Automated Safety Warning Controller

Final Report

by

Daniell Richter
Research Associate

Kelvin Bateman
Research Associate

Douglas Galarus
Program Manager
Systems Engineering and Development Integration

Western Transportation Institute
College of Engineering
Montana State University

A report prepared for the

State of California, Department of Transportation
Division of Research and Innovation

December 21, 2009

DISCLAIMER

ACKNOWLEDGEMENTS

The authors would like to thank Sean Campbell, Chief, ITS Special Project Branch, California Department of Transportation, Division of Research and Innovation, Ian Turnbull, Chief, Office of ITS Engineering and Support, Caltrans District 2, Ken Beals, ITS Engineer, and the staff of Caltrans District 2 for their support and assistance on this project.

TABLE OF CONTENTS

LIST OF FIGURES

EXECUTIVE SUMMARY

The California Department of Transportation (Caltrans) has contracted with the Western Transportation Institute (WTI) at Montana State University (MSU) to develop an "Automated Safety Warning Controller." The controller will interface with roadside devices such as sensors and signs. The controller will allow for automated data collection and application of best practice algorithms to analyze sensor data and to actuate related warning messages to motorists. For instance, wind warning messages might be actuated on a changeable message sign (CMS) when wind speed, as read from a sensor, exceeds a given threshold.

The purpose of this document is to present a summary of Phase 1 of this project.

At the conclusion of Phase 1, all project deliverables will have been delivered, and the project will have achieved its goals. A second phase is anticipated to start in early 2010, to carry forward with the work conducted in Phase 1, and prepare the system for eventual production deployment.

1. INTRODUCTION

The purpose of this document is to present a summary of Phase 1 of the project to develop an Automated Safety Warning Controller.

1.1. Project Goals

The primary goal of this project was to develop an Automated Safety Warning System Controller that will interface with roadside devices such as sensors and signs. The controller will allow for automated data collection and application of best practice algorithms to analyze sensor data and to actuate related warning messages and signals. For instance, wind warning messages might be actuated on a changeable message sign (CMS) when wind speed, as read from a sensor, exceeds a given threshold. The controller system has been designed for flexibility and extensibility, allowing for the integration and control of a variety of roadside devices. As such, it could be used as a standardized component with widespread applicability.

The end product of this project is a prototype hardware and software system that has been tested in the field.

1.2. Project Tasks

The work plan for Automated Safety Warning Controller consisted of the following eight tasks:

- Task 1: Project Management
- Task 2: System Concept
- Task 3: System Requirements
- Task 4: Testing and Development Lab Setup
- Task 5: System Design
- Task 6: System Development
- Task 7: System Testing
- Task 8: System Evaluation

1.3. Report Organization

This report presents a summary of activities that were and were not completed during the Automated Safety Warning Controller effort. As Project Management activities encompassed work related to budget maintenance, communications with the project sponsor, scheduling and the like, a discussion of the work completed for that specific task has been excluded. Remaining chapters of this report will summarize each of the remaining tasks listed above. A more comprehensive discussion of each task can be found in the deliverable document(s) associated with the task.

1.4. System Concept

Current practice for setting and activating CMS, EMS, and flashing beacon signs involves manual operation from the traffic management center (TMC). Data from field elements such as RWIS and loop detectors is retrieved by a TMC operator, who decides whether the data from the field elements warrants activation of a CMS, EMS, or flashing beacon. The speed of this process is subject to the efficiency of the TMC operator at retrieving data from numerous locations and deciding where, if at all, to activate a warning or information system. Further impacting this process is the fact that a TMC is generally not manned 24 hours per day, so there may be "off hours" in which there is further delay in activating warnings or information systems. Furthermore, communication lines might not be reliable, particularly during a severe weather event, making communication with field elements unreliable.

The goal of this project was the development of a system that frequently and automatically monitors field element data and determines, according to best practice algorithms, if a warning should be activated. The automated warning controller device would be installed near the warning devices it controls, so the system could continue to work even if communication with the TMC is disrupted. The controller can also poll field elements much more frequently than a TMC operator and quickly and automatically make the decision whether to activate a warning based on the retrieved information. Another goal of this project was to develop a standardized system that is versatile enough to be used at any location, with any number and type of field elements from which to draw data. Figure 1 shows an early conceptual diagram of the system.

