Western States 2011

Rural Transportation Technology Implementers Forum

IRIS Open Advanced Traffic Management System

Development, Deployment, Capabilities, and Maintenance

June 14-16 2011

Michael Darter

Software Engineer • Formerly with AHMCT University of California

Kin Yen

Development Engineer • AHMCT University of California

John Castro, PE

Sr. Transportation Electrical Engineer • Caltrans • District 10

Roadmap

John
- ATMS introduction and importance
- Timeline
- Before and after overview
- Benefits
- Caltrans contributions to IRIS
- Collaboration

Michael
- Caltrans IRIS research project
- IRIS background
- What worked well and didn't
- IRIS functional overview + demo
- AWS
- VDS
- System configuration and customization
- Architecture + design
- Software engineering and testing
- Costs

Kin
- Video
- System maintenance
- Mapping
- Scalability
- Current and future enhancements

What is an ATMS?

An Advanced Transportation Management System (ATMS) is a software tool that provides Transportation Management Center (TMC) operators and Traffic Managers with a real time view of highway conditions so that accurate and timely actions can be performed in response to adverse environments or traffic incidents.

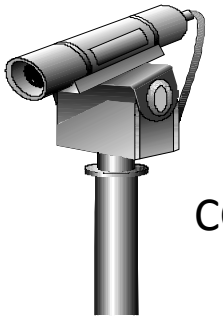
Why an ATMS?

An ATMS allows Caltrans to:

1. Effectively manage the freeways
2. Reduce traveler commuting times
3. Maximize roadway capacity
4. Provide a safer traveling medium for the general public.

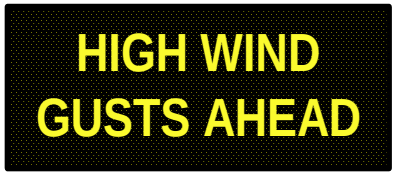
john_castro@dot.ca.gov

ATMS Real-Time Functions



CCTV

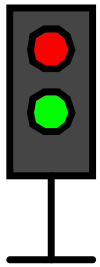
CMS



VDS

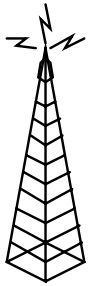
Traffic Management Tools:

