LAFAYETTE
COLLEGE

Maintainability Program Plan for:
<u>LFEV – SP17 – Y5</u>

Version: 0.1

Approval date: 02/22/2017

DOCUMENT CONTROL PANEL		
File Name:	Maintainability Program Plan Template	
File Location:	VSCADA Git repo	
Version Number:	1	
	Name	**Date**
Created By:	Martin Townley	02/10/2017
Reviewed By:		
Modified By:	Kemal Dilsiz	02/18/17
Approved By:		

Table of Contents

1 Overview 1
- *1.1 Scope* 1
- *1.2 Purpose* 1
- *1.3 Reference Documents* 1

2 General Requirements 4

1.1 Software Installation on Hardware 5
1.2 Errors, exceptions and logs 5
1.3 Backup and Restoring 5
1.4 Deployment on new Hardware 5
- *1.5 Log file trimming* 5
- *1.6 System API* 5
- *1.7 System Configuration Maintainability* 5
- *1.8 System Configuration Checking* 5
- *1.9 Tool Chain, Design Suite* 5
- *1.10 Third Party Software* 5
- *1.11 Requirements of GPR007* 6

1 Overview

1.1 Scope

This Reliability and Maintainability (R&M) Program Plan (RMPP) describes the necessary tasks, responsibilities, and controls that should be implemented in the Lafayette Formula Electric Vehicle Project.

The primary function of the R&M effort is to document the procedures; ensure both high operational readiness and availability; and minimize life-cycle cost. The RMPP should address the aspects of the design and engineering in relation to:

- Management
- Schedule
- Analytical tasks
- Control tasks
- Evaluation tasks
- Design

1.2 Purpose

The purpose of the RMPP is to:

- Define the R&M tasks to be accomplished
- Define the R&M organization and its interfaces to the engineering program and other support organizations
- Define the R&M management and control processes
- Identify, describe, and schedule the deliverable documentation
- Describe maintainability qualification testing
- Describe reliability qualification testing
- Identify reporting requirements necessary for logistic support analysis
- Describe the maintenance data collection and reporting system

2 General Requirements

The vehicle for commitment to effective R&M engineering is the R&M program plan developed for the project. The RMPP should emphasize early participation commencing with requirements definition and system development, followed by a comprehensive test, corrective action, and demonstration program to identify and correct deficiencies as required. The RMPP should be implemented at the onset of a development and subcontractor/vendor selection process.

The R&M program should cover the following major elements:

1. How will the software be installed on new hardware? What happens if the hardware goes obsolete?
2. How are errors and exceptions handled? How are logs viewed? How are exceptions configured and modified as requirements change?
3. How is backup performed? What is the restore procedure?
4. How is a fresh system deployed and validated on new hardware?
5. Are system logs and data files automatically trimmed? On do they grow and require manual trimming or offloading? If so, how is this accomplished?
6. What is the design of the system API and how will this design support ongoing reliable operation, maintenance and expansion?
7. How is system configuration maintained? Will the system auto detect hardware configuration changes or will configuration maintenance be required? If the latter, what is the consequence of misconfiguration? How will the software function when only some of the system hardware is available? Are demo or simulation stubs available for major hardware?
8. How is system configuration checked? Are tools provided for generating valid configurations?
9. What tool chain will be used? Is the tool suite up-to-date and actively supported? Is the tool suite mature enough to have stable functionality? How is the tool chain installed in a new development system.
10. What third party software will be incorporated into the system? How will this be maintained, upgraded, or patched during the life of the system.
11. How are requirements in GPR007 met?
    a. All software source code must be maintained under configuration control. Release snapshots must be archived on the project website.
    b. The system must start from cold power-up and boot to full operational status without requiring user interaction beyond enabling power and safety procedures
    c. Any PC software must be packaged for installation with a SETUP.EXE, RPM, “make install” or equivalent installer allowing it to be installed easily on any compatible computer.
    d. Configuration parameters, calibration factors, preferences, and options shall not be hardcoded within the software source code. It shall be possible to alter these various factors without recompiling software or physically disassembling hardware. Altered configuration parameters must be persistent through power cycling and reboots. The system must have a function to initialize itself with sane (factory default) configuration content if requested.

e. All data and configuration files must be in a generally supported format (e.g. XML) or the format required by a mature and well supported application (e.g. MySQL database files, Berkeley db, etc…). Files shall be accessible either through removable media or network file transfer or both.