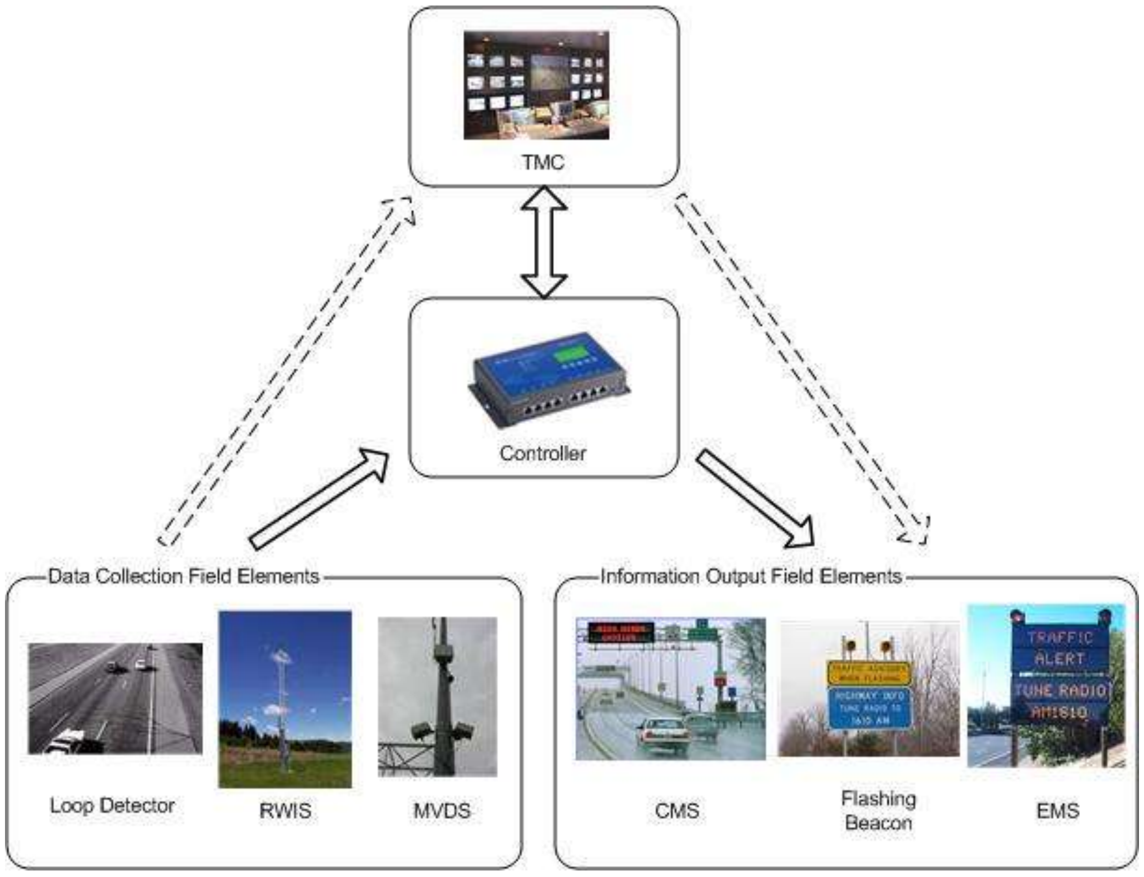

Figure 1: Controller Interaction with TMC and Field Elements

2. SYSTEM REQUIREMENTS

A system concept and requirements specification was produced and approved by the Caltrans project manager. The concept portion of the document outlined the overall goals of the controller. Discussed in this document were the advantages and disadvantages of different platforms, primarily embedded systems running Linux. Also, this document started to list key features and general capabilities of the Automated Safety Warning Controller. This served to validate the concept of the controller from a high level perspective and cleared the way toward the more specific system requirements.

The system requirements specify the basic operation of the Automated Safety Warning Controller. Specified in the document are:

- Field elements that should be supported,
- Management of specific field elements that a particular system shall communicate with,
- The capabilities of alert logic scripts,
- Methods and scope of administration and monitoring of the system.

