Automated Safety Warning Controller (ASWC) Phase I – Proof of Concept

Kelvin Bateman
Research Associate
and
Dan Richter
Research Associate

Western Transportation Institute
Montana State University

MONTANA STATE UNIVERSITY | College of ENGINEERING

Mountains & Minds

Agenda

- Introduction to Automated Safety Controller System
- General Requirements
- System Hardware and Software selection
- Testing Lab Setup
- System Design and Development
- System Administration and Configuration
- Field Testing setup and evaluation
- General lessons learned
- Next Steps

Original Problem Statement

Caltrans posed the following problem statement:

In order to provide better safety warning information to motorists, how can roadside condition sensor data be automatically analyzed and real-time road condition information be displayed to the traveling public?

[Caltrans Research Problem Statement]

Background

<u>Automated warning systems are not a new concept within the transportation community.</u> There are several projects on the state highway that use the concept of a roadway sensor initiating some type of motorist warning. To date, <u>all of these systems are unique implementations that use one-of-a-kind software for control</u>. The system controller is a custom device which can only be used with that particular project's physical and electrical layout. The department has benefited from a standardized approach to individual field elements such as CMS, EMS and detection loops. <u>A standardized automated warning system controller, which controls standardized field elements in a system environment, has not been developed to date</u>.

[Caltrans Research Problem Statement]

Statement of Urgency, Benefits, and Expected Return on Investment

<u>The demand for the deployment of automated warning systems will continue to increase</u> as a lower cost alternative to major highway realignment for safety improvements. If the department acts now to <u>develop a standard approach that considers maintenance, flexible implementation, reliability and communications issues</u>, a multitude of one-of-a-kind deployments can be avoided. <u>The benefits of standardization with regard to personnel training, equipment purchase and repair are well known within the department.</u> Development of this type of controller is totally consistent with the technical foundation defined in the Transportation Management System (TMS) Standardization Plan.

[Caltrans Research Problem Statement]

Deployment Potential

The controller could be deployed in a wide variety of situations to mitigate safety problems. Applications such as ice detection and warning, wind warning, queue detection and warning would all be readily deployable. Portable systems could also be deployed in workzones for queue detection and warning. Currently, the requirement of custom controller software development is a significant barrier to automated system deployments. The availability of an "off the shelf" controller specifically designed to interoperate with standard TMS elements would tend to promote its use both within California and throughout the nation.

[Caltrans Research Problem Statement]

Current Practice

Manual data retrieval and analysis at TMC

- Someone in the field notifies Dispatch of inclement conditions
- Dispatch notifies the TMC
- The TMC operator verifies conditions on RWIS
- The TMC operator manually sets the warning (CMS, EMS, Flashing beacon)
- The warning stays in place until the TMC operator is informed of a change or he notices it himself

Alternatively:

- The TMC operator is expecting inclement weather and watches the current conditions
- Puts the warning up when conditions get bad enough
- The warning stays in place until the TMC operator is informed of a change or he notices it himself

Current Practice

Problems:

- Manual process: requires people
- Latency: delay between conditions worsening and warning being put up
- Long connection distance: higher chance of failure
- Human error
- Off hours issues – TMC not manned 24/7
- Process often breaks down at the warning removal step

Current Practice

- One of a Kind Controllers
  - Purpose built for a single type of warning
  - A complete package (sensors, signs, and other hardware)

- Problems:
  - Only one type of warning
  - Requires installation of single-use hardware instead of re-using existing sensors, signs, etc.
  - Non-standard: hardware, interface may vary from site to site

Proposed Solution

WTI proposed to conduct research 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.

Automated Safety Warning Controller Phase I Objectives

- Develop an automated warning system controller that can be easily configured to acquire sensor data from existing, standard Roadside Weather Information Systems (RWIS), detection loops, Microwave Vehicle Detection Systems (MVDS) and video detection systems

- Controller must use best practice algorithms to analyze the sensor data and determine operating conditions (i.e., ice is present on pavement; a vehicle queue is present)

- Controller must be able to be easily configured to actuate a Changeable Message Sign (CMS), Extinguishable Message Sign (EMS) or flashing beacon

Automated Safety Warning Controller Phase I Objectives (cont)

- Controller must be able to be located at the roadside, in very remote locations, and meet all of the environmental, power and communications requirements necessary for reliable operation

- Controller must also be able to operate as a stand-alone field system or be able to internetwork with a Transportation Management Center (TMC) or other automated systems

Automated Safety Warning Controller
Phase I Concept

- General purpose controller
- Uses existing sensors
  - RWIS, Loop Detector, MVDS
- Uses existing displays
  - CMS, EMS, Flashing Beacon
- User configurable by Field Engineers
  - Flexibility to specify devices for individual site configuration
  - Users can write Alert Scripts to meet the needs of the site

Automated Safety Warning Controller Phase I Concept

- Advantages
  - Located at site, short network length
  - Frequent poll and evaluation of data
  - 24/7/365
  - Standard for all sites and all uses

Automated Safety Warning Controller Phase I Outcome

- A prototype device was produced
- Software was developed to:
  - Poll field element sensors
  - Make decisions based on sensor data
  - Place messages on or activate signs

An Automatic Process

- Field elements are already deployed
  - ASWC doesn't need it's own single-use sensors
  - Doesn't monopolize existing sensors

- Solutions
  - Automatic: 24/7/365
  - Data retrieval and decision making happens every few minutes
  - Device is co-located at the site with existing field elements

General Requirements

- Automated
  - 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
- Flexible
  - the controller system will be designed for flexibility and extensibility, allowing for the integration and control of a variety of field elements
- Extensible
  - additional field elements are simple to add and configure into the system

General Requirements (cont)

- Standardized
  - the controller will provide a single, open, multipurpose system, as opposed to many single purpose systems
- Self Contained
  - one piece, similar look and feel to a network router.
- Include the necessary hardware and software interfaces to communicate with, control and acquire data from field elements
- Include hardware and software interfaces for remote management

General Requirements (cont)

- Easy to setup for varied field site configurations
- User configurable/customizable alert scripts

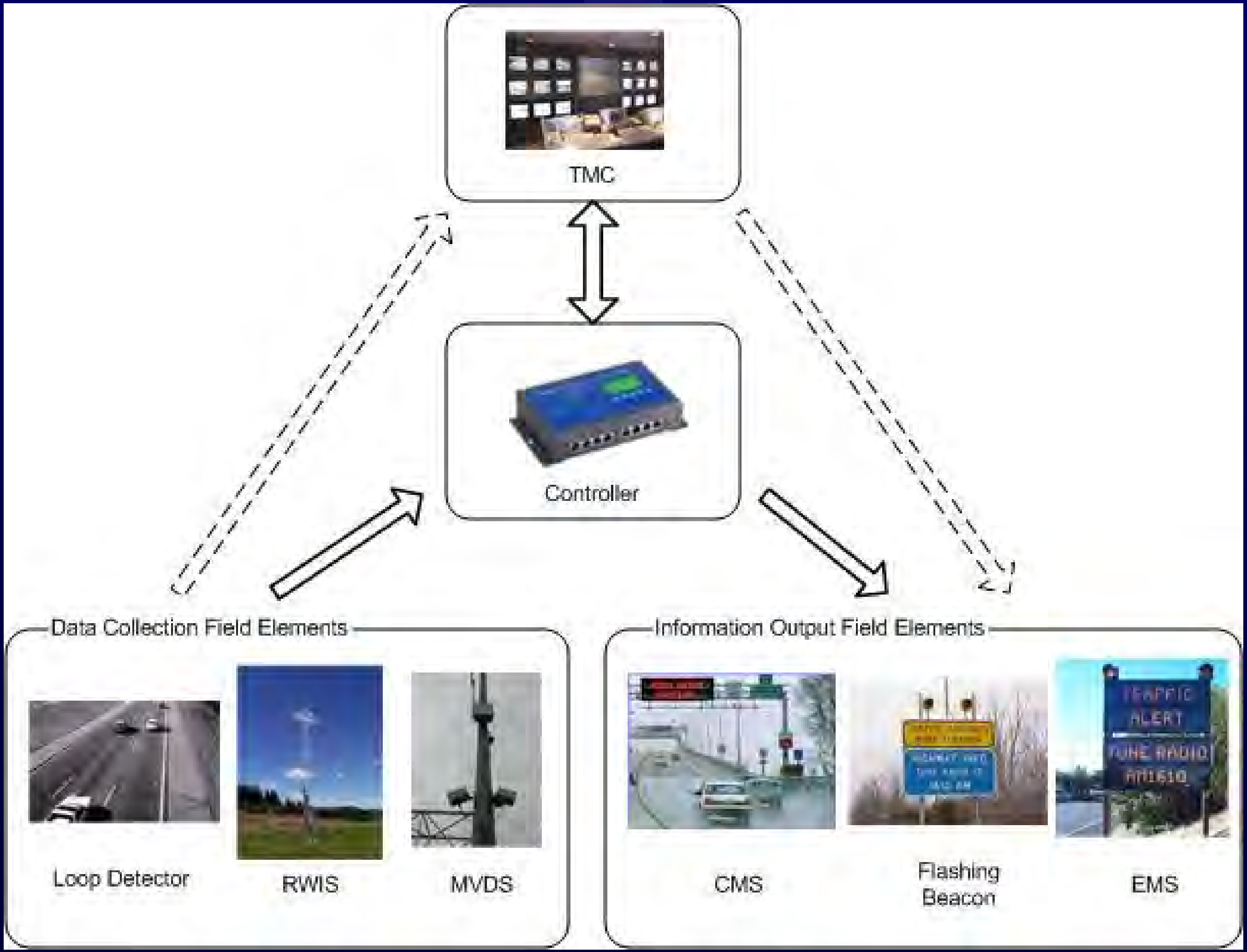
Field Elements

- Road Weather Information Systems (RWIS)
- Changeable Message Signs (CMS)
- Extinguishable Message Signs (EMS)
- Highway Advisory Radios (HAR)
- Loop Detectors
- Microwave Vehicle Detection Systems (MVDS)

Field Elements

- Protocols
  - – RWIS: SNMP with NTCIP defined OIDs
  - – CMS: proprietary byte stream
  - – EMS, Flashing Beacon: HTTP (WebRelay)
  - – Loop Detector: Proprietary byte stream

