scenario a	Click "Print...", "Measurement results".
Precondition	Open project.
Postcondition	Measurement results printed via [Java] standard printing window.
Error condition	Let know if printing error occurs.
Requirements	R17

UC17: Print graph sheet (with 7 different graphs; described elsewhere)

Scenario a	Click "Print...", "Grap sheet".
Precondition	Open project.
Postcondition	Measurement results printed via [Java] standard printing window.
Error condition	Let know if printing error occurs.
Requirements	R18

3.4 Projects

As in project files, which store all measurement results.

UC18: Automatically save all measurement cycles in project file

Scenario a	Make any measurement action (see section 3.1 for those).
Precondition	Project file open.
Postcondition	After measurement (step) is done, new results appended to project file.
Goal-derived	5.1.1, 5.1.2, 5.1.3.
Requirements	R6, R10

UC19: Create new project

Scenario a	Click the empty line below filenames in project explorer, enter new project name, chooce AF/TH (thellier) project, click "Create new" or press enter.
Precondition	Project explorer in desired directory (UC21).
Postcondition	New project created and selected.
Goal-derived	5.1.1 after calibration, 5.1.3 at the beginning.
Requirements	R5

UC20: Load project

Scenario a	Click any filename in project explorer.
Precondition	Project explorer in desired directory (UC21). A measurement can be running at the same time (it will be highlighted in the project explorer).
Postcondition	Existing project loaded and selected.
Goal-derived	5.1.2 all samples already as project files.
Requirements	R7, R10

UC21: Change project explorer directory

Scenario a	Click into current directory textbox in project explorer, write directory to change to. If the typed directory exists, the program will open it automatically. [OR WILL A PUSH OF ENTER BE ANYWAY REQUIRED TO CHANGE DIRECTORY? SHOULD WE TRY BOTH BEFORE DECIDING?]
Scenario b	Click into current directory textbox in project explorer, start writing directory; when autocomplete-results appear below, use up/down+enter or mouseclick to select the directory.
Scenario c	Click the down-arrow on the right side of current directory textbox in project explorer, choose desired directory from appearing directory history.
Scenario d	Click "Browse..." in project explorer, use standard [Java] file chooser to select directory to change to.
Precondition	There is no measurement running right now (to prevent the user from getting lost).
Postcondition	The contents of the chosen directory are shown in the project explorer. The newest project file in the directory will be selected and loaded.
Error condition	If the user has typed an invalid directory, then a "No such directory" message will be displayed in place of the autocomplete-results.
Goal-derived	5.1.1, 5.1.2, 5.1.3.

UC22: Insert/edit project data

Scenario a	Click any of the project data checkboxes, or textboxes and enter value.
Precondition	Project file open (usually just created it).
Postcondition	Project data changed, measuring results recalculated, saved automatically to project file. Saving can happen after a short delay to avoid unnecessary file system access.
Goal-derived	5.1.1 after creating new project.
Requirements	R6, R8, R9

3.5 Sequences

As in automatic demagnetization-measuring sequences (AF sequences), or, thellier temperature sequences.

UC23: Insert sequence manually

Scenario a	Click "Tesla" or "Temp" column in the last line (which is empty) of measurement result table, enter value; press down, [tab?] or enter to advance to a new line, enter value; and so on.
Precondition	An open project.
Postcondition	Sequence appended to measurement result table and current project.
Goal-derived	5.1.3 after creating new project.
Requirements	R6, R12

UC24: Insert sequence with start-step-stop values

Scenario a	Click "Start" textfield, insert start value (default value is the last value of the Tesla/Temp column), click "Step" textfield, insert step value, click "Stop" textfield, insert stop value, click "Add sequence" or press enter.
Scenario b	Click "Start" textfield, insert start value (default value is the last value of the Tesla/Temp column), press tab, insert step value, press tab, insert stop value, press enter or click "Add sequence".
Precondition	An open project.
Postcondition	Sequence from *start* to *stop*, increasing by *step* for every step, appended to measurement result table of the current project. Value of "Start" textfield is the previous value of "Stop". "Step" and "Stop" textfields are empty. "Step" textfield has focus.
Error condition	If the last value in the old sequence (excluding completed measurements) is equal to the *start* value of this new sequence, then start the new sequence from *start + step*.
Goal-derived	5.1.3 after creating new project and entering manual values.
Requirements	R6, R12

