Maintainability Report

ECE 492 - Spring 2016

Abstract

This document outlines the language and software choices of VSCADA and the plans of maintaining the software.

Table of Contents

Summary

This document outlines how the VSCADA team plans to address the general maintainability requirements. This maintenance plan is software-specific. The software must be maintainable over the 5-year life of the LFEV system. The plan answers the following questions listed in the Statement of Work (SOW).

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

The software communicates with the web user interface over websockets. Messages about the system state are transferred between client and server. The web application is served by the SCADA daemon and any local file can be served, allowing the user interface to be changed without modifying any of the daemon's code. The daemon learns about the system state over CAN and the addresses of each subsystem datum can be easily changed.

2. 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?

The system topology is specified in a YAML configuration file. The file maps to a hash table and supports lists, numbers, booleans, strings, and comments, which make the configuration file self documenting. There will be no auto detecting hardware because then there would be no way of knowing if the system is completely online. Misconfiguration means forgotten subsystems are not being supervised or controlled, and will have to make do without input.

3. 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? Evidence must be provided to support assertions.

Ubuntu provides long term support versions of their software, they also have an active community and forums.

Systemd will be used as a system management platform for daemon creation as it has most major linux distributions support.[1]

The GCC compiler system created by the GNU Project, will be used as it supports programming languages such as C++ and Java, and the GNU Project regularly updates and maintains the compiler.[2]

4. What third party software will be incorporated into the system? How will this be maintained, upgraded, or patched during the life of the system.

SocketCAN, an open source CAN drivers along with a network stack was developed by Volkswagen Research. SocketCAN is included in the Linux Kernel 2.6.25 and up and is well documented on the linux kernel website.[3]

[1] http://www.freedesktop.org/wiki/Software/systemd/

[2] https://gcc.gnu.org/releases.html

[3] https://www.kernel.org/doc/Documentation/networking/can.txt

The web interface will be written using React, which is a view rendering framework developed by facebook. Instagram already uses it and it has a large user base. Check it out here. https://facebook.github.io/react/

We are using Python because the software is already written in Python. We will be using the asyncio library introduced in Python 3.4 and the native async syntax introduced in 3.5. This library has support from Python's BDFL and will likely continue to be around for awhile.

The software will be kept under version control using git and instructions for installing and configuring it will be kept in a README or wiki associated the software. All of this will live at git.lafayette.edu. The repository can be zipped and uploaded to the website. Tags can also be used to mark the progress of individual class years.

5. How are requirements in GPR007 met?

Software Maintainability

Software must be developed according to an explicit Software Maintainability Plan.

That's what this is.

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

git.lafayette.edu/groups/ece

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

systemd

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.

1. git clone
2. Read README.md
3. follow the instructions (they will be short)

Configuration parameters, calibration factors, preferences, and options shall not be hard-coded 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.

Calibration functions will be stored in a configuration file

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...). The use of custom formatted ASCII or binary files for configuration or data storage is not permitted. Files shall be accessible either through removable media or network file transfer or both.

YAML is a thing people use see http://yaml.org/.
TinyDB is really simple. Check it out here https://github.com/msiemens/tinydb
And it has 3578 downloads in the last month. https://pypi.python.org/pypi/tinydb

You can get at the logs with scp.

The use of removable media (thumb drives, flash media cards) is permitted for configuration parameters, offline storage, access, and backup. If removable storage is used for configuration, the system must have the inherent capability to operate without media, and to initialize blank media with sane configuration content.

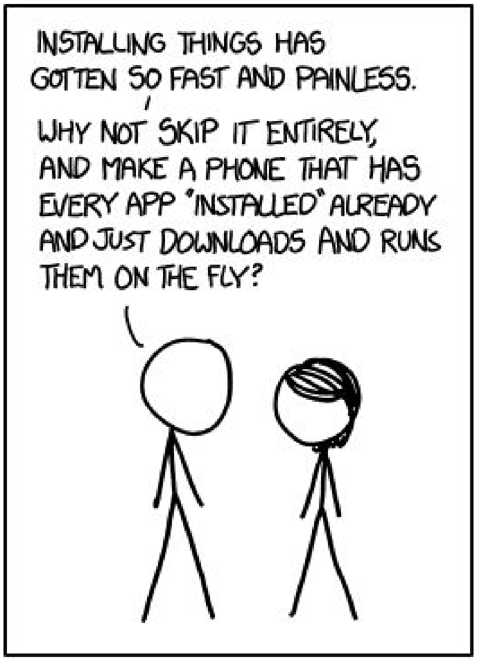
N/A

Enumerated devices, such as USB, must be automatically discovered by the system and assigned correct port designations such that the system operates correctly after re-enumeration without any interaction or re-programming by the user. Port designations may not be “hard coded” in the software or firmware.

Just put the USB device id in the configuration file.

```
/dev/usb/by-id/this-should-not-change
```

Any cell phone software must be packaged and available from an online “App store” for easy installation on any compatible phone without requiring special alterations of the phone such as SDK installation or jailbreaking.


INSTALLING THINGS HAS GOTTEN SO FAST AND PAINLESS.
WHY NOT SKIP IT ENTIRELY, AND MAKE A PHONE THAT HAS EVERY APP "INSTALLED" ALREADY AND JUST DOWNLOADS AND RUNS THEM ON THE FLY?

I FELT PRETTY CLEVER UNTIL I REALIZED I'D INVENTED WEBPAGES. [4]

Hardware Maintainability

Maintainability requirements for hardware must be demonstrated in the ATP both by analysis and by Inspection. The use of MIL-HDBK-472 (N1) and MIL-STD-470B, ISO/IEC 25000:2005, or other equivalent techniques are encouraged for the analysis.

This will be achieved as suggested in the Requirement and Test matrix.

In the hardware maintainability analysis you should assume a stock of recommended spare parts. The list of these spare parts should be included in the ATP. The Users Manual should include a section giving simple troubleshooting procedures. The Maintenance Manual should have more elaborate diagnosis and troubleshooting resources.

The analysis of spare parts will be included in the final version of the ATP after extensive research on different embedded and communication systems. The budget is included in the PDR. The Troubleshooting and Maintenance Manual will be included in the User Manual.

[4] source: https://xkcd.com/1367/

Should there be a failure, the system wide Mean Time To Repair (MTTR) must be less than 1 week over the system lifetime. MTTR applies to both hardware and software.

Thel testing methods and the time required will be analyzed in the CDR.

In addition, a maintainability inspection shall be conducted during ATP where a novice using procedures included in the User Manual demonstrates the diagnosis and repair of a likely failure, and an expert using resources included in the Maintenance Manual demonstrates the diagnosis and repair of an UN-likely failure.

This will be further developed in the User Manual.