Basic Concept Summary



Hardware Options

MOXA UC-7420

- Intel XScale IXP-422/425, 266/533 MHz processor
- 128 MB RAM on-board, 32 MB flash
- 8 RS-232/422/485 serial ports
- Dual 10/100 Mbps Ethernet
- USB 2.0 host
- CompactFlash socket for storage expansion
- LCM display and keypad for HMI
- Ready-to-run Linux or WinCE 5.0 platform
- DIN-rail or wall-mount installation
- Robust, fanless design
- Environmental Limits:
  - **14 to 140°F**
  - **5 – 95% RH**
- ~$900

Moxa Photo

Hardware Options

Another option: DA-661-16-LX

- Intel XScale IXP-425, 533 MHz processor
- 128 MB RAM on-board, 32 MB flash
- 16 RS-232/422/485 serial ports
- Dual 10/100 Mbps Ethernet
- USB 2.0
- CompactFlash socket for storage expansion
- LCM display and keypad for HMI
- Ready-to-run Linux or WinCE 5.0 platform
- Rack-mount installation
- Robust, fanless design
- Environmental Limits:
  - **14 to 140°F**
  - **5 – 95% RH**
- ~$1350

Moxa Photo

Hardware Options

IPAD

- 1GHz Apple A4 custom-designed, high-performance, low-power system-on-a-chip
- Operating temperature: 32° to 95° F (0° to 35° C)
- Touchpad display
- Limited connectivity
- Limited Programming options
- Limited mounting options
- $500-$900
- Real Cool

Apple Photo

Other Hardware Options

- Sun Jbox
  - VIA C3 (x86) processor
  - Choice of Solaris, Java JDS3, or Linux 2.6.10 Operating System
  - Full capabilities of a desktop system
  - Not hardened
  - $400
- Single Board Computers
  - Some assembly required
- Techsol gateway
  - < 2 watts
  - 32 bit RISC processor
  - Linux 2.6
- Axiomtek eBox
  - Wide range of x86 processors (Core 2 Duo, AMD LX800, Atom, Pentium M)
  - Wide temperature range, hardened

None of the above have a built in LCD or input buttons

Hardware Selected

- Moxa UC-7420
  - Software: Moxa modified version of MontaVista
    - Linux Kernel 2.4.18
    - Busybox
    - Bash
    - Apache
  - Moxa supplied a cross-compilation toolchain to install on a desktop Linux distribution
  - Python had to be built by us and installed on the device

- Moxa DA-661-16-LX (rack mount) is next …

Operating System Options

- Embedded Linux
  - E.g. MontaVista
  - Pros:
    - Lightweight
    - Better for a single use embedded system
  - Cons:
    - Different deployment and development environments
- Mainstream Linux
  - E.g. SUSE, Debian, RedHat
  - Pros:
    - Full featured development environment
    - All the tools Linux users are used to
  - Cons:
    - Not as lightweight as embedded specific distributions
    - More tools and features than necessary
- Embedded Windows
  - E.g. Windows CE, Windows Embedded Standard
  - Pros:
    - More language options
  - Cons:
    - Cost of software and development tools
- Roll our own
  - Pros:
    - Exactly what we need. No more, no less.
    - Lower hardware requirements
    - Can be done subsequent to initial development, after requirements are better understood
  - Cons:
    - Very time and resource intensive

Programming Options

- C/C++
  - Pros:
    - Native binary
    - Speed and efficiency
  - Cons:
    - Cross-compiling
    - Less flexible after compilation
- Java
  - Pros:
    - Cross platform binary
    - Almost as fast as C/C++
  - Cons:
    - Wasn't available on Moxa early on.
    - Wasn't open source.
    - Requires compilation.
- Python
  - Pros:
    - Doesn't need to be compiled
    - Cross-platform script
    - Very flexible
    - Broad range of modules
  - Cons:
    - Mostly interpreted
    - Less speed and efficiency
- Perl
  - Pros:
    - Same as Python
  - Cons:
    - Interpreted
    - Scales poorly
    - Difficult to maintain
    - Doesn't handle OO well

Programming Selection

- Python
  - No need to develop an interpreter for alert logic scripts, they can be written in Python too
  - Natively object oriented
  - A lot of functionality supported in language
  - Clean and readable syntax
  - Native multithreading
  - Conducive to rapid development and testing

Testing Lab Setup

- Thanks to Caltrans and Ian Turnbull
- Simulate Caltrans field elements and communication via elements and TMC.
- Equipment includes:
  - Two CMS Controllers
  - Loop Detector
  - SNI Servers
  - Power Supplies
  - Modems and Routers
- NTCIP exerciser software to simulate RWIS

Testing Lab Setup

WTI Photo

Caltans Loaner Equipment

Communications equipment

Loop Detector (Model 222 GP5 in a DTS 170e Controller)

CMS (SignView 170, Model 500)

Testing Lab Setup

- WTI Systems Lab
  - Phone lines for dial up networking
  - Communication tower on roof
  - Private network separate from MSU/public network
    - Connectivity to Systems Staff Offices, Labs and Rooftop