The System Concept and Requirements Specification was completed relatively early in the project, and served as the principal guiding document throughout the remainder of Phase 1. A revised concept, shown in Figure 2, and the associated data flow, shown in Figure 3, were key architectural decisions made at this point. The separation of the data layer, the principal storage mechanism for the controller, from alert logic and field element modules allowed for a truly modular design that could be developed, lab-tested and field-tested in the same modular fashion.

This architecture was viewed as important enough to be addressed prior to a formal design phase and considered as a core requirement for subsequent development.

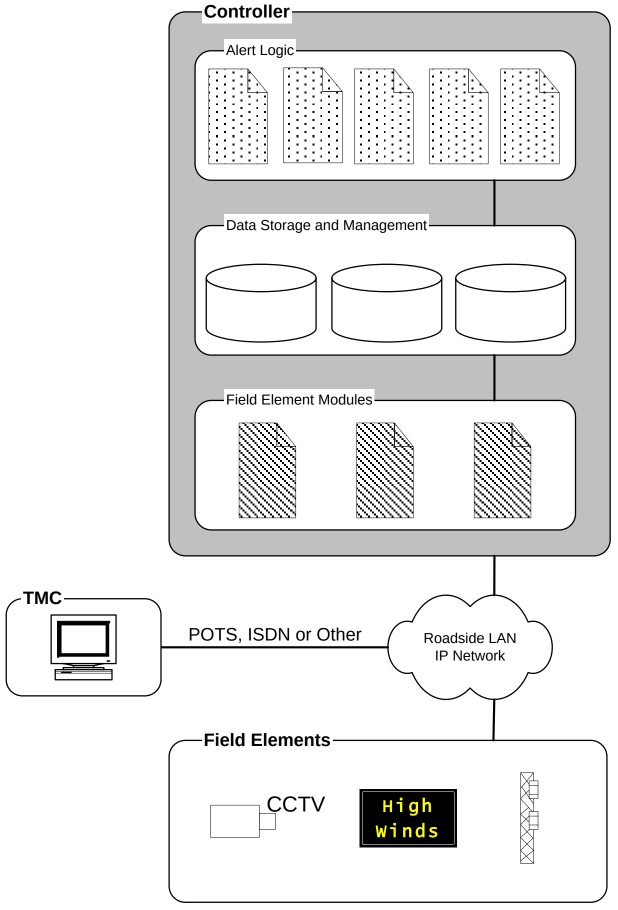

Figure 2: Revised System Concept

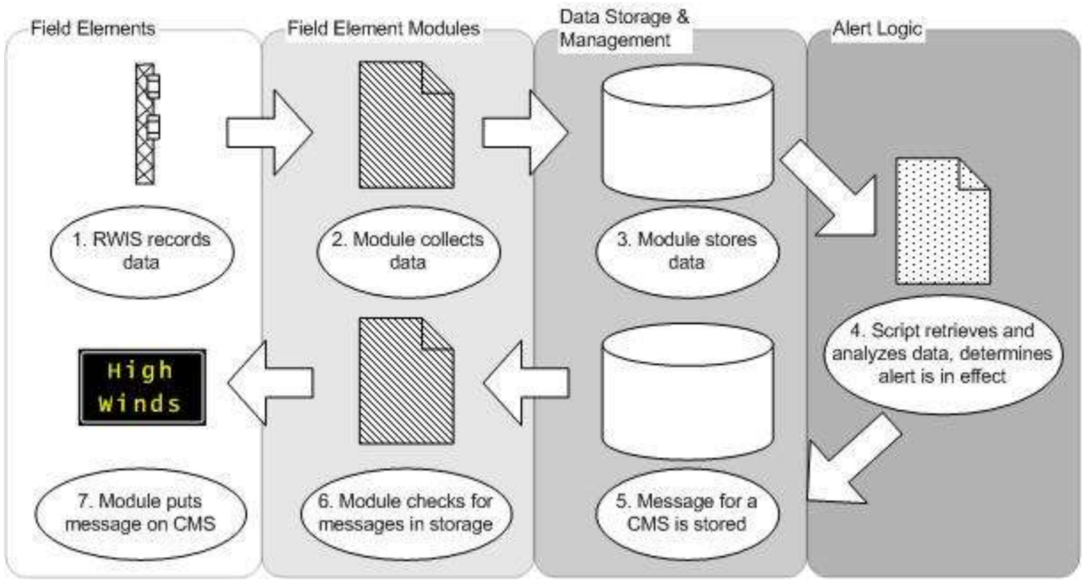
Field Elements
Field Element Modules
Data Storage & Management
Alert Logic
1. RWIS records data
2. Module collects data
3. Module stores data
4. Script retrieves and analyzes data, determines alert is in effect
5. Message for a CMS is stored
6. Module checks for messages in storage
7. Module puts message on CMS
High
Winds