2.1 Software Installation on Hardware

The Android application will be packaged in an Android application package (APK) and this format allows the user to easily install the application to an Android based operating system. This apk file only needs to be transferred to the tablet/phone and the android system will install it if the file is started.

2.2 Errors, exceptions and logs

Errors and exceptions are not very critical for the cell phone application as we only have read only access to the data from SCADA. Therefore, in case of an error for accessing the data, the application will store a log about the details of the error. There shouldn't be any errors while generating the views for the data after the application is deployed.

The logs will be stored on the android device and can be viewed easily through the Log View that will be implemented in the system.

Exceptions will be tested throughly with unit testing but in case of an exception or need of a modification, the code will be on a version control platform and can be fixed by a future team.

2.3 Backup and Restoring

The software will generate the views every time it is launched. This makes sure that the application will not need a continuously running function which requires constant backup. However, the versions of the software will be stored in a version control platform. This makes it possible for future teams to create an apk for a different version of the software and be able to experiment with branching. Therefore, the backup will be done while coding through version control and the restore procedure will be deploying an apk of a different version.

2.4 Deployment on new Hardware

Through the apk file.

2.5 Log file trimming

Log files will be automatically trimmed and will store upto 5mb of data. This is a ridiculous amount of space for the simple logging that the application will do. If wanted, the log files can manually be copied from the android device to another storage. An automatic process for saving the log files beyond 5 mb currently is a low priority of the design.

2.6 System API

What is the design of the system API and how will this design support ongoing reliable operation, maintenance and expansion?

The system will have a graphical user interface generated through the android studio tools. We hope to build an innovative environment for users where they can generate a look special to their need. Configuration of the API is one of the biggest aims of the application.

First of all, we want to make it possible for the application to run both on tablets and phones smoothly without any problems. There are multiple tools in Android Studio to make this possible. The android layout tools makes it possible to have multiple displays for different situations. For differently sized screens, different displays can be generated.

Also, it is possible to use “dp” for size of the layout. Dp makes it possible to calculate a certain size according to the relative size of the screen. Therefore, a chart can take 50% of the screen size with only one layout code.

These two tools makes it possible to generate displays that will support not only both tablets and mobile devices but at the same time it will make it possible to generate displays that are similar in all devices. Additionally, we hope to make it possible to zoom in the layout.

Another very important aspect of the API is the expandability. If we were to add more components to the LFEV system, how will the cell application update the display accordingly. Will it be able to adapt?

For this situation, we decided to make it possible for the user to generate their own displays. The incoming data will be stored in a certain way to make it possible to generate your own displays according to the options made available by us.

The user can choose their x-axis and y-axis together with adjustable time for a specific chart. The use can generate a speedometer and decide on the lower and higher limit and see simultaneous update. The user can choose to see the log messages according to a certain time.

Finally, another feature we are hoping to implement will make this application API much more flexible. This is to be able to generate your own display. Instead of user displaying a single part of the data, for example only one chart of Speed vs Time, the user can create their own displays. For instance, on a tablet, the user can first make a chart of SOC vs time and put this chart on top left of the display. Then the user could create a speedometer and put it on bottom left. And lastly, they could display the log messages on the right of the screen. Such generation of displays would give the user full control over what they want to see. And if new components are added to the system, new system API possibilities would be generated automatically by the application.

2.7 System Configuration Maintainability

The system configuration will be in two major steps. First step will be how the system will adapt to the changes in the hardware and the transfer of the data. This will make it possible for the application to generate new displays without ever needing to change the code. The second part of the maintainability plan is concerned with the software itself. If one wishes to improve the software or make new display options and such, how will the application design allow this.

Firstly, I will provide a rough example of how the data will be packaged and how will this make it possible to generate new displays without needing to change the application.

```
{
  "tsi": {},
  "tsv": {
    "voltage": 24.0,
    "current": 3.2,
    "state_of_charge": 0.93,
    "enabled": false,
    "pack_1": {
      "voltage": 7.0,
      "current": 0.8,
      "state_of_charge": 0.87,
      "enabled": false,
      "cell_1_1": {
        "voltage": 1.0,
        "current": 0.1,
        "state_of_charge": 0.93,
        "enabled": false,
      },
      "cell_1_2": {},
      "cell_1_3": {},
      "cell_1_4": {},
      "cell_1_5": {},
      "cell_1_6": {},
      "cell_1_7": {}
    "pack_2": {},
    "pack_3": {},
    "pack_4": {}
  },
  "glv": {},
  "dyno": {},
  "physics": {
    "cruise_control": false
  },
  "cooling": {
    "temp": 1.0
  },
  "dashboard": {
    "speed": 82.1,
    "rpm": 5200
  }
}
```

The gson package on the left gives an idea of how the package will contain the information. As you can see, every new category will be done in a deeper header level.