UC25: Load sequence

Scenario a	Click down-arrow in right side of "Load set" combo box; from appearing list, click sequence to load.
Precondition	An open project, at least 1 saved sequence.
Postcondition	Selected sequence appended and selected to measurement result table, and appended to current project.
Goal-derived	5.1.1, after typing project data, load "Basalt" set; 5.1.2 set already loaded for each project file (when created).
Requirements	R6

UC26: Edit sequence on-the-fly

Scenario a	Click any unmeasured "Tesla" or "Temp" column in measurement result table, enter new value.
Scenario b	Click any unmeasured line in measurement result table, press del to delete that line.
Scenario c	Right-click any unmeasured line in measurement result table, choose "Delete" to delete that line.
Scenario d	Click-drag any unmeasured line in measurement result table to new position within unmeasured lines.
Scenario e	Select multiple unmeasured lines in measurement result table with shift-clicks and ctrl-clicks, make any of above actions (except enter new value).
Precondition	Unmeasured lines in current project's measurement result table (note that measuring sequence can be going on).
Postcondition	Editing committed, saved to current project.
Error condition	Editing or screwing up measured lines won't be allowed.
Goal-derived	5.1.3, when sequence demagnetizes too rapidly, Tomas deletes unmeasured lines.
Requirements	R6

UC27: Save sequence

Scenario a	Click "Load set" combo box, enter new sequence name, click "Save set". [2005-02-25 NOT IN CURRENT (ALMOST FINAL) UI PROTOTYPE!]
Scenario b	Right-click any line in measurement result table, choose "Save full sequence...", enter name, press enter.
Scenario c	Select any lines in measurement result table with shift-clicks and ctrl-clicks, right-click on selected lines, choose "Save selected sequence...", enter name, press enter.
Precondition	At least 1 line in measurement results table (although you propably don't want to save a sequence with only one step x).
Postcondition	New sequence set saved to predefined sequence file and available from "Load set" combo box.
Error condition	Ask whether to overwrite, if sequence with the same name already exists (allow to enter new name if choose not to overwrite).

UC28: Edit stored sequence

Scenario a	UC25 "Load sequence" -> UC26 "Edit sequence" -> UC27 "Save sequence" with same name as the loaded sequence.
Scenario b	Click menu "Options"->"Sequences...", edit any sequence in appearing window... [2005-02-25 NOT IN UI PROTOTYPE, MUST BE IMPROVISED :)]
Precondition	At least 1 saved sequence.
Postcondition	Changes saved to predefined sequence file.

UC29: Delete stored sequence

Scenario a	Click down-arrow in right side of "Load set" combo box; from appearing list, right-click sequence to delete, choose "Delete".
Scenario b	Click menu "Options"->"Sequences...", delete sequence in appearing window... [2005-02-25 NOT IN UI PROTOTYPE, MUST BE IMPROVISED :)]
Precondition	At least 1 saved sequence.
Postcondition	Sequence deleted, changes saved to predefined sequence file.

UC30: Rename stored sequence

Scenario a	Click down-arrow in right side of "Load set" combo box; from appearing list, right-click sequence to rename, choose "Rename...", enter new name, press enter.
Scenario b	Click menu "Options"->"Sequences...", rename any sequence in appearing window... [2005-02-25 NOT IN UI PROTOTYPE, MUST BE IMPROVISED :)]
Precondition	At least 1 saved sequence.
Postcondition	Sequence renamed, changes saved to predefined sequence file.

4 Requirements

Goals of the software set by client, project team and environment. Requirements were collected from 1) powerpoint slides received from client, 2) lectures given by client, 3) user observations held at the squid lab with client (used to collect goal-derived use cases), and 4) user interface prototype demonstrations given to client, after which the client gave suggestions and corrections to prototype functions.