Figure 3: Data Flow in Revised Concept

3. TESTING AND DEVELOPMENT LAB SETUP

For the purposes of developing and testing the controller system it was necessary to simulate the devices that the controller would be communicating with. The test and development lab was designed, largely by Caltrans, for that purpose. It simulates a roadside network similar to what a controller system would be connected to in the field. The test lab was necessary to verify that the controller communicated properly with those various field elements.

The project champion, Ian Turnbull, built the test lab components at Caltrans and delivered the system to WTI. He worked with WTI staff to install the equipment in the WTI Systems Lab, where it continues to operate in support of this project. These components are shown in Figure 4.

Most of the hardware installed in the test lab is identical to what is installed in the field, so if it was verified that the controller could send and receive information with the field elements in the lab, it would also be able to send and receive information with the field elements in the field. The one exception to this in the lab is the RWIS elements. The lab implementation of the RWIS is NTCIP Exerciser software running on a computer connected to the field elements test network. The software emulates the communication protocol used by a real RWIS for testing and implementation; however, verification against a real RWIS was still necessary. The RWIS at the Fredonyer Pass location was used for this purpose, after being configured to allow dialing in from a remote location so that the controller could connect with it.

A testing and development lab setup summary document outlined the details of the lab. This document was submitted as a deliverable earlier in this phase.