Ramp Metering



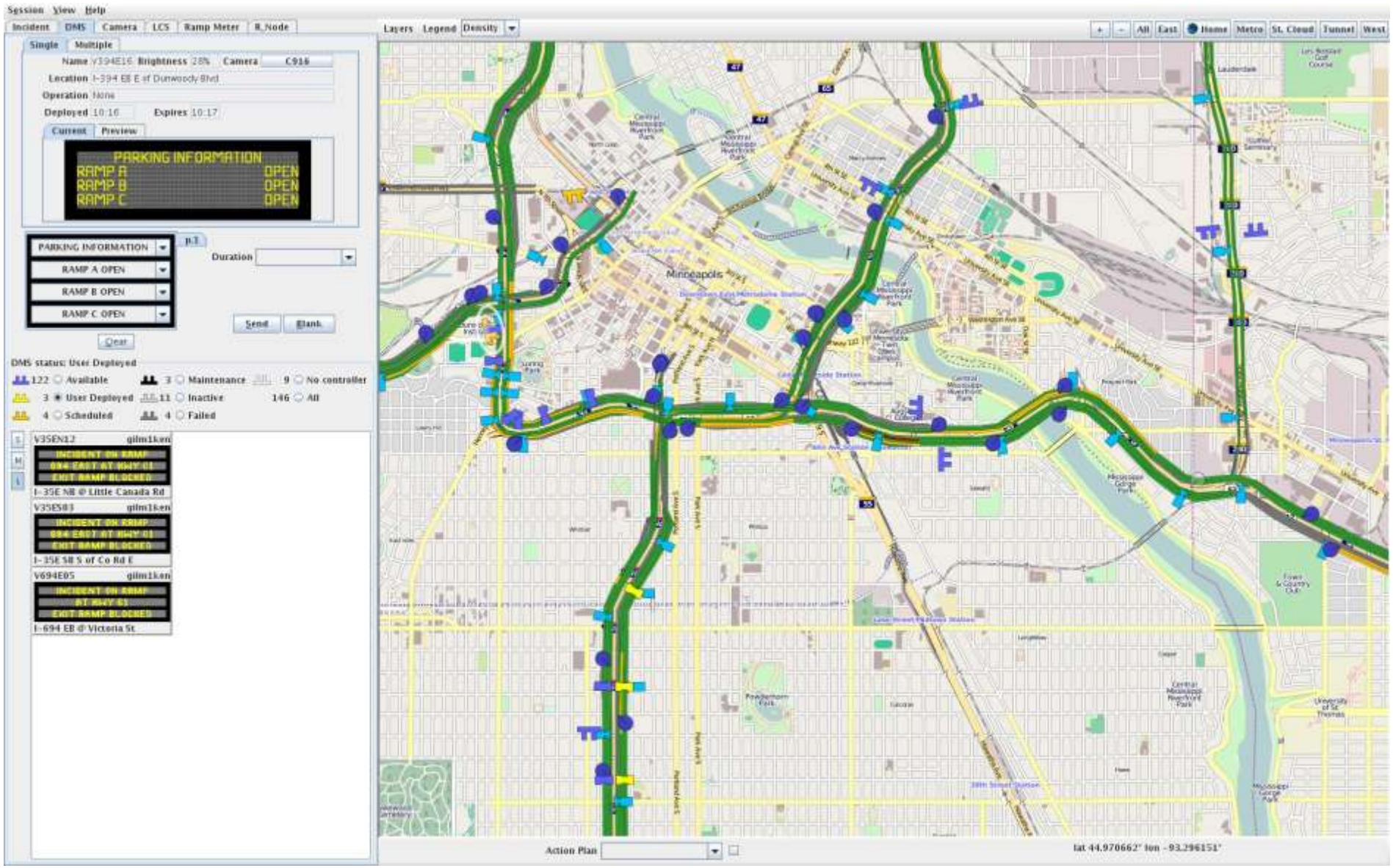
RWIS

HAR



john_castro@dot.ca.gov

Screenshot of IRIS DMS Functionality



john_castro@dot.ca.gov

Caltrans District 10 + IRIS Timeline

Original CAWS Deployed 1996

Mn/DOT IRIS development starts early 1990s

Middleware deployed 2006

Mn/DOT releases IRIS source code 05/2007

D10 IRIS Pilot Project 2007

Middleware malfunctions 2009-2010

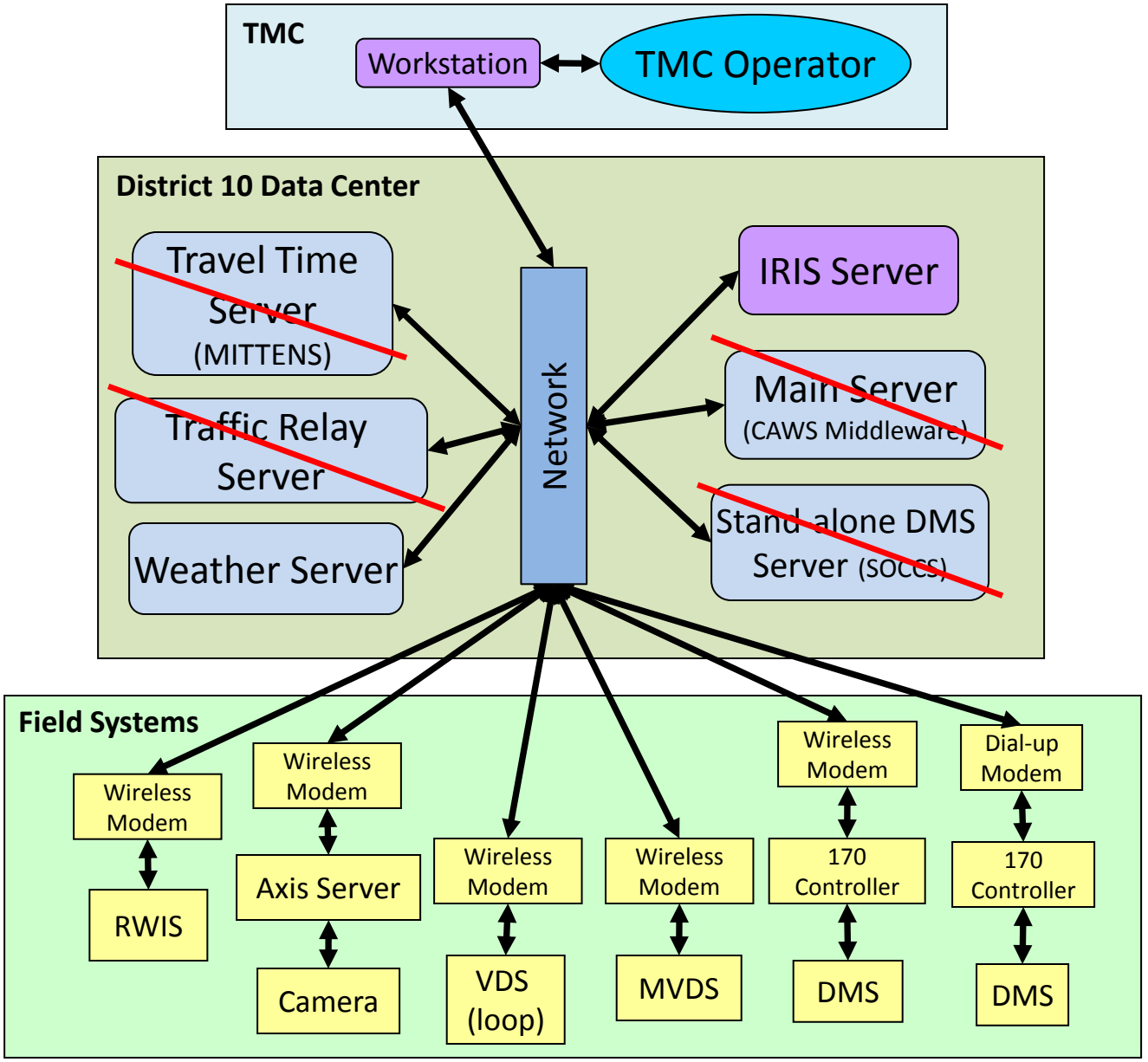
IRIS AWS replaces Middleware 2/2011

Caltrans IT acceptance (FSR) of IRIS

IRIS Deployment in D1, D2, D5 Fall 2011

john_castro@dot.ca.gov

Caltrans District 10 IRIS Architecture Before and After



john_castro@dot.ca.gov

Benefits of IRIS

Before and After Comparison

	Before	After
Mapping	None	✓
Number of server machines	4	1
Number of CMS controlled by AWS	9	28 (unlimited*)
Types of VDS supported (MVDS, Loops, etc)	1	5+ (unlimited*)
Types of RWIS supported (Manufacturers)	1	2+ (unlimited*)
Types of CMS supported (Manufacturers)	1	4+ (unlimited*)
Mapped incidents	Via stand-alone app	Integrated
Standby backup system	None	✓
Source code availability	None or proprietary	✓
D10 traffic engineer's maintenance time	50%	