Priorities are from 1 (highest) to 4 (lowest). Our goal is to fulfill at least the requirements with priorities 1 and 2. Requirements with priority 3 will be fulfilled only if time permits, and priority 4's will propably be left out.

Overview of requirements is presented in Table 1.

Requirement format:

Identifier: Requirement identifier
Name: Name of the requirement
Description: Description of the requirement.
Priority: Requirement priority from 1 (most important) to 4 (least important)
Set by: Who set the requirement; client, project team or environment
Use Cases: Use cases derived from this requirement

4.1 Functional requirements

Define the services which the software offers and how the software behaves, including error handling. May also define things the software shall not do.

Identifier	Name	Priority	Set by
R1	SQUID control and usage	1	Customer
R2	Manual control	3	Customer
R3	Calibration reminder	2	Customer
R4	Warning signal	3	Customer
R5	Create project	1	Customer
R6	Autosave project	1	Customer
R7	Load project	1	Customer
R8	Edit project	2	Customer
R9	Recalculate derived measurement data	2	Customer
R10	Append project	2	Customer
R11	Export to other file formats	1	Customer
R12	Create a measurement sequence	2	Customer
R13	Automatic sequence	2	Customer
R14	Single step sequence	2	Customer
R15	Pause automatic sequence	2	Customer
R16	Panic abort	1	Customer
R17	Numeric presentation of measurements	1	Customer
R18	Graphic presentation of measurements	4	Customer
R19	Hotkeys	3	Customer
R20	Change hotkeys	4	Customer
QR1	Error free	1	Customer
QR2	Ease of use	1	Customer
QR3	Quality of documentation	2	Group
QR4	Performance	2	Group
QR5	Expandable architecture	2	Group

Table 1: Requirements summary

4.1.1 Magnetometer

Identifier: R1
Name: SQUID control and usage
Description: Able to control Squid-magnetometer and make measurements with it.
Priority: 1
Set by: Customer
Use Cases: UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8, UC9, UC10, UC11

Identifier: R2
Name: Manual control
Description: Able to operate the magnetometer manually. User can move and rotate sample handler and measure and demagnetize sample.
Priority: 3
Set by: Customer
Use Cases: UC11, UC12

Identifier: R3
Name: Calibration reminder
Description: Magnetometer must be calibrated every 24 hours. The program will remind the user when it would be time to do calibration.
Priority: 2
Set by: Customer
Use Cases: UC8, UC9, UC10

Identifier: R4
Name: Warning signal
Description: Program should warn user with alarm signal if magnetic field remains on for too long. The implementation depends on if the Degaussing System's protocol makes it possible.
Priority: 3
Set by: Customer
Use Cases: UC2, UC3

4.1.2 Project files

Identifier: R5
Name: Create project
Description: Create a new project file which will include the measurement sequence, measurement results and information about the sample. The file format will be custom made for this program.
Priority: 1

Set by: Customer
Use Cases: UC19

Identifier: R6
Name: Autosave project
Description: Program will save measurement data and project information after every measurement step and modification.
Priority: 1
Set by: Customer
Use Cases: UC1, UC2, UC3, UC6, UC7, UC8, UC9, UC10, UC11, UC18, UC22, UC23, UC24, UC25, UC26

Identifier: R7
Name: Load project
Description: Saved projects can be loaded into program.
Priority: 1
Set by: Customer
Use Cases: UC20

Identifier: R8
Name: Edit project
Description: Ability edit project information and measurement data afterwards. (TODO: which data fields?)
Priority: 2
Set by: Customer
Use Cases: UC22

Identifier: R9
Name: Recalculate derived measurement data
Description: When the measurement data or the mass/volume of the specimen is changed, recalculate all numbers that have been derived from the modified data.
Priority: 2
Set by: Customer
Use Cases: UC22

Identifier: R10
Name: Append project
Description: New measurements can be added to existing projects.
Priority: 2
Set by: Customer
Use Cases: UC1, UC2, UC3, UC6, UC7, UC11, UC18, UC20

Identifier: R11

Name: Export to other file formats
Description: Measurements can be saved in .dat, .tdt and .srm files.
Priority: 1
Set by: Customer
Use Cases: UC13, UC14, UC15