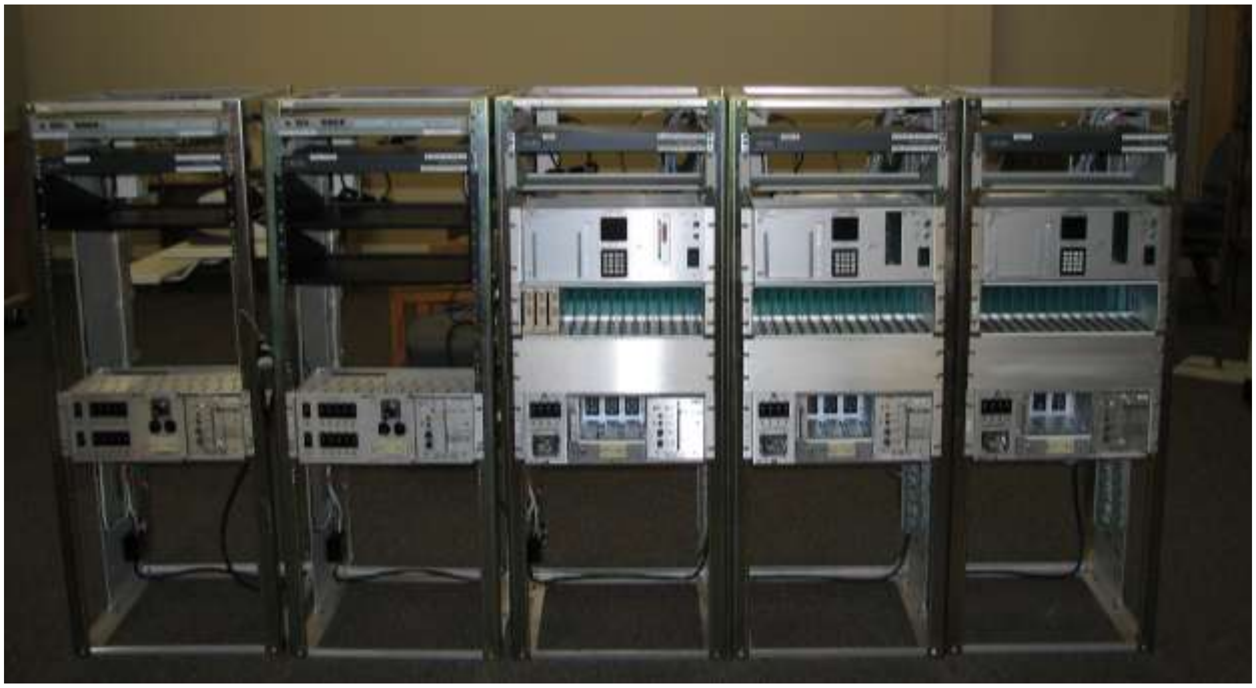

Figure 4: Equipment delivered by Ian Turnbull

4. SYSTEM DESIGN

The selected development language for the controller software was Python, chosen for its support of a highly modular system as desired for the project. This modularity supported the loosely coupled architecture, easing development and making a more flexible system. Modules are completely independent of each other, relying on the data layer to communicate with each other.

The software architecture centers around a small core unit that initializes and activates the other modules, and then sits in the background and waits for the modules to finish. The field element modules each communicate with their respective type of field element, storing and retrieving data on disk through the data management layer. The alert logic modules retrieve data from the data management layer and, based on the values in the data and the logic in the alert script, may write a command back to the data management layer for the CMS, EMS, or flashing beacon field element to perform.

Two additional modules provide the user interface to the system. A command line interface is provided to users that can connect to the device via a network or serial connection, which provides summaries of module status and controls to change the operation of the system. Another module drives the front panel interface through an LCD and five buttons. This interface only provides summary information; no values can be changed from the front panel.

A high-level design document describing this process in detail was submitted by the project team early in the project. This design was considered nearly complete at that point, but finalization did not occur until near the end of the current phase. Several minor modifications were made to the original document to incorporate new information and to correct several minor discrepancies. This design was a natural extension of the revised concept shown in Figure 2 and Figure 3.

A key design decision, made by the project team and Caltrans, was the use of a Moxa UC–7420 embedded computer as the platform for the controller. The software platform was a Moxa-supplied variation of the MontaVista Linux distribution based on the 2.4.18 kernel and Python version 2.5. This device has proven suitable for development and field testing on this project, and certainly could be considered for subsequent deployment. In general, this is a typical Linux-embedded device, so alternatives could be considered. Figure 5 shows the Moxa UC–7420.

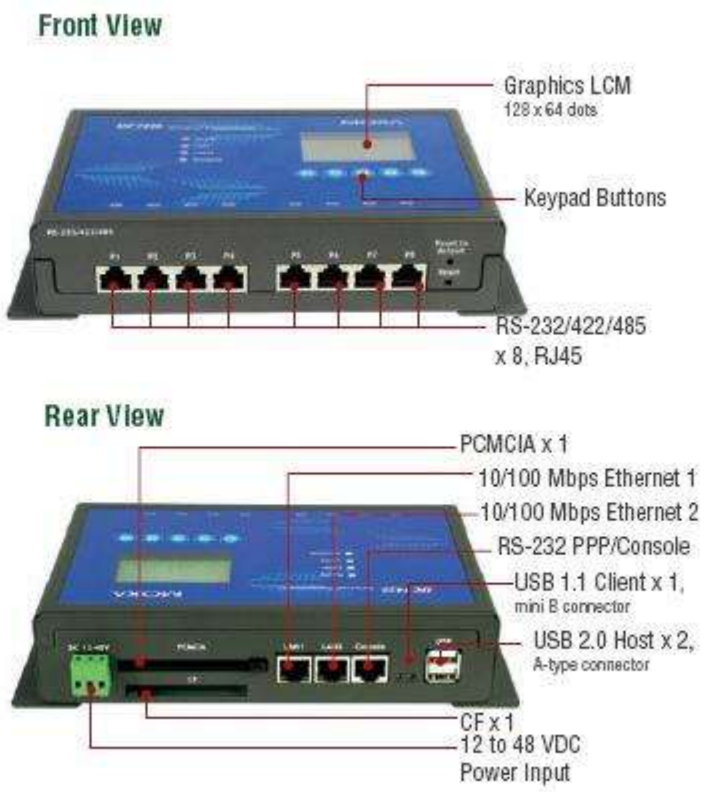

Figure 5: Moxa UC-7420 Embedded Computer (image courtesy of Moxa)

5. SYSTEM DEVELOPMENT

The selected hardware for the controller system used an Intel XScale processor, which is based on an ARM architecture. However, most development was done on standard desktop systems that use an x86 architecture. The use of Python as a development language simplified any portability problems that may have occurred due to the difference in architectures; as a scripting language, Python code will run unmodified on any platform with the Python interpreter installed.

