Verification and Validation Report: Measuring Microstructure Changes During Thermal Treatment

Team #30, ReSprint
Edwin Do
Joseph Braun
Timothy Chen
Abdul Nour Seddiki
Tyler Magarelli

March 8, 2023

1 Revision History

Date	Developer	Change
Date 1	1.0	Notes
Mar. 8, 2023	Joseph Braun	Added Sections 5, 7, 8
Mar. 8, 2023	Joseph Braun	Added Reflection

2 Symbols, Abbreviations and Acronyms

symbol	description
T	Test

[symbols, abbreviations or acronyms – you can reference the SRS tables if needed —SS]

Contents

List of Tables

List of Figures

This document ...

3 Functional Requirements Evaluation

4 Nonfunctional Requirements Evaluation

4.1 Usability

Usability Tests				
Requirement	Related Unit Tests	Description	Expected Result	Result
Afghanistan	AF	AFG	004	PASS
Aland Islands	AX	ALA	248	
Albania	AL	ALB	008	
Algeria	DZ	DZA	012	
American Samoa	AS	ASM	016	
Andorra	AD	AND	020	
Angola	AO	AGO	024	

4.2 Performance

4.3 etc.

5 Comparison to Existing Implementation

Below is an image of the exisiting implementation's GUI.

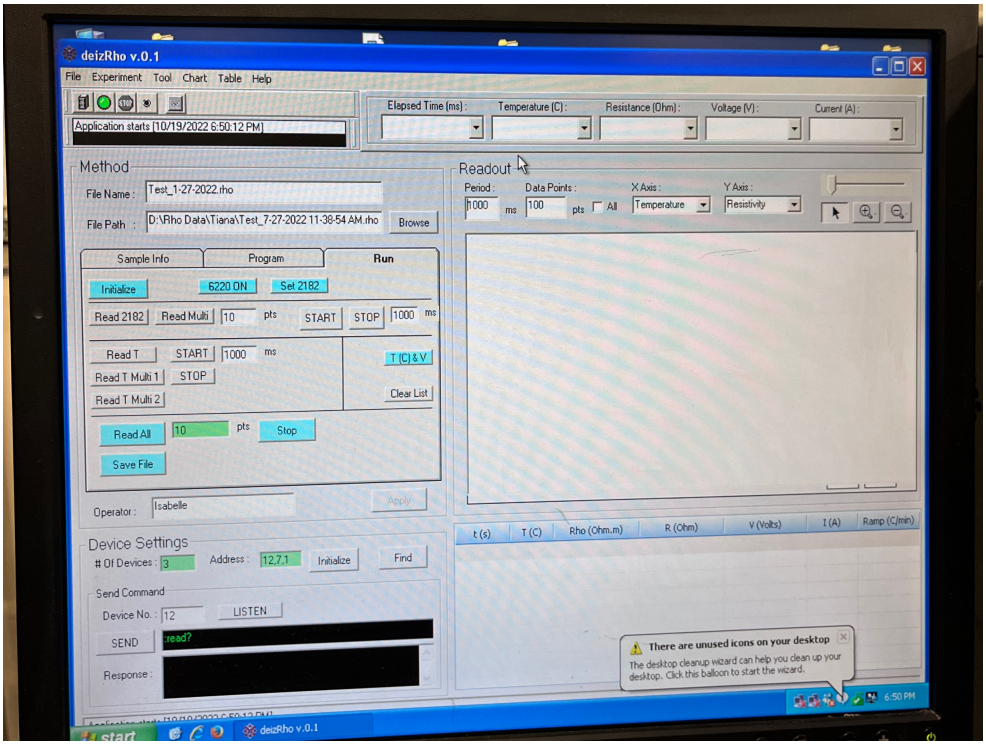

Figure 1: Previous software user interface design

The key elememts of the existing implementation will also be included in our implementation. This is so that the application will be familiar to the user and as intuitive as possible. These elements are listed below:

- Method Panel: controls which device to read data from (temperature or voltage), and which file to save the data in
- Device Settings Panel: used to send SCPI commands directly to a device
- Readout: includes graphical output and listed output of relevant values (current, voltage, resistance, etc.)

The primary differences between the existing implementation and our implementaion are the appearance of the GUI and the option of remote access. The existing implementation was developed for Windows XP whereas our implementation is developed for Windows 10, which gives it an updated look. One of the stretch goals for this project is to enable remote access to the application to be able to monitor and stop experiments remotely. The existing implementatio does not have this functionality.

6 Unit Testing

7 Changes Due to Testing

Based on the feeback from our Rev 0 Demo, we found that our application failed some requirements tests. The major failures were the lack of two main features: the graphical output (which still did not display real data as of Rev 0 Demo) and the ability to save output data to a chosen file. While we did not need to "test" our application to know this, we still consider this to be a failed test, as our application failed to meet requirements set out by the project supervisor. These two features have been implemented for the final demo.

Test ID	Failure Observed	Change Made
ST6	Graph only displays dummy data	Bug with displaying real-time data was fixed
ST7	There is not method to output data to a file	File system browser and writing to output file was added

8 Automated Testing

We acheived automated unit testing through the use of the NUnit testing framework in Visual Studio. NUnit is one of the most popular test frameworks used for running tests on a .NET project.

NUnit tests are setup by first creating a new project file in Visual Studio and adding it to the solution file for your project (in our case, the application). In the new project file, a new class is created. Each unit test we want to carry out is written as a method of the test class. Since the project file for the test class is inlcuded in the same solution file as our application, we are able to call the test methods from our main application to run the tests.

9 Trace to Requirements

10 Trace to Modules

11 Code Coverage Metrics

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. In what ways was the Verification and Validation (VnV) Plan different from the activities that were actually conducted for VnV? If there were differences, what changes required the modification in the plan? Why did these changes occur? Would you be able to anticipate these changes in future projects? If there weren't any differences, how was your team able to clearly predict a feasible amount of effort and the right tasks needed to build the evidence that demonstrates the required quality? (It is expected that most teams will have had to deviate from their original VnV Plan.)

There were several differences between what we had planned for VnV and what we ended up carrying out. When completing the initial revision of VnV plan, we had not yet completed the Design documents (MG, MIS, System Design) and so we did not have a plan for unit/module testing, only for system testing. Our initial VnV plan included three main sections: SRS Verification, Design Verification, and Implementation Verification.

For SRS Verification, we planned to meet every two weeks to discuss potential updates to the SRS doc. While we continued to meet frequently (weekly or bi-weekly), our team focussed instead on the next deliverable rather than revisiting old documents during meetings. Because of this, our SRS was not continually being updated during the project. This change occurred mainly due to time constraints, as we faced some technical issues leading up to the first demo which took priority over other tasks. In hindsight, frequently revisiting and updating the SRS certainly would have created less work for us in the long run. For future projects, though we can't predict any exact technical issues that would set us back on time, we should be able to anticipate issues arising which cause delays. We should have a plan to stay on schedule despite such issues.

For Design Verification, we planned to use the MIS checklist to ensure that requirements in the SRS are met and hazards in the Hazard Analysis are covered. Our team followed the MIS checklist when testing. We also planned to use feedback from the course instructor, teaching assistants, classmates, and our project supervisor. There were not many changes made in this section of the plan, except that our team decided to focus primarily on feedback from Dr. Zurob, as he is the end-user of the application.

For Implementation Verification, we planned to used GitHub issues and pull requests to maintain our code base. Any pull request made to the main branch requires at least two other team members to review and approve. We did not make any major changes to our Implementation Verification.