4.1.3 Measurements

Identifier: R12
Name: Create a measurement sequence
Description: Able to create AF and Thellier measurement sequences with several different sized steps.
Priority: 2
Set by: Customer
Use Cases: UC23, UC24

Identifier: R13
Name: Automatic sequence
Description: Do all the measurement steps automatically.
Priority: 2
Set by: Customer
Use Cases: UC3

Identifier: R14
Name: Single step sequence
Description: Do only one measurement step at a time.
Priority: 2
Set by: Customer
Use Cases: UC1, UC2, UC6, UC7

Identifier: R15
Name: Pause automatic sequence
Description: Able to stop the measurement sequence after current step. After being stopped, the user can make modifications to the sequence and continue from where the sequence was left.
Priority: 2
Set by: Customer
Use Cases: UC4

Identifier: R16
Name: Panic abort
Description: Able to stop any measurement immediately. The demagnetizer will be

turned off and the sample holder will stop where it is. The current measurement data will be discarded.
Priority: 1
Set by: Customer
Use Cases: UC5

Identifier: R17
Name: Numeric presentation of measurements
Description: Program shows measurement data in numbers. Measurement data will be displayed using scientic notation (1.23e4).
Priority: 1
Set by: Customer
Use Cases: UC1, UC2, UC3, UC6, UC7, UC8, UC9, UC10, UC11, UC16

Identifier: R18
Name: Graphic presentation of measurements
Description: Program draws graphs from measurement data. Available graph types are: Stereo Plot, Intensity, Zijderveld, Difference Vectors, Susteptibility, O63, Great Circles. (TODO: what are the definitions of these graphs? priorities per graph?)
Priority: 4
Set by: Customer
Use Cases: UC1, UC2, UC3, UC6, UC7, UC8, UC9, UC10, UC11, UC17

4.1.4 Others

Identifier: R19
Name: Hotkeys
Description: The program has hotkeys for the most often used operations.
Priority: 3
Set by: Customer

Identifier: R20
Name: Change hotkeys
Description: Possibility to change hotkey assignments.
Priority: 4
Set by: Customer

4.2 Quality requirements

[*Some definition/explanation here?*]

Identifier: QR1
Name: Error free
Description: The program should not crash. All the functions need to work as documented. The calculations that the program makes from measurements must be done right.
Priority: 1
Set by: Customer

Identifier: QR2
Name: Ease of use
Description: Program should be easy to use for first time users after it has been explained to them how to take measurements with the magnetometer.
Priority: 1
Set by: Customer

Identifier: QR3
Name: Quality of documentation
Description: The documentation must be so good that other teams can easily continue the development. Program should have good help pages for the end-users.
Priority: 2
Set by: Group

Identifier: QR4
Name: Performance
Description: The program must keep up with the SQUID hardware. The user interface should not freeze for periods of over 0,5 seconds.
Priority: 2
Set by: Group

Identifier: QR5
Name: Expandable architecture
Description: It should be possible to expand the program to process the measurement data in more ways.
Priority: 2
Set by: Group

4.3 Environment

Define requirements for the software which origin from the environment in which the sowtware will be working, and which the software must satisfy for being able to work in its environment.

Program will be used in normal PC which is connected to magnetometer. The computer

that will run this program will be equal or better to 1GHz CPU, 256MB RAM, 1280x1024 resolution. The current computer is running Windows XP.

Taking into account rapid phase of computer evolution it is possible that computer in which program is used can change, accordingly the program should be installable by outsiders. We need not prepare to changing of magnetometer, as new magnenetometer will probably have its own program. Program doesn't control the temparature of the magnetometer.

The SQUID hardware and communication with it is described in the section "External interfaces".

4.4 Maintainability

It must be possible to continue the development of the program. The documentation must be complete so that other teams can quickly continue the development even if they have not studied user interface designing. It must be possible to add new graph types to the program and export the measurement data to other programs.

5 User interface

Overview of UI described here..