Traffic Sensors

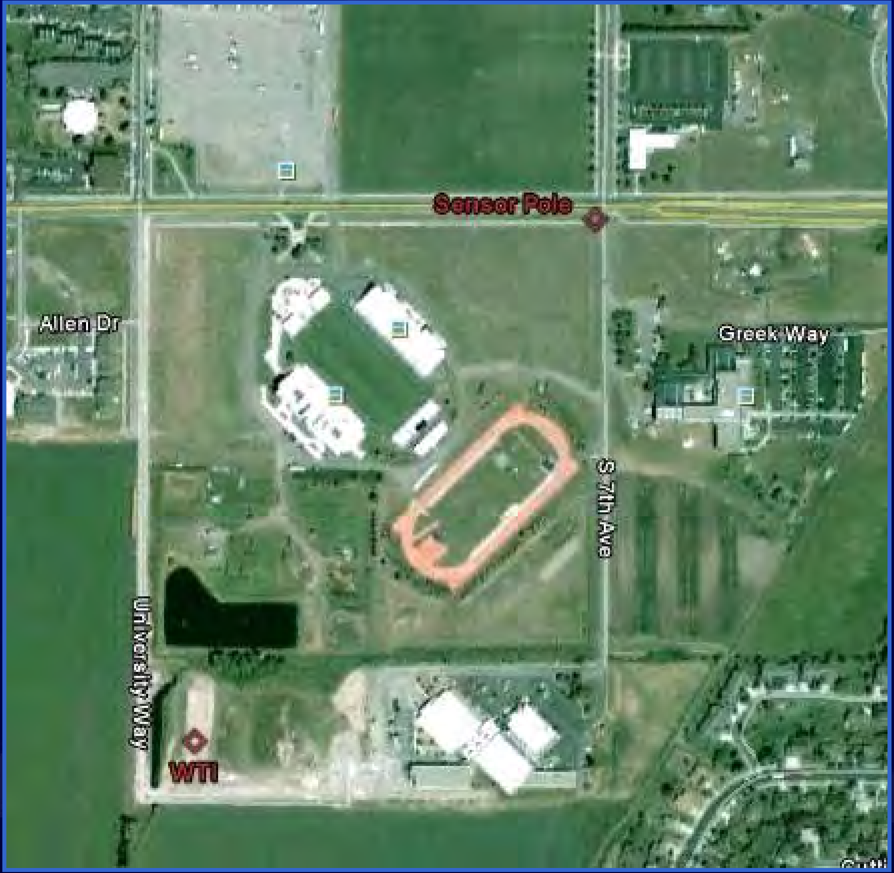
Location

- S. 7th and Kagy
- ≈ 0.4 mi NE of WTI

Communication

- 802.11g WiFi
- Mast on WTI roof

WTI Photo



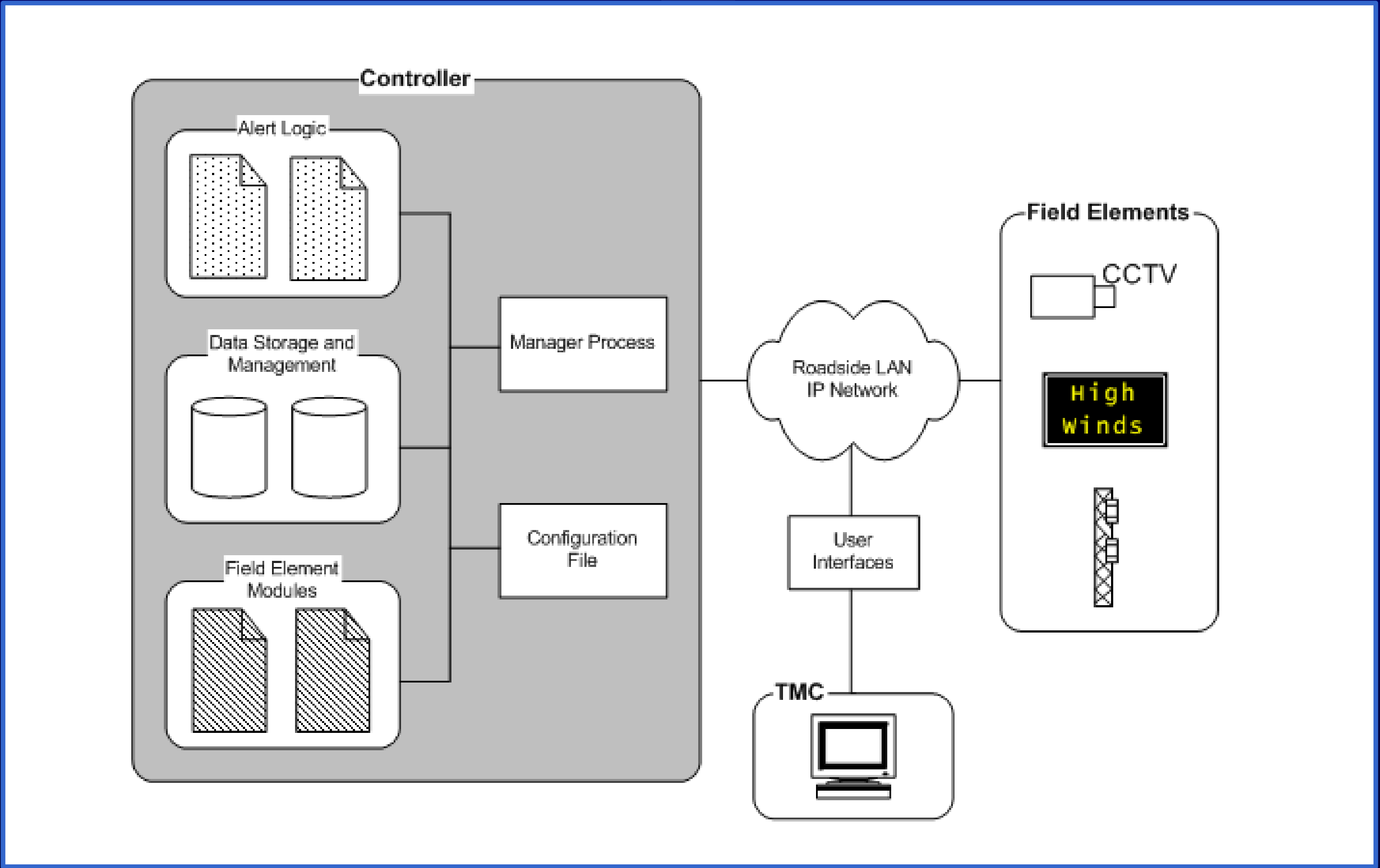
Map from Google Maps

WTI Photo

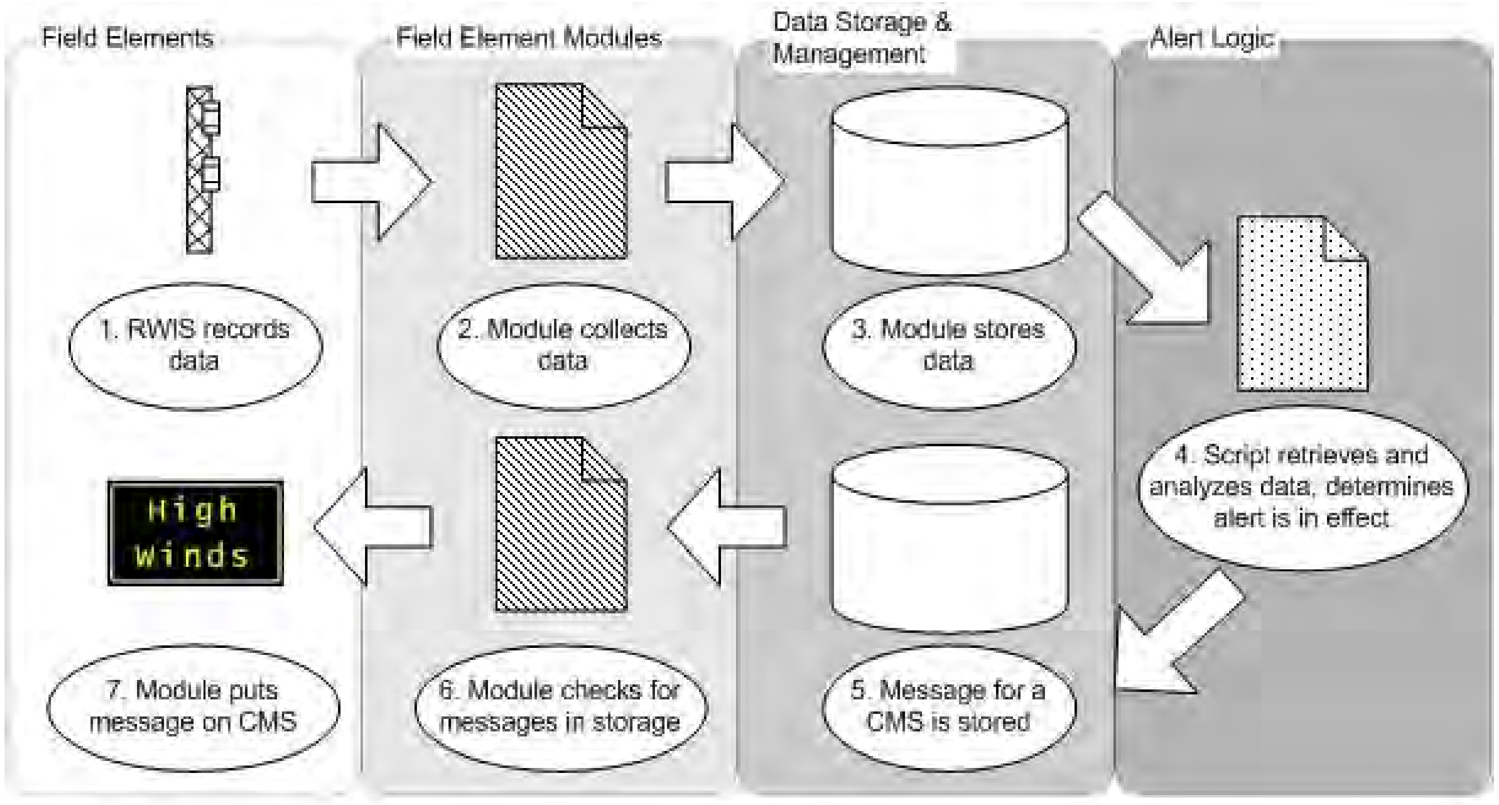
Equipment

- Wavetronix MVDS
- EIS X3 RTMS
- Cohu Camera

Proposed System Architecture



System Data Flow


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

Software Design Considerations

- Not real-time, modules configured to run at set intervals
- Separate module for each type of field element
- Modules read configuration to configure self for specific application
- Data stored in flat text files

Individual Module Design

- Data Module interface used by all modules to access data files
- Where applicable separate logic functions from communications functions
  - CMS Module is generalized, uses CMS_SV170Model500 for protocol specific functions
- Different types of modules
  - Manager modules
  - Field Element Input (RWIS, Loop Detector)
  - Field Element Output (CMS)
  - Logic modules

Manager Modules

- Thread monitor
  - One instance
  - Provides command line interface over telnet
  - Serves as interface for 2 of 4 different permissions groups

- Front panel interface
  - One instance
  - Provides management interface through front panel LCD and buttons

Field Element Input Modules

- RWIS
  - Gets data from RWIS using NTCIP over SNMP
  - Data gets parsed out and stored in a file
- Loop Detector
  - Protocol is a byte stream over TCP/IP
  - Data gets parsed out and stored in a file
- MVDS
  - Device specific protocol is a byte stream over TCP/IP
  - Data gets parsed out and stored in a file

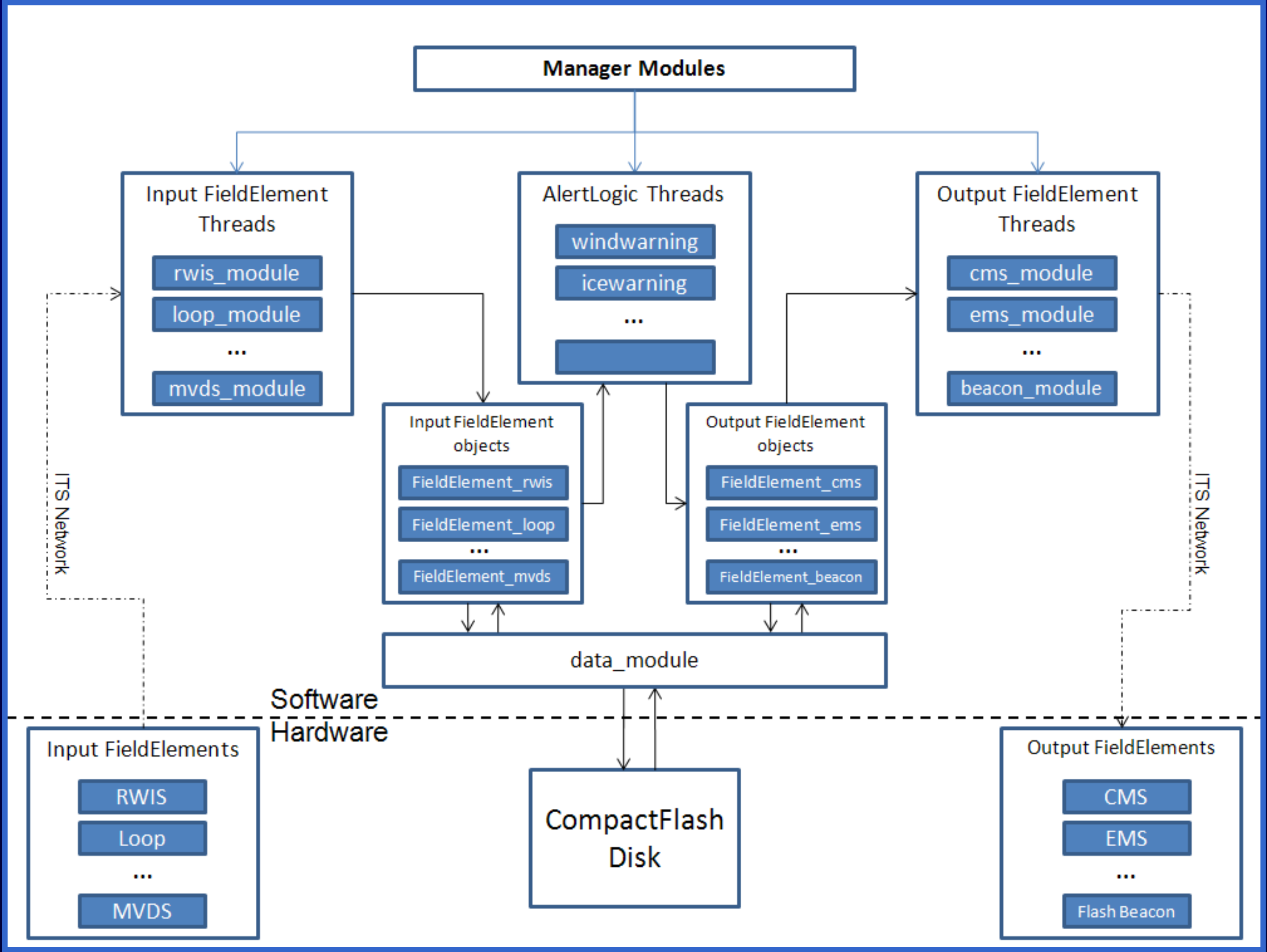
Field Element Output Modules

- CMS
  - Reads message queue from a file
  - Determines what message, if any, should be sent
  - Sends using a byte stream protocol over TCP/IP
- EMS
  - Either on or off
  - Contact closure using a WebRelay device
  - HTTP protocol
- Flashing Beacon
  - Same as above