The application will generate expanding tabs according to the depth level of the information in the package. For example, the information for “cell-1-1” will be stored under, “tsi → pack-1 → cell-1-1 → voltage” in the software display. This will automatically be generated every time the software is run, according to the data available on the webserver created by the SCADA team.

Therefore, if a future teams decides to add a “pack-5”, this will automatically be available in the application. Such data can be used to generate custom charts or other forms of display.

More information about how this will be made possible will be available in the design proposal, especially considering the parts, “data structures” and “maintainability”.

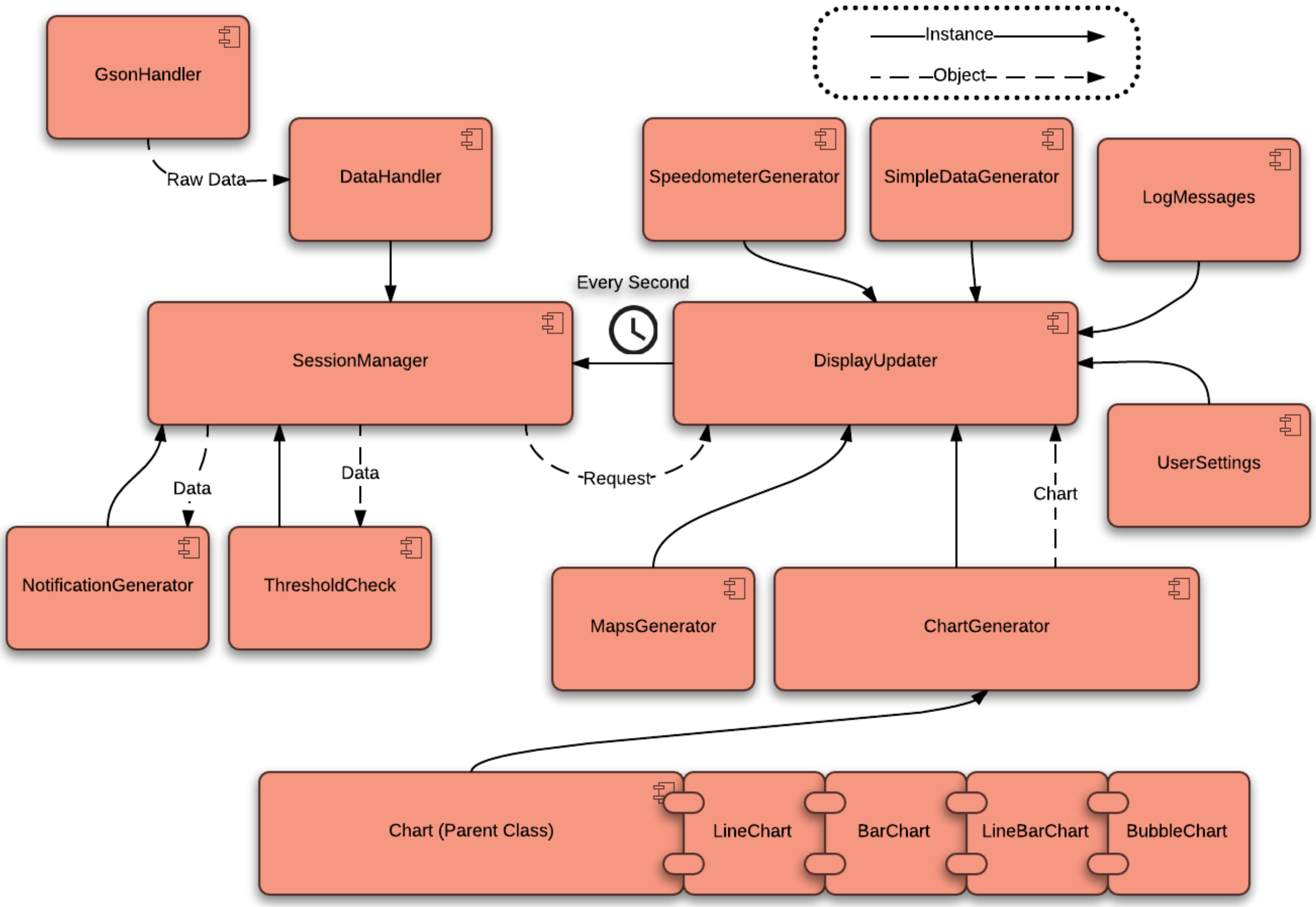
Another important aspect of this is that if a future team wants to implement a different type of database system, they will have to configure it so that it will pass through the application as a certain kind of object. This will make it easier for the maintainability of the system.