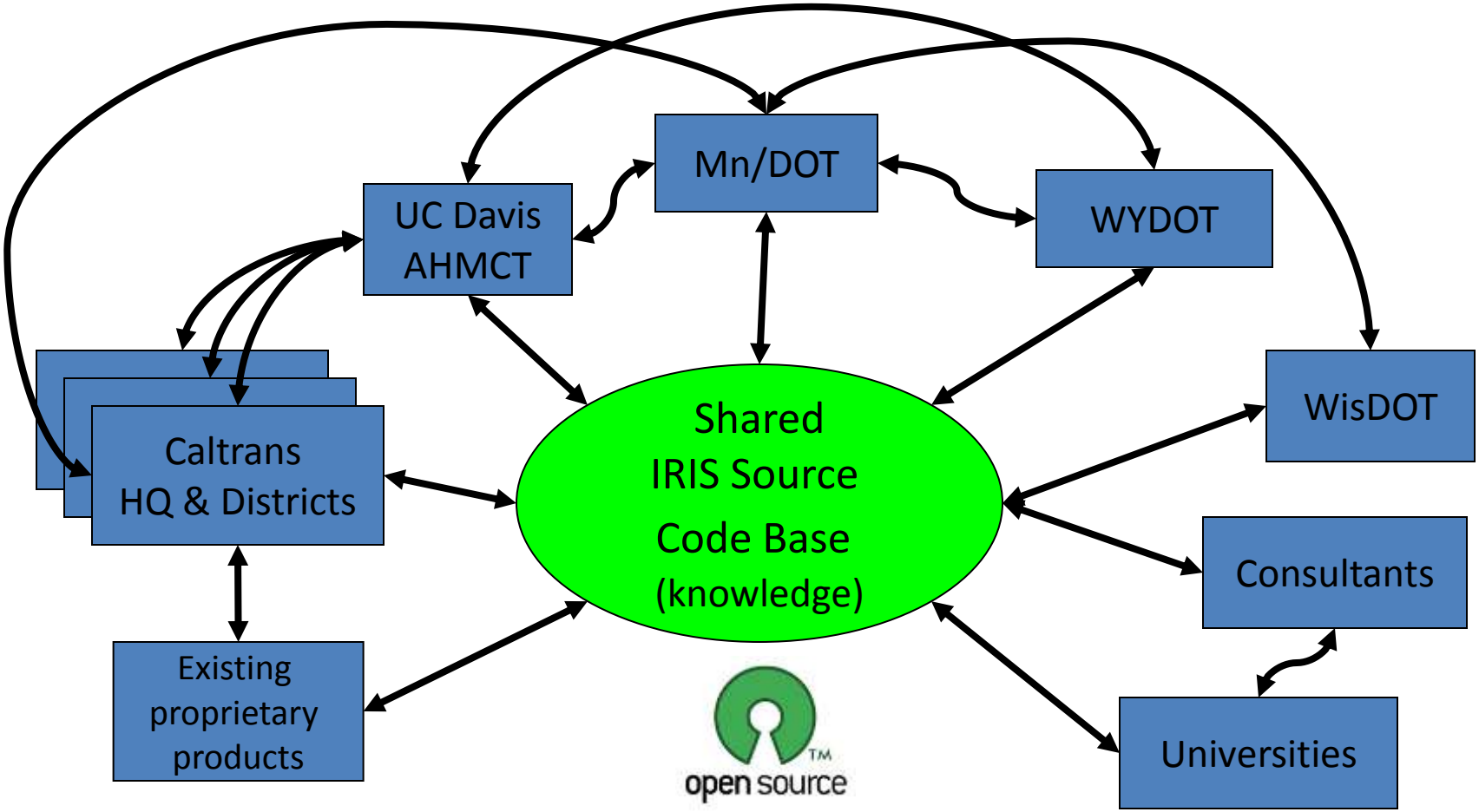
*via device driver interface

john_castro@dot.ca.gov

Caltrans Contributions to IRIS

- Device drivers
  - VDS: MVDS (EIS RTMS), URMS 2070, Wizard
  - CHP Incident
  - RWIS (SSI)
  - PeMS
  - Integration with external AWS (no longer used by D10)
- Automated Warning System (AWS)
- Testing
  - Automated unit test cases
  - CMS simulation
- Generalization of IRIS, e.g. system attributes
- CMS message library
- Google Earth output
- RWIS map integration
- IRIS Developer Ticketing System

Multi-agency Collaboration



- Knowledge Base is estimated at approx. $4.2 Million.
- 4 DOTS using or evaluating the use of IRIS as their ATMS

john_castro@dot.ca.gov

Michael Darter
Software Engineer
Formerly with AHMCT University of California

Discussion topics…

- IRIS background
- What worked well and didn't
- IRIS functional overview and demo
- Automated Warning System (AWS)
- Vehicle Detector Stations (VDS)
- System configuration and customization
- Architecture + design
- Software engineering and testing
- Costs

michael@bt-systems.com

Caltrans IRIS Research Study

The Research Problem…

- ATMS not deployed in 5 of 12 Caltrans Districts
- Why? Cost

Potential solution?