Logic Module

- Alert module
  - Instance for each alert script
  - Alert script retrieves data from input field element modules, performs logic, and writes to output field element modules

Software Architecture



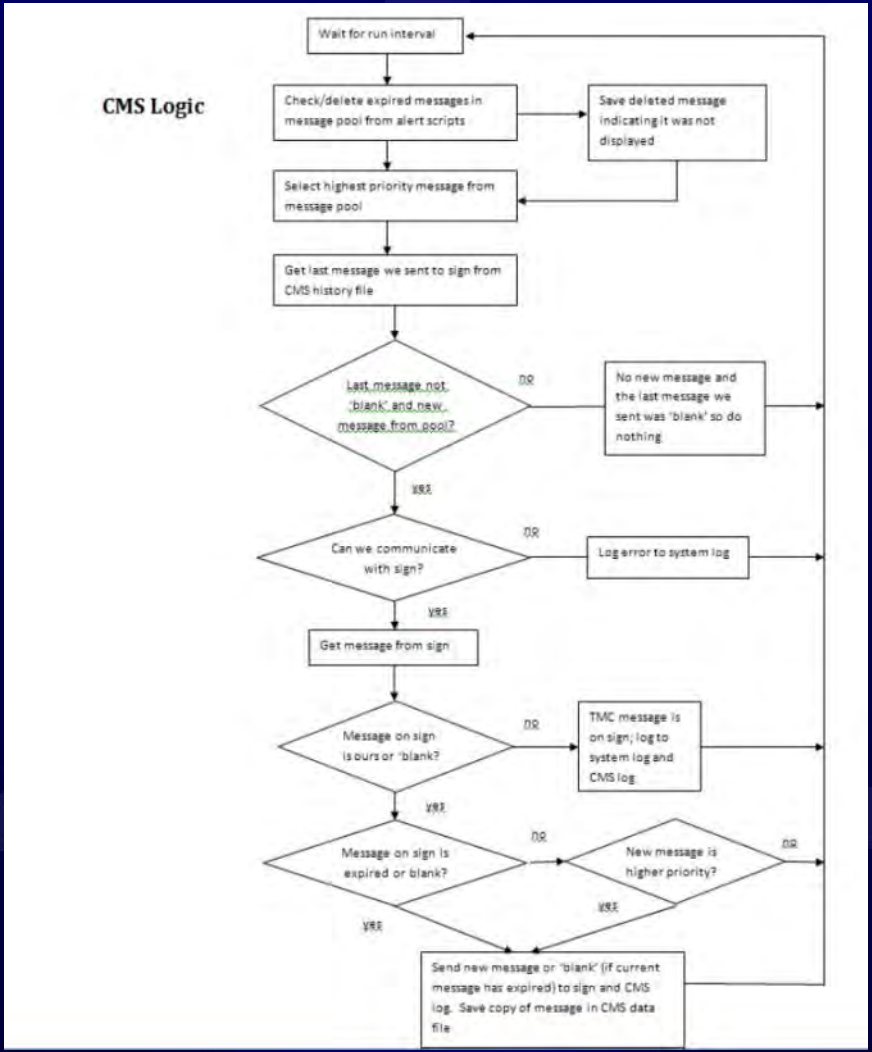
MONTANA STATE UNIVERSITY | College of ENGINEERING

Mountains & Minds

CMS Module

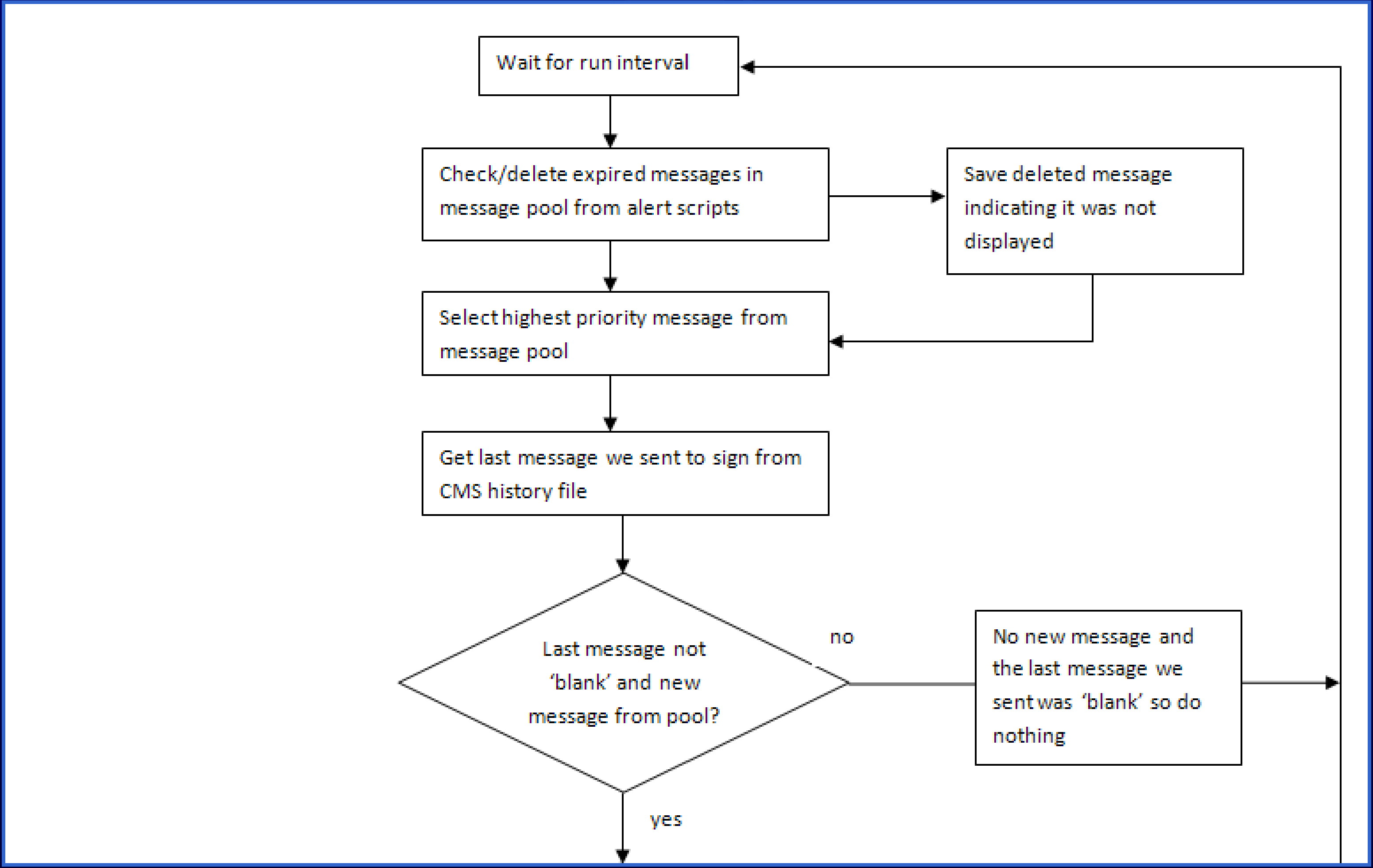
- Separate protocol handler (SV170Model500)
- Uses data module routines to read/write to data files
- Logs all activities for audit trail of events
- Contains logic to handle the control of the sign

CMS Module Logic

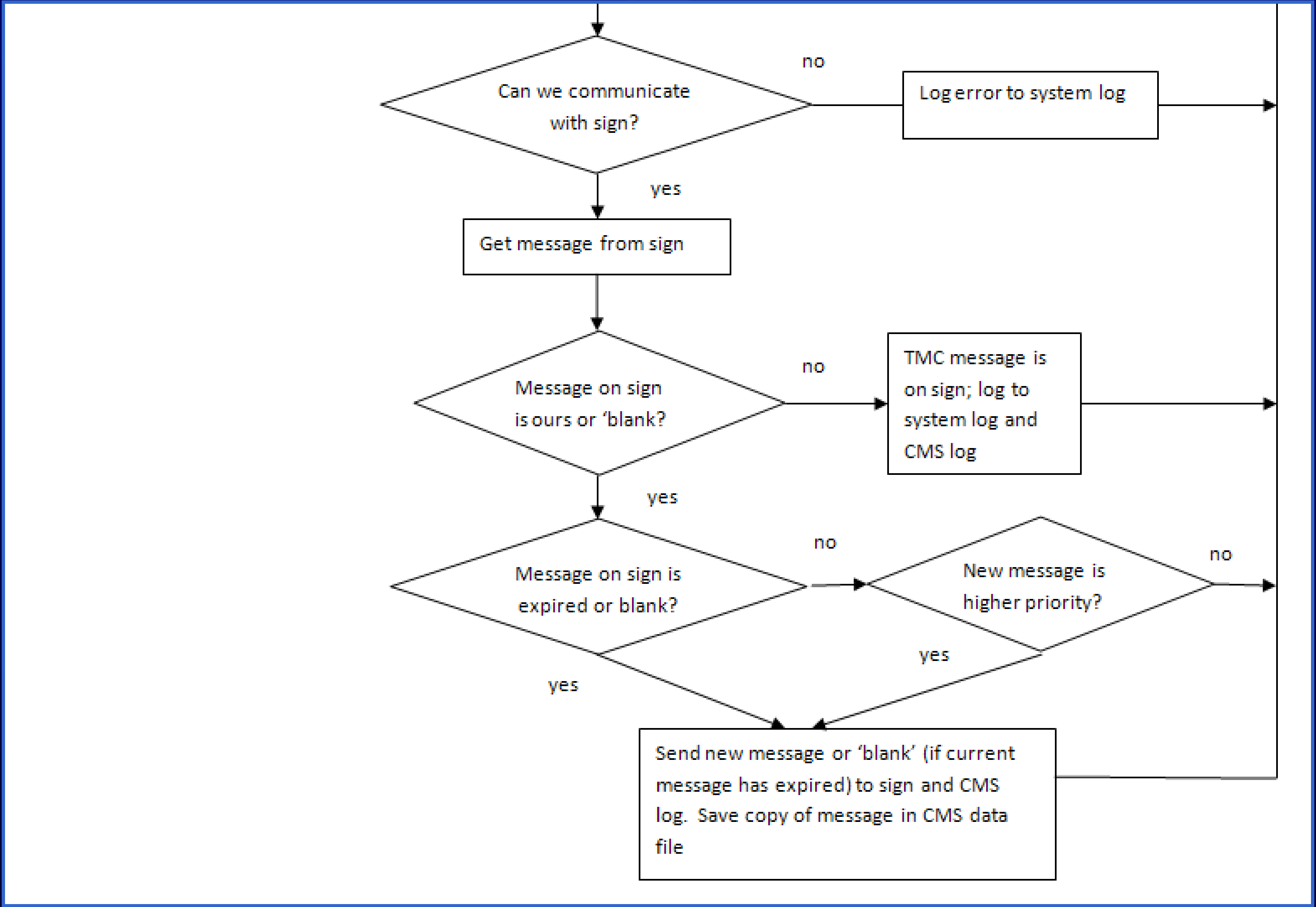


CMS Logic
Wait for run interval
Check/delete expired messages in message pool from alert scripts
Save deleted message indicating it was not displayed
Select highest priority message from message pool
Get last message we sent to sign from CMS history file
Last message not 'blank' and new message from pool?
no
No new message and the last message we sent was 'blank' so do nothing
yes
Can we communicate with sign?
no
Log error to system log
yes
Get message from sign
Message on sign is ours or 'blank?
no
TMC message is on sign; log to system log and CMS log
yes
Message on sign is expired or blank?
no
New message is higher priority?
no
yes
yes
Send new message or 'blank' (if current message has expired) to sign and CMS log. Save copy of message in CMS data file