Development started with field element modules as standalone programs—e.g., the RWIS module was initially developed as a program that would retrieve sensor values from an RWIS and store them in a text file. The individual programs were later modified to work as modules that could be integrated with the controller software. Currently supported field element modules include RWIS, CMS, and loop detectors.

Alert scripts were intended to be simple enough for non-programmers to write. It was decided that, in order to avoid the complexity of developing the equivalent of a new scripting language, alert scripts would be written in Python. Keeping alert scripts simple to write while using a full-featured programming language like Python required a support structure that would wrap around each alert script and take care of the details that add complexity to the alert scripts. There was a lot of discussion between WTI and Caltrans about the amount of complexity that should be taken out of the alert scripts and what level of expertise would be required to author an alert script. The final alert script system provided all the required sensor values by field element name and sensor name. Most alert scripts will look very much like the template that is provided with each system: a decision statement comparing one or more sensor values to threshold limits, followed by an action to perform (such as placing a message on a CMS) should the decision statement be true. Actions involve some setup (defining the content of the CMS message, how long the message should be on the sign, and the relative priority of the message, etc.), followed by a statement to perform the action (such as PutMessageOnSign). More complex alert scripts are certainly possible.

The controller system stores all data collected from or sent to field elements on the local file system. There is a common data module that provides an interface to all the other modules and the alert scripts for data storage. Data storage was abstracted out to a common module to make it easier to use different data storage methods in the future. Currently all data is stored in plain text files using the comma separated values (CSV) format. A series of performance tests between the current format and a database platform showed no advantage to using a database; the current file format is just as fast or faster while being simpler to implement and readable.

Management of the controller system is done through a command line interface. As requested by the project champion, the interface is modeled after Cisco’s IOS interface. To TMC operators the interface provides summaries of controller internals, such as field element sensor values and status summaries of installed modules. After elevating privilege to supervisor via an IOS-inspired enable command, modules can be paused or stopped, and threshold values can be changed.

The field elements that are fully supported at the conclusion of this phase are:

- RWIS
- Loop Detector (Model 222 GP5 in a DTS 170e Controller)
- CMS (SignView 170, Model 500)

Field element modules in partial development, but not finalized include:

- MVDS
- EMS
- Flashing Beacon

In addition there has been no development on SOCCS Automated Controller or on a field element module for the video detection system. These incomplete features will be developed in the next phase, after a successful pilot test proving the viability of the system design and architecture. A subsequent phase will also involve further refinement of the complete modules and of the system as a whole.

There was development on a browser interface that provided most of the functionality that would be offered by a SOCCS Automated Controller system. The interface was developed primarily as a proof of concept to show the capabilities a user interface (such as the future SOCCS interface) may have. In the next phase this interface may evolve into the SOCCS interface or at least provide a starting point.

6. SYSTEM TESTING

Testing of the controller system involved unit tests of each module, integration tests of the controller system as a whole, and long-term reliability tests. The controller system is based on various modules that were originally developed as standalone applications that could be tested individually and in isolation.

Field element modules were all thoroughly tested in the original standalone format, verifying proper operation when communicating with a field element. Input field element modules such as RWIS module simply retrieved information from the field element and stored it to a text file. The test lab that simulated all the field elements made it easy to manually set each sensor value and verify that the value stored by the field element module matched. In the case of RWIS, the module was tested against the simulated system in lab, and then tested against a real RWIS system at Fredonyer Pass in District 2. Output field elements such as CMS were tested to verify that data was properly sent to the field element. For the CMS module a message and all its parameters could be stored in a text file, sent to the field element, and verified using the SOCCS CMS software.

Instead of testing the individual operation of each module, the integration tests were used to test interoperation of each module with all the others. Field elements and alert scripts had to share data, and the manager module needed to exert control over every module. Integration tests involved checking and verifying the interactions between modules, that accurate data was passed and that no errors occurred.