5.1 Goal-derived use cases

5.1.1 Goal-derived use case 1: Erkki...

5.1.2 Goal-derived use case 2: Fabio...

5.1.3 Goal-derived use case 3: Tomas...

6 Architecture overview

The program architecture can be divided into three parts: SQUID interface, measurement project and user interface. The program will also communicate with the SQUID hardware and local file system. A graphical representation of the architecture can be seen in Figure 2.

SQUID interface is responsible for controlling the hardware in an orderly manner. It will provide high- and low-level controls for using the hardware. Communication with the hardware is done via COM ports. The SQUID interface will hide the protocol-level commands from the programmer and prevent illegal use of the hardware.

Measurement project is responsible for managing the project information, measurement

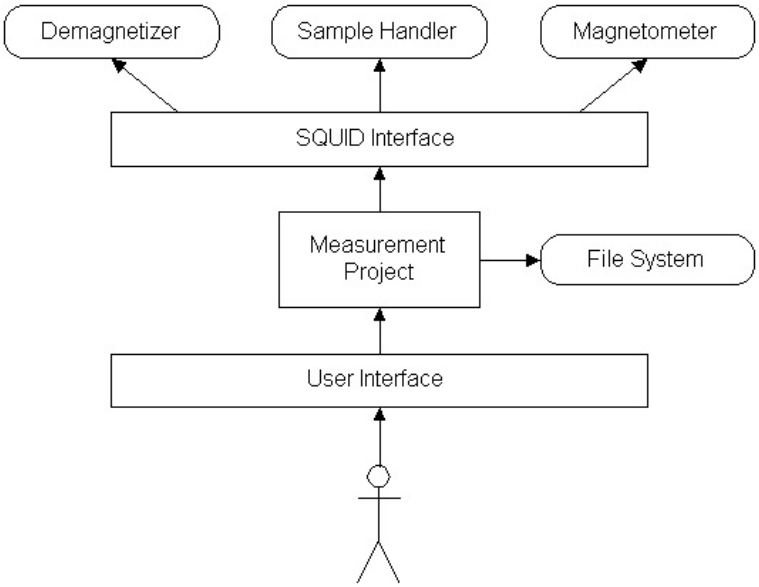

Figure 2: Architecture overview

sequence and measurement data. It will recieve commands from the user interface and notify the UI when the state of the project changes. It will send commands to the SQUID interface, recieve measurement data and save it. When the internal data of the project changes, the copy on the local file system will be automatically synchronized after a short delay (1 second or less).

User interface is responsible for communicating with the user of the program. It will update itself whenever the state of the measurement project is changed. It will send commands from the user to the program.

7 External interfaces

Interfaces to existing software and hardware are described here.

7.1 Existing program

The existing software for using the SQUID is "2G Enterprises Data Acquisition". We have access to the source code for version 2.99.3 of the program. From the old source code we will reuse basically only the SerialIO component. We will build an interface for communicating with the SQUID hardware by using Java and JNI (Java Native Interface).

7.2 Hardware control protocols

The SQUID consists of three independent units:

- Automated Sample Handler System (MODEL 2G800)
- Automatic Sample Degaussing System (MODEL 2G600)
- Superconducting Rock Magnetometer (MODEL 755R or 760R)

Automated Sample Handler System controls the movement and rotation of the sample holder. Its protocol is described in Appendix 1.

Automatic Sample Degaussing System controls the demagnetizer. Its protocol is described in Appendix 2.

Superconducting Rock Magnetometer reads the measurements from the magnetometer. Its protocol is described in Appendix 3.

8 Validation

In validation it is shown that program meets requirements set by user. Project documents are validated in reviews. Program code is validated by testing. In reviews it will be established that requirement document and design document are in line with user requirements and will lead to production of program which meets these requirements. Testing shows that program actually works as specified by design document.

Appendix 1. Automated Sample Handler System Protocol

Korvaa tämä sivu tiedostolla "Automated Sample Handler System - Protocol.pdf"

Appendix 2. Automatic Sample Degaussing System Protocol

Korvaa tämä sivu tiedostolla "Automatic Sample Degaussing System - Protocol.pdf"

Appendix 3. Superconducting Rock Magnetometer Protocol

Korvaa tämä sivu tiedostolla "Superconducting Rock Magnetometer - Protocol.pdf"