CMS Module Logic


Wait for run interval
Check/delete expired messages in message pool from alert scripts
Save deleted message indicating it was not displayed
Select highest priority message from message pool
Get last message we sent to sign from CMS history file
Last message not 'blank' and new message from pool?
no
No new message and the last message we sent was 'blank' so do nothing
yes

CMS Module Logic



Can we communicate with sign?
no
Log error to system log
yes
Get message from sign
Message on sign is ours or 'blank?
no
TMC message is on sign; log to system log and CMS log
yes
Message on sign is expired or blank?
no
New message is higher priority?
no
yes
yes
Send new message or 'blank' (if current message has expired) to sign and CMS log. Save copy of message in CMS data file

Alert Script Design

- Alert script paradigm:
  get data → perform logic → output data
- Alert Module handles most house keeping tasks to simplify getting data and outputting data
- Tried to reach a balance between flexibility vs. ease of creation
- Scripts are written in Python so they integrate easily with the software

Alert Script Format

```
if FieldElementName.sensorname() comparison threshold:
    CMSSignName.MessageType = Page1Normal
    CMSSignName.FontPage1 = DoubleStroke
    CMSSignName.FontPage2 = SingleStroke
    CMSSignName.DisplayTime = 25
    CMSSignName.Priority = 10
    CMSSignName.Expiration = 600
    CMSSignName.MessagePage1Line1 = “Message First Line”
    CMSSignName.MessagePage1Line2 = “Message Second Line”
    CMSSignName.MessagePage1Line3 = “Message Third Line”
    CMSSignName.MessagePage2Line1 = “Message Forth Line”
    CMSSignName.MessagePage2Line2 = “Message Fifth Line”
    CMSSignName.MessagePage2Line3 = “Message Sixth Line”
    CMSSignName._testmode = False
    CMSSignName.logmsg = “put this message in the CMS log”
    CMSSignName.AddMessageToQueue()
```

Alert Script Example

```
FROST = 13                              #NTCIP essSurfaceStatus value for frost
ICEWARNING = 7                                     #NTCIP essSurfaceStatus value for ice warning

# Icy Curves Warning
if FredonyerPassEastRWIS.SurfaceStatus1() == ICEWARNING or
    FredonyerPassEastRWIS.SurfaceStatus2() == ICEWARNING:
        FredonyerPassEastCMS.logmsg = "Icy Curve Warning. Values: " +\
          str(FredonyerPassEastRWIS.essSurfaceStatus1()) +\
          str(FredonyerPassEastRWIS.essSurfaceStatus2())
        FredonyerPassEastCMS.MessagePage1Line1 = "ICY CURVES"
        FredonyerPassEastCMS.MessagePage1Line2 = "AHEAD"
        FredonyerPassEastCMS.MessageType = Page1Normal
        FredonyerPassEastCMS.FontPage1 = SingleStroke
        FredonyerPassEastCMS.Priority = 10
        FredonyerPassEastCMS.Expiration = 10 * MinuteInSecs
        FredonyerPassEastCMS.AddMessageToQueue()

 else:
        # Default Case - No Icy Curves Warning
        CMSlogger.info("No Icy Curve Warning")
```

Administration

- Four levels of security:
  - Operator
    - Monitor behavior and check for errors, but no changes permitted
  - Supervisor
    - Edit threshold variables
    - start/stop/pause modules
  - Technician
    - Modify alert scripts
    - Edit configuration files
    - Add/remove field elements
  - Administrator
    - Root access, full control over the device

Configuration

- Four configuration files:
  - AlertLogic.ini
    - Specifies alert scripts to use
    - Includes status, interval, and threshold variables
  - FieldElement.ini
    - Specifies field elements to use
    - Includes status, IP address, interval, list of sensors for input field elements
  - QualityControl.ini
    - Specifies minimum and maximum values for a unit
  - passwd
    - A single line that specifies the supervisor password

AlertLogic.ini

```
windwarning:
{
   status : 1
   initial_delay : 120
   interval : 60
   timeout : 2
   threshold:
   {
      highwindwarning : 56
      windwarning : 41
      windgustwarning : 46
      windadvisory : 26
      WindDistanceWest:5
   }
}
icewarning:
{
   status : 1
   initial_delay : 120
   interval : 60
   timeout : 4
   threshold:
   {
      freezetemp : 32.0
      precipforice : 1.25
   }
}
```

FieldElements.ini

```
SpringGardenRWIS:
{
        status : 1
        Type : "rwis"
        IPAddress : "10.20.199.132"
        LocalAddress : 1
        Port : "161"
        Timeout : 30
        Interval : 60
        AutoArchive  : 1
        MaxFileLength : 2000
        SavePeriod : 30
        DataTimeout : 340
        Location : { 'Lat' : 40.0024, 'Long' : 0-120.9579}
        County : "Plumas"
        Highway : "70"
        PostMile : "50.86"

                              …
```

FieldElements.ini (cont)

```
                    …

    sensorlist:
    {
        "essAirTemperature1"  : "degF"
        "essRelativeHumidity" : "pct"
        "essAvgWindDirection" : "degrees"
        "essAvgWindSpeed"     : "mph"
        "essMaxWindGustSpeed" : "mph"
        "essDewpointTemp"     : "degF"
        "essVisibility"       : ""
        "essSubSurfaceTemperature" : "degF"
        "essPrecipitationTwelveHours" : ""
        "essPrecipitationSixHours" : ""
        "essPrecipitation24Hours" : ""
        "essPrecipitationThreeHours" : ""
        "essPrecipitationOneHour" : ""
        "essSurfaceStatus1" : ""
        "essSurfaceStatus2" : ""
        "essSurfaceStatus3" : ""
        "essSurfaceTemperature1" : ""
        "essSurfaceTemperature2" : ""
        "essSurfaceTemperature3" : ""
    }
}
```

QualityControl.ini

```
degF:
{
		min: 0-80
		max: 150
}

pct:
{
		min: 0
		max: 100
}

degrees:
{
		min: 0
		max: 360
}

mph:
{
		min: 0
		max: 200
}
```