It also had to be verified that the system would work in the long term with minimal supervision. Much of the testing late in the project was simply letting the system run with a variety of sensor values. Values that fell within the alert threshold were simulated by the field elements, and the CMS controller was checked with SOCCS to verify that the expected message was placed. The system was occasionally supervised via the management interface to make sure each module was running as expected; logs were checked to verify that errors weren’t occurring or that intentionally simulated errors were properly handled. The system was also tested over several months at the Spring Garden pilot location in District 2 with an icy curve warning script, and it has successfully placed warning messages on the CMS there.

7. SYSTEM EVALUATION

Evaluation of the system's technical performance was performed via testing at the Spring Garden pilot location. Several data and log files that included a time period in which the weather should have triggered an alert were retrieved from the device and sent back to WTI for evaluation. Inspection of the files showed that the controller was working as it should; a CMS message was properly generated by the alert script and was placed on the sign without error.

Reliability was evaluated by unsupervised tests of at least several months in duration. This included both in-lab and in-field tests. The device has been running continuously at the pilot location for close to four months at the time of this writing and continues to work properly and without error. None of the in-lab systems have been tested continuously for as long, but they also continue to operate reliably and without error.

Usability evaluation was accomplished through a list of survey questions sent to Ken Beals of Caltrans. No TMC operator interacted with the pilot device, and there is not currently a SOCCS interface, so the usability survey focused on setup and installation. Considering the autonomous nature of the controller software, ease of setup and installation are probably the main areas of usability focus, and the responses to the usability survey were predominantly positive.

The system is intended to be largely autonomous, so maintainability only becomes an issue in the very long term. The evaluation did not cover a long period of time, so the maintainability evaluation is based on projected data. The hardware used for the controller is suitably hardened for the anticipated environment; hardware failure should be very infrequent and, should it occur, will require servicing by Moxa or outright replacement. Normal operation of the controller system and field elements will result in the storage of a large amount of information, but the 4GB card used should hold at least a year's worth of data. However, the maintainability evaluation recommends system inspection and data backup at least quarterly.

Evaluation of the controller system's security consisted of measuring the size of the device's attack surface, or the number of network ports accepting connections. By default the Linux installation that Moxa packages with the device has several ports open, but the security evaluation recommends which ones should be closed to minimize the attack surface. The system also uses time-tested authentication methods built into Linux to minimize the possibility of a bug in the controller code creating a vulnerability. While not exhaustive, evaluation of the security of the system has been satisfactory.

8. INTELLECTUAL PROPERTY PLAN

An intellectual property plan was listed in the original scope of work as a deliverable for this project. While the handling of intellectual property is a critical element of this project, a specific plan has not been created. This work is viewed as deferred until the next phase of the project, in which it can be addressed in a more complete and meaningful fashion. The general intent of the project team and the Caltrans project champion and project manager is that the system will be open-source. The use of Linux as an operating system and Python as the primary programming language are consistent with this intent. An anticipated complication is that certain vendor protocols may be proprietary, and not made available for public or otherwise open release. In such a case, provisions will need to be made for the handling of such protocols. Otherwise, the greatest remaining issues will be determining suitable repositories and mechanisms for system source code and general redistribution. At the time of this writing, the project team is confident that a suitable means can be found to accomplish the general intent in regard to handling of intellectual property.

9. CONCLUSION

A number of deliverables were produced during the course of the project effort, including:

- Final Report
- Quarterly Reports
- Project Plan
- Intellectual Property Plan
- Concept Specification and Validation
- Requirements Specification
- Testing and Development Lab Summary
- System Design Summary
- Logical Design Document
- System Development Summary
- System Testing Summary
- Testing Plan
- Evaluation Summary
- Evaluation Plan

Some of these documents were consolidated, resulting in the following:

- Automated Safety Warning Controller System Concept and Requirements Specification
- Automated Safety Warning System Controller Project Plan
- Automated Safety Warning Controller High Level Design Specification
- Automated Safety Warning Controller Testing and Development Lab Summary and System Development Summary
- Automated Safety Warning Controller System Testing Plan and Summary and System Evaluation Plan and Summary

Note that the Intellectual Property Plan is generally considered deferred until the next project phase, and is addressed with brief commentary within this document.

These deliverables, along with the developed system, considered to be at a stage somewhere between prototype and pilot, represent the work conducted within this phase of the project. In general, the effort has been successful to the point that a subsequent phase is merited and planned.