All this information will be documented with Javadoc as well as a tutorial prepared for the users and also a guide for the future teams about maintaining the application.

Secondly, another exciting feature of the application will be the easy possibility of expansion in the code of the software. Expansion in the software will be available thanks to the system design. Below is the content view from the design proposal. As you can see from the diagram, different View generators are sent to the SessionManager through the DisplayUpdater. This is a key part of the design as it will allow the future teams to easily add new Views for the system. The new Views can be easily added and passed through the Display Updater for the system.

Another expandability will be possible through the SessionManager. Currently, both software teams are planning to use Gson for data transfer to the android application. If in future, another database is wished to be implemented, DataHandler will make it possible to pass any sort of data to the SessionManager. It is a medium between the preferred data transfer and the system.

Additionally, the NotificationGenerator and ThresholdCheck provides easy ways to add new notifications or threshold checks for the system.



2.8 System Configuration Maintainability

The configuration will be specified in documents about the project and will be stored in version control. The system will run on any Android above version 4.0.3 and if future teams wish to increase the lowest requirement for the system, no changes should be necessary for the existing code.

The software requires a functioning android system to be able to run. There is not possibility for some of the functionality when the hardware is not complete. For the hardware, we are testing the parts of the software both in emulators and also in a real android phone.

2.9 System Configuration Checking

System configuration is checked automatically through the android system and the application.

2.10 Tool Chain, Design Suite

What tool chain will be used? Is the tool suite up-to-date and actively supported? Is the tool suite mature enough to have stable functionality? How is the tool chain installed in a new development system.

Android Studio will be used for building the software and this is the main IDE provided by Google for android development. Therefore, the software has been up to date and should be up to date as long as android operation systems are around.

The application will be easily installed through an apk file and the stability of the application will be checked through instrumented testing that is available in the android studio.

2.11 Third Party Software

The only third party software we are currently planning to use is MpAndroidCharts. This is a third party library that is constantly maintained and all version are always available through Mavin and version control. If a future team wants to upgrade the system, then this may require some change in the code. Parts where MPAndroidCharts are used will be specifically explained in the final proposal and in the javadoc as well.

2.12 Requirements of GPR007

2.13 All software source code must be maintained under configuration control. Release snapshots must be archived on the project website.

We are using version control for the application and thus closely follow who is making changed in the software.

2.14 The system must start from cold power-up and boot to full operational status without requiring user interaction beyond enabling power and safety procedures

The application will start with user prompt and work until it is exitted(destroyed).

2.15 Any PC software must be packaged for installation with a SETUP.EXE, RPM, “make install” or equivalent installer allowing it to be installed easily on any compatible computer.

The software will be in APK format and for android environment.

2.16 Configuration parameters, calibration factors, preferences, and options shall not be hardcoded within the software source code. It shall be possible to alter these various factors without recompiling software or physically disassembling hardware. Altered configuration parameters must be persistent through power cycling and reboots. The system must have a function to initialize itself with sane (factory default) configuration content if requested.

Please see the sections, 2.6 and 2.7 in this document.

2.17 All data and configuration files must be in a generally supported format (e.g. XML) or the format required by a mature and well supported application (e.g. MySQL database files, Berkeley db, etc…). Files shall be accessible either through removable media or network file transfer or both. How are requirements in GPR007 met?

Code will be available in github, documentation will be javadoc, database system will be gson from the webserver created by SCADA team.

DOCUMENT REVISION HISTORY			
Version Numbe r	**Approved Date**	**Description of Change(s)**	**Created/ Modified By**
1	2/10/17	Document Created	Marty Townley
2	2/18/17	Document Updated	Kemal Dilsiz