Take a drive through a potential application site …

Photo from Google Street View

Photo from Google Street View

Photo from Google Street View

Photo from Google Street View

Photo from Google Street View

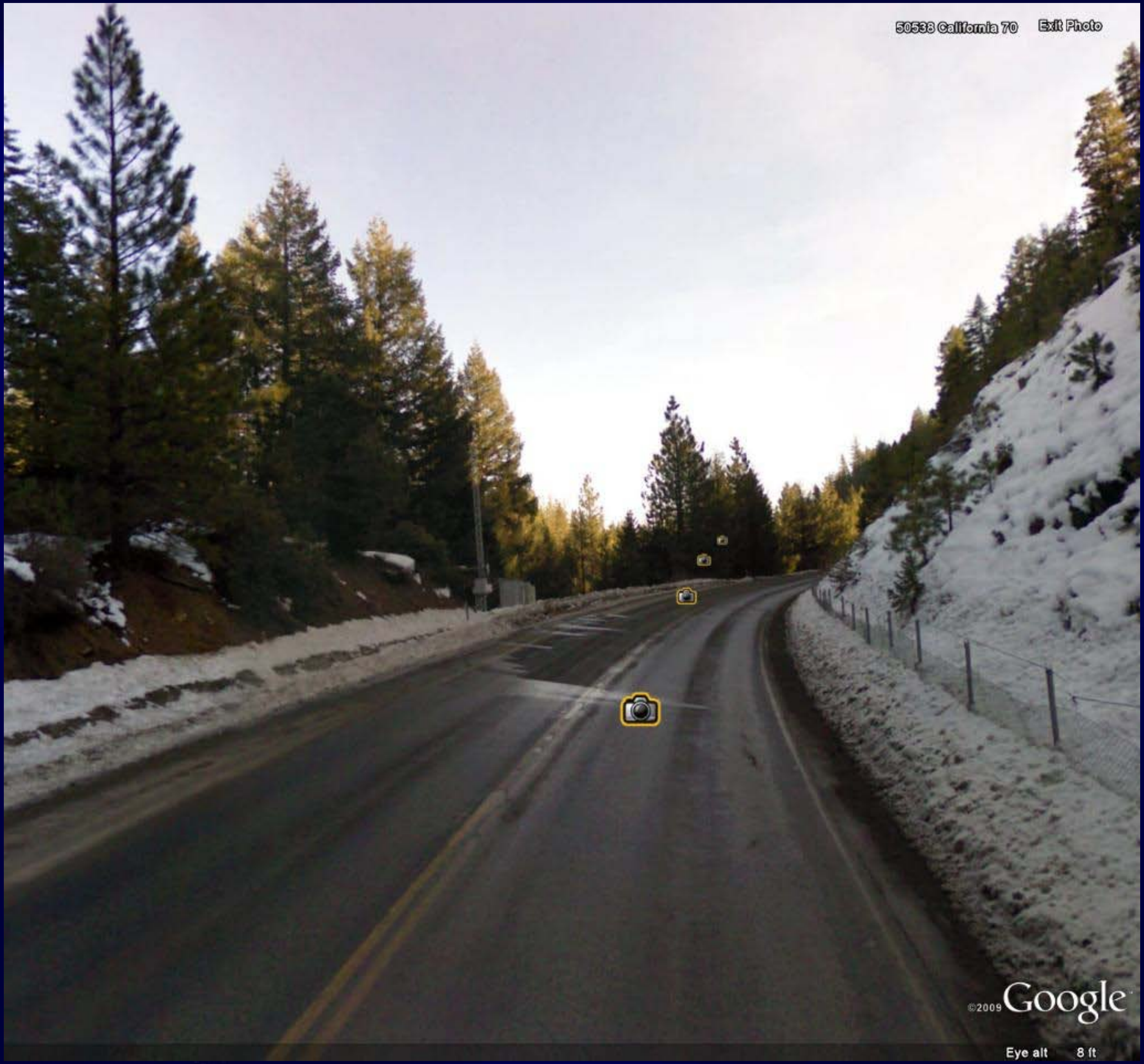


Photo from Google Street View

Photo from Google Street View

Photo from Google Street View

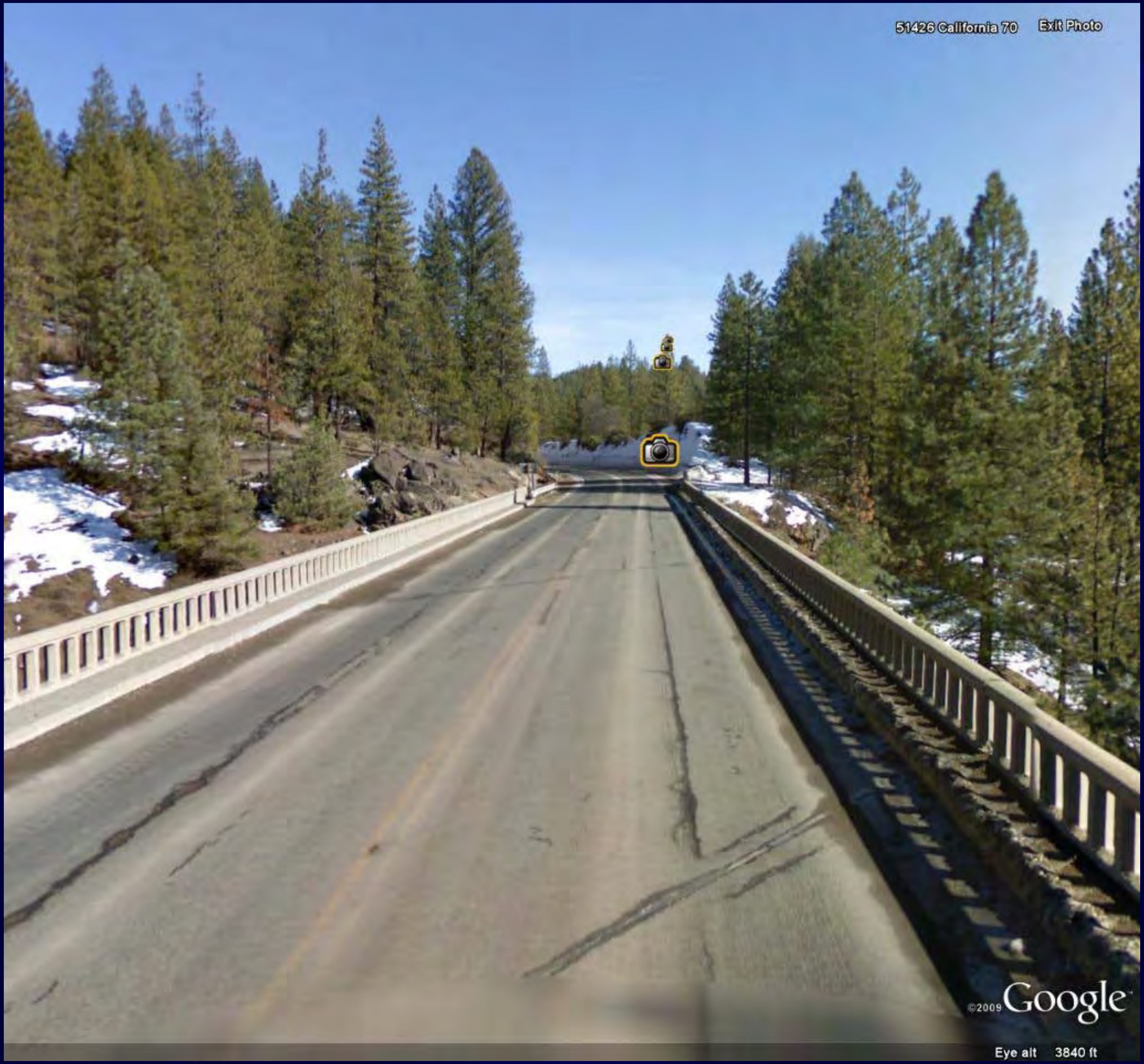


Photo from Google Street View

Photo from Google Street View

Photo from Google Street View

Testing/Evaluation

- In-house
  - Individual modules tested to verify communications, data parsing, etc.
  - Integration Testing
  - Dial in to Fredonyer Pass RWIS to get real data from real RWIS
- Pilot Field Testing
  - In lab at Caltrans D2
  - Field Site – Spring Garden

Evaluation

- Technical Performance
  - Log and data files examined to evaluate correctness
- Reliability
  - Long term testing in lab and at Spring Garden
- Usability
  - Survey sent to Ken Beals of Caltrans District 2 concerning ease of system setup and administration

Evaluation cont.

- Maintainability
  - Autonomous system, goal to require little maintenance
  - Tried to simplify as much as possible what maintenance tasks there were (scripts, config files, checking status and logfiles)
- Security
  - Minimize surface area
  - Only ssh should be necessary, http is optional

Data Logging

RWIS.csv

```
2009-11-09 03:13:42,4.03,4,0,17703,0,68,120,2.02,0,22.28,3,0,3,0,4,4,0.67,4,32.0,32.0,,33.8,32.72,77.9,32.54
2009-11-09 03:14:44,4.03,8,0,17703,0,67,100,2.02,0,22.1,3,0,3,0,4,4,0.67,4,31.82,77.9,,33.8,32.72,77.9,32.36
…
2009-11-09 16:19:18,1.79,3,0,17703,0,53,85,0.22,0,17.06,3,0,3,0,8,3,0.66,3,26.78,76.46,,32.9,77.9,76.46,76.64
2009-11-09 16:20:20,1.79,3,0,17703,0,53,85,0.0,0,17.06,3,0,3,0,4,3,0.65,3,26.78,76.64,,32.9,32.0,76.46,76.82
```

CMS.log

```
2009-11-08 19:14:06 - icewarning - INFO - No Icy Curve Warning
2009-11-08 19:14:07 - CMSWest - INFO - CMS - No new message, no last message.
2009-11-08 19:14:40 - CMSEast - INFO - CMS - No new message, no last message.
2009-11-08 19:15:06 - icewarning - INFO - Icy Curve Warning. Values: 8, 4, 4, 4, 3, 3
2009-11-08 19:15:06 - icewarning - INFO - Icy Curve Warning. Values: 8, 4, 4, 4, 3, 3
2009-11-08 19:15:09 - CMSWest - INFO - Sign Message Blank. Placing New CMS Message: ICY CURVES AHEAD
2009-11-08 19:15:42 - CMSEast - INFO - Sign Message Blank. Placing New CMS Message: ICY CURVES AHEAD
…
2009-11-09 08:19:24 - icewarning - INFO - Icy Curve Warning. Values: 3, 3, 3, 8, 3, 3
2009-11-09 08:19:47 - CMSEast - INFO - New Message and Sign Message Are the Same. Leaving Message
2009-11-09 08:20:02 - CMSWest - INFO - New Message and Sign Message Are the Same. Leaving Message
2009-11-09 08:20:24 - icewarning - INFO - No Icy Curve Warning
…
2009-11-09 08:39:26 - icewarning - INFO - No Icy Curve Warning
2009-11-09 08:40:19 - CMSEast - INFO - Blank Message sent to CMS
```

CMS.csv

```
2009-11-09 03:15:06.634551,1,1,1,10,ICY CURVES,AHEAD,,,,,10,1200,2009-11-09 03:15:41
…
2009-11-09 16:19:24.086605,1,1,1,10,ICY CURVES,AHEAD,,,,,10,1200,2009-11-09 16:39:11
2009-11-09 16:40:19,8,1,1,0, , , , , , ,0,0,2009-11-09 16:40:19
```

Data Logging

Timeline

```
From RWIS.csv:
   19:13:42 (03:13:42 UTC): RWIS reads all surface status sensors as either 3 (dry) or 4 (trace)
From CMS.log:
   19:14:06 (03:14:06 UTC): icewarning script reads RWIS data, reports No Icy Curve
From RWIS.csv:
   19:14:44 (03:14:44 UTC): RWIS reads all surface status sensors, SurfaceStatus1 is 8 (icewatch)
From CMS.log:
   19:15:06 (03:15:06 UTC): icewarning script reads RWIS data, generates Icy Curve Warning
From CMS.csv
   19:15:41 (03:15:41 UTC): CMSEast places Icy Curve message on sign
```

```
From RWIS.csv:
   08:19:18 (16:19:18 UTC): RWIS reads all surface status sensors, SufaceStatus4 is 8 (icewatch)
From CMS.log:
   08:19:24 (16:19:24 UTC): icewarning script reads RWIS data, generates Icy Curve Warning
From RWIS.csv:
   08:20:20 (16:20:20 UTC): RWIS reads all surface status sensors as either 3 (dry) or 4 (trace)
From CMS.log:
   08:20:24 (16:20:24 UTC): icewarning script reads RWIS data, reports No Icy Curve
```

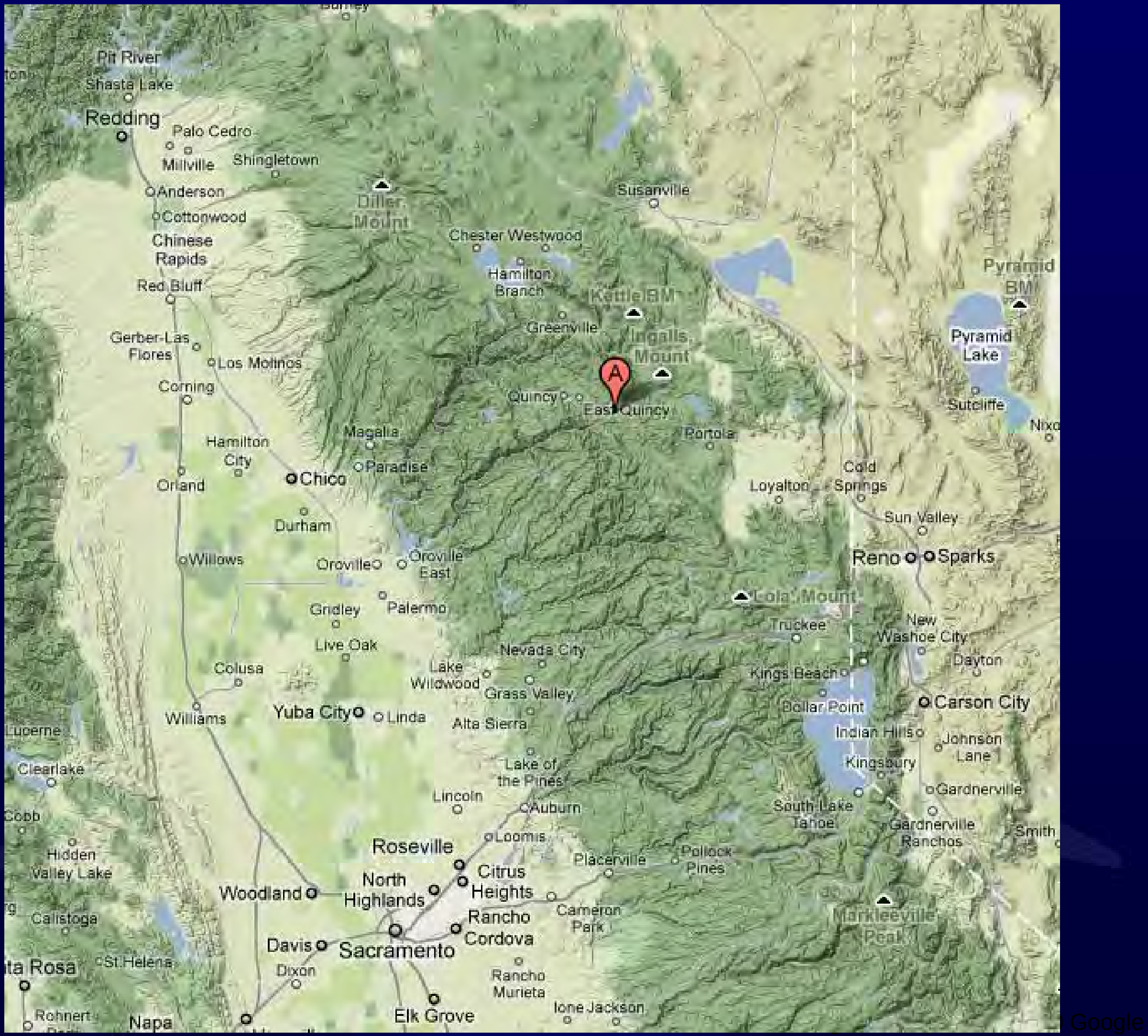
Data Logging

```
From CMS.csv:
   08:39:11 (16:39:11 UTC): last Icy Curve message is placed on sign
   08:39:24 (16:39:24 UTC): last Icy Curve message expires (20 minutes after it was created)
From CMS.log:
   08:39:26 (16:39:26 UTC): icewarning script reads RWIS data, reports No Icy Curve
From CMS.csv:
   08:40:19 (16:40:19 UTC): CMSEast places blank message on sign
```

Pilot Test - Spring Garden

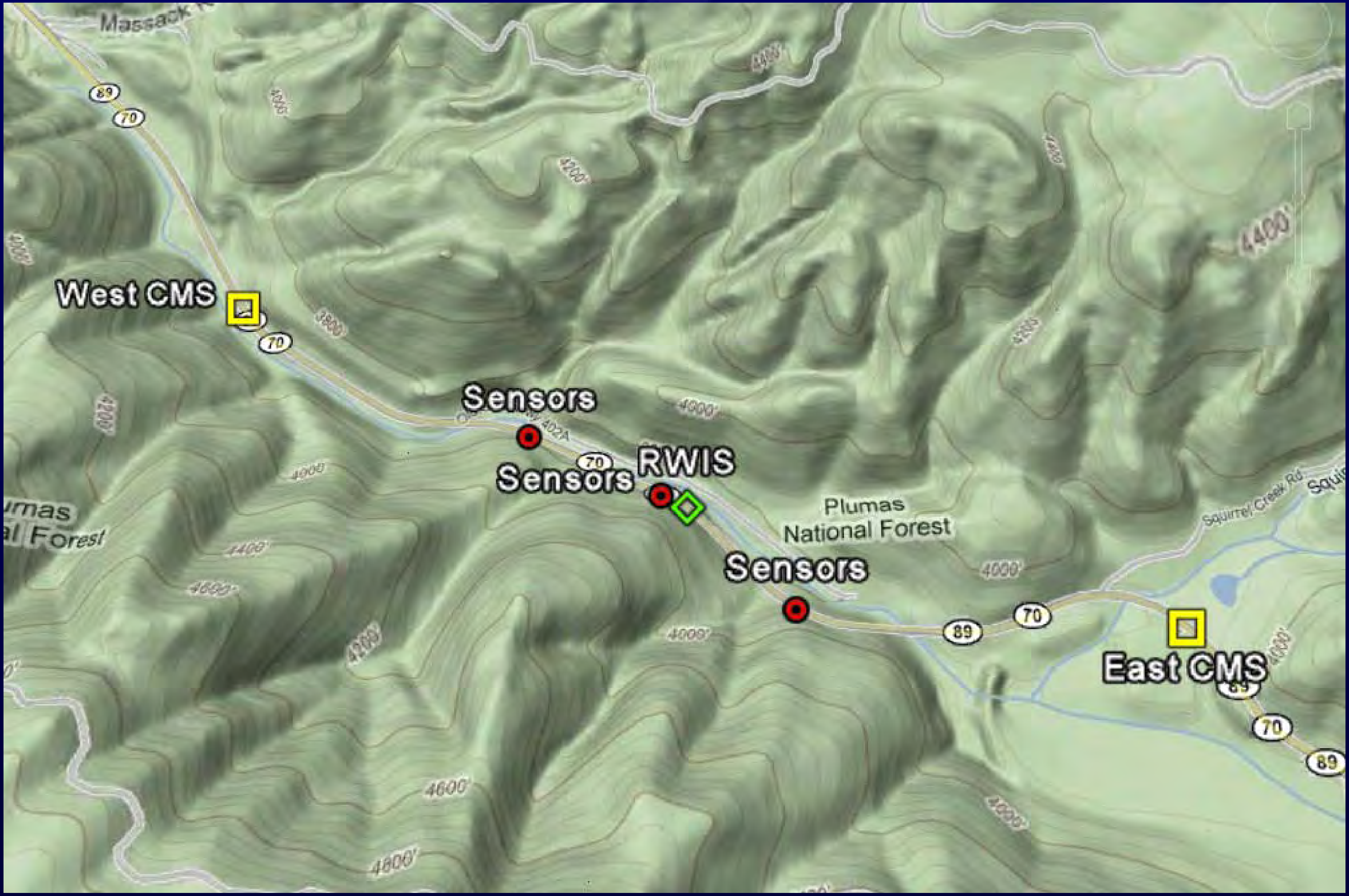
Photos from Ian Turnbull

Pilot Test - Spring Garden



Map from Google Maps

Pilot Test - Spring Garden



Map from Google Maps and Caltrans District 2 Data

Pilot Test - Spring Garden

Photos from Ian Turnbull

Pilot Test - Spring Garden

Photo from Doug Galarus

Photo from Ian Turnbull

Pilot Test - Spring Garden

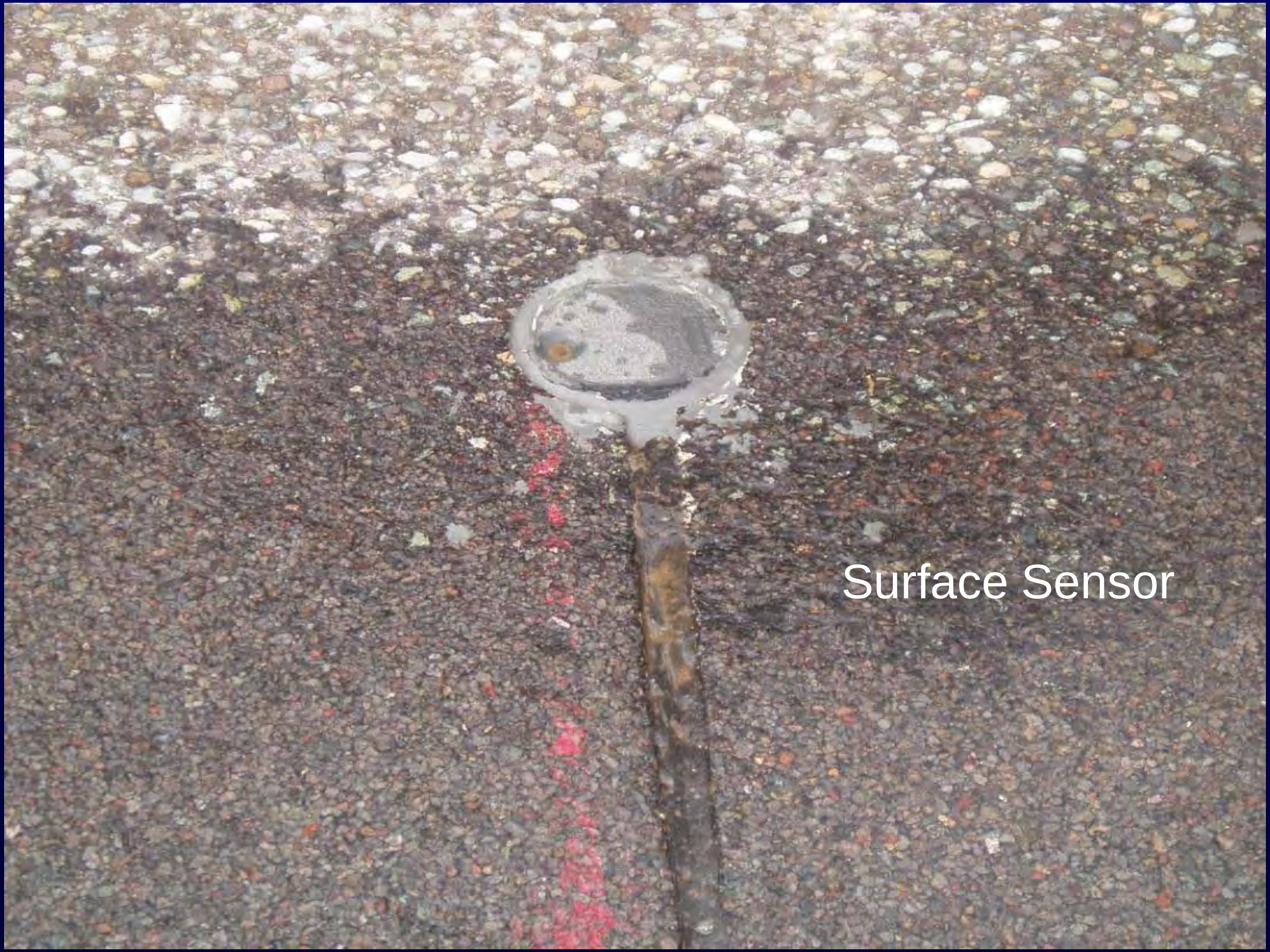


Photo from Dan Richter

Pilot Test - Spring Garden

Photos from Doug Galarus

Ice Warning Script for Spring Garden

```
#Name: Icy Curve Alert Script
#Reference: Caltrans District 2 Icy Curve Alert Parameters
#Date: 12-04-2008
#updated 25Aug2009 K. Beals
# added additional pucks and second CMS control
# added ice watch surface condition
# updated 11Dec2009
# changed sign message to "CAUTION ICY ROAD" in double stroke, flashing
# Gerry Reyes determined message

ICEWARN = 7                             #NTCIP essSurfaceStatus value for ice warning
ICEWATCH = 8                                                       #NTCIP essSurfaceStatus value for ice watch

FROST = 13                              #NTCIP essSurfaceStatus value for frost

SurfaceIceWarning = False
SurfaceIceWatch = False
SurfaceFrost = False

SignTest = test                                      # value can be changed from TMC for testing CMS displays

SurfaceIceWarning = RWIS.essSurfaceStatus1()==ICEWARN or \
                    RWIS.essSurfaceStatus2()==ICEWARN or \
                    RWIS.essSurfaceStatus3()==ICEWARN or \
                    RWIS.essSurfaceStatus4()==ICEWARN or \
                    RWIS.essSurfaceStatus5()==ICEWARN or \
                    RWIS.essSurfaceStatus6()==ICEWARN
```

Ice Warning Script for Spring Garden

```
SurfaceIceWatch = RWIS.essSurfaceStatus1()==ICEWATCH or \
                  RWIS.essSurfaceStatus2()==ICEWATCH or \
                  RWIS.essSurfaceStatus3()==ICEWATCH or \
                  RWIS.essSurfaceStatus4()==ICEWATCH or \
                  RWIS.essSurfaceStatus5()==ICEWATCH or \
                  RWIS.essSurfaceStatus6()==ICEWATCH

SurfaceFrost = RWIS.essSurfaceStatus1()==FROST or \
               RWIS.essSurfaceStatus2()==FROST or \
               RWIS.essSurfaceStatus3()==FROST or \
               RWIS.essSurfaceStatus4()==FROST or \
               RWIS.essSurfaceStatus5()==FROST or \
               RWIS.essSurfaceStatus6()==FROST

CMSEast.MessageType = Page1Flash
CMSEast.FontPage1 = DoubleStroke
CMSEast.DisplayTime = 10     #in 1/10 seconds
CMSEast.Priority = 10
CMSEast.Expiration = 20 * MinuteInSecs

CMSWest.MessageType = Page1Flash
CMSWest.FontPage1 = DoubleStroke
CMSWest.DisplayTime = 10     #in 1/10 seconds
CMSWest.Priority = 10
CMSWest.Expiration = 20 * MinuteInSecs
```

Ice Warning Script for Spring Garden

```
# Icy Curves Warning
if SurfaceIceWarning or SurfaceIceWatch or SurfaceFrost or SignTest:
        CMSEast.logmsg = "Icy Road Warning. Values: %d, %d, %d, %d, %d, %d" %(RWIS.essSurfaceStatus1(),\
                     RWIS.essSurfaceStatus2(),\
                     RWIS.essSurfaceStatus3(),\
                     RWIS.essSurfaceStatus4(),\
                     RWIS.essSurfaceStatus5(),\
                     RWIS.essSurfaceStatus6())
        CMSEast.MessagePage1Line1 = "CAUTION"
        CMSEast.MessagePage1Line2 = "ICY ROAD"
        CMSEast.AddMessageToQueue()
        CMSWest.logmsg = "Icy Road Warning. Values: %d, %d, %d, %d, %d, %d" %(RWIS.essSurfaceStatus1(),\
                     RWIS.essSurfaceStatus2(),\
                     RWIS.essSurfaceStatus3(),\
                     RWIS.essSurfaceStatus4(),\
                     RWIS.essSurfaceStatus5(),\
                     RWIS.essSurfaceStatus6())
        CMSWest.MessagePage1Line1 = "CAUTION"
        CMSWest.MessagePage1Line2 = "ICY ROAD"
        CMSWest.AddMessageToQueue()

# Default Case - No Icy Curves Warning
if not SurfaceIceWarning and not SurfaceIceWatch and not SurfaceFrost:
    CMSlogger.info("No Icy Curve Warning")
```

Automated Safety Warning Controller Pilot Test

- Controller was pilot tested at Spring Garden
- Pilot test was limited to RWIS and CMS field elements
- Pilot test was limited to Ice Warning alert script
- Device was installed by Ken Beals of Caltrans District 2
  - Installation of field elements
  - Network configuration
  - Alert scripts

Pilot Test

Fixes and Changes

- Added a sign test option to the front panel interface at Ken's request
- CMS DisplayTime was off by a factor of 10
- Python logging module has a bug if more than one rotate interval passes between messages being logged
- CMS messages were logged when placed on sign *and* when deleted. A subtle timing issue would make the Controller think a different message was on the sign than there was.
- There was an issue with archiving of the data files. Files are archived on read, but the ice warning script only reads from memory, so the RWIS file wasn't being archived.

General lessons learned

- Python as a language?
  - Good for rapid prototyping
  - A lot of functionality built in to the language
- Flat files vs. SQLite
  - SQLite is faster than just reading a flat file, but with some buffering techniques flat files become much faster
- Buffering data in memory vs. reading files
  - Managing buffers is complex
  - Offers significant speed improvements

Automated Safety Warning Controller Phase 2

Kickoff meeting conducted May 25th, 2010 at Caltrans District 2 headquarters

Attendees:

- Ian Turnbull
  - Chief, Office of ITS Engineering and Support Caltrans District 2
- Sean Campbell
  - Chief, ITS Special Project Branch
  - Caltrans Division of Research and Innovation
- Ken Beals
  - ITS Engineer, Caltrans District 2
- Mohammad Battah
  - Electrical Systems Branch, Caltrans District 10
- Doug Galarus, Dan Richter, Kelvin Bateman
  - Western Transportation Institute

Automated Safety Warning Controller
Phase 2 Goals

- Outcome of Phase I of this project was a prototype hardware and software system that is being tested in the field
- Outcome of Phase II will be a hardware and software system that has been pilot tested by multiple users in the field and has been prepared for wider deployment
- Additional research and development will be conducted to prepare for deployment including fault tolerance of system software, production of training materials and other documentation, and preparation of business case material to assist in deployment justification

Automated Safety Warning Controller Phase 2 Tasks

- Review Phase 1 Results
  - System Review
    - Use results from Phase 1 to review system architecture and performance
  - Evaluation Review
    - Identify gaps in Phase 1 functionality and items deferred to Phase 2
  - Solicit input from Caltrans
    - Identify additional functionality or performance issues that need to be addressed

Automated Safety Warning Controller Phase 2 Tasks

- Review Phase 1 Results (continued)
  - Submit a review summary and recommendations document to Caltrans
- On-going Development
  - Software will be re-factored to make it more robust and ready for production use
  - Additional functionality will be added and lab tested

Automated Safety Warning Controller Phase 2 Tasks

- Lab Enhancement
  - Any necessary equipment to support the additional functionality will be purchased by WTI or loaned by Caltrans to WTI for the project

Automated Safety Warning Controller Phase 2 Tasks

- Add additional Field Element Modules
  - Work with Caltrans to identify additional elements
    - Complete work started in Phase 1 on loop detector and RTMS
    - Wavetronix microwave vehicle detection
    - EIS X3 microwave vehicle detection
    - Video detection
    - HAR
    - EMS
    - Flashing Beacon

Automated Safety Warning Controller Phase 2 Tasks

- Integration with SOCCS
  - SOCCS Automated Controller will provide administration interface to TMC
  - Implement hooks and protocols for SOCCS software to communicate with the device
  - Work with Caltrans to develop a SOCCS interface

Automated Safety Warning Controller Phase 2 Tasks

- Research implementing real-time processing
  - Evaluate current response time
  - Evaluate software architecture and applicability to real-time processing
  - Increase poll rate or make event based
- Research additional applications
  - Ramp metering
  - Other?
- Revise the User Guide
- Revise the System Management and Maintenance Guide

Automated Safety Warning Controller Phase 2 Tasks

- Pilot testing
  - Conduct outreach to identify prospective users from within Caltrans interested in participating in pilot testing
  - Develop training materials and conduct training sessions to prepare crews for pilot testing
  - Additional pilot test sites
    - Evaluate and install at potential test sites

Automated Safety Warning Controller Phase 2 Tasks

- Evaluation
  - Evaluation criteria will include, but not be limited to: technical performance, reliability, usability, maintainability, and security
  - An evaluation summary will be submitted to Caltrans summarizing methods and results

Automated Safety Warning Controller
Acknowledgements

Ian Turnbull – Project Champion

Sean Campbell – Project Manager

Ken Beals – Caltrans District 2 ITS Engineer

Doug Galarus – Principal Investigator

Others …

Potential Applications Discussion

- Sites?
- Situations?

Demonstration

Questions?

More information and future updates can be found at www.westernstates.org