- Implement the open-source IRIS ATMS in D10
- Extend & customize IRIS for other districts

The Result…

- Provides ATMS capabilities
- Extendible, reliable, scalable
- Reduced life cycle costs 72%
- High relevance for ITS research
- Importance of open development process

Origin of IRIS ATMS

- IRIS developed by the Minnesota Department of Transportation (Mn/DOT)
  - Development started in the early 1990s
  - Used in Minneapolis/St. Paul, St. Cloud, Rochester
    - 135 DMS, 476 cameras, 5452 VDS, 433 ramp meters, 4 RWIS, 194 LCS,
      1 Lane Marking (in-road lighting), 2 static signs with wig-wag beacons
  - 100% developed in-house
  - Significant investment (COCOMO: > $4 million)
- Motivation for open-sourcing
  - Insure affordable and manageable longevity of IRIS
  - Collaboration with other transportation agencies
    - Source contributions back to IRIS
  - Additional IRIS developers
    - Lowers risks
- Released as open-source (GPL) 05/2007
- 4+ agencies using or evaluating

What Was Challenging?

- Managing ticket priorities: defer or fix?
- Bottom up versus top down
- Unexpected regressions
  - Automated test cases helped
  - Need end-to-end automated test cases
- Lack of mid-level technical documentation
- Tradeoff between generalization and agency-specific code
  - Writing agency-specific code is easy
  - Generalizing an existing feature can be complex
  - Code merges are voluntarily (in both directions)
- Timing merges between agencies
- Excellent generalized designs take additional effort
  - E.g. RWIS icons: metric, English, physical quantity classes
  - E.g. Incidents
- Communicating how the open-source process is different
  - Collaborative development model
  - Cost savings are easy to explain

michael@bt-systems.com

What Went Wrong?

- IRIS 9.0 release
  - 4 defects caused client freezing and termination (threading issues)
  - Defects were regressions due to rewriting DMS code
  - Defects only visible on Windows client
  - Defects were not visible at Mn/DOT or during developer testing
  - Short-term solution: over 2-3 weeks released ~3 upgrades that fixed problems (UCD & Mn/DOT patches)
  - Long-term solutions: GUI testing on Windows only, improved communication w/ Mn/dot, developer ticketing system
- Client defects when user manipulates multiple CMS
  - A series of ~6 defects causing client termination (NPE) when user selects multiple CMS
  - Cause: could not test IRIS w/ multiple DMS except in production → not enough integration testing
  - Solution: DMS simulator (CASPER)
  - Future: implement DmsXml protocol in CASPER so other agencies can use multiple simulated DMS

michael@bt-systems.com

What Went Wrong?

- Video uptake in D10
  - Software problems
    - Reliability issues with Pelco PTZ driver from 2008
    - GUI issue: can't tell if camera should be working or in maintenance
    - GUI issue: no GUI feedback when connecting to cameras
    - GUI issue: couldn't configure presets (only in the field)
  - Result: at some point, operators stopped using PTZ and rarely viewed cameras
  - Prioritization of fixes was low, not enough resources
  - Solution: priority of camera work increased, lab testing camera / controller

michael@bt-systems.com

What worked well?

- Providing ATMS functionality with IRIS
  - Ability to customize for an agency is endless
- IRIS takes over AWS (4 months start to end)
  - Developing VDS data acquisition (RTMS, loops)
  - Developing RWIS data acquisition + user interface
  - Forwarding traffic to PeMS
  - Developing AWS module
  - Testing and verification
- Reliability, code quality
  - Particularly on the server
- Low cost
- Collaborative design
- Ability to customize and generalize
- Managing code between multiple agencies
  - Has worked surprisingly well (w/ exceptions)
  - Change-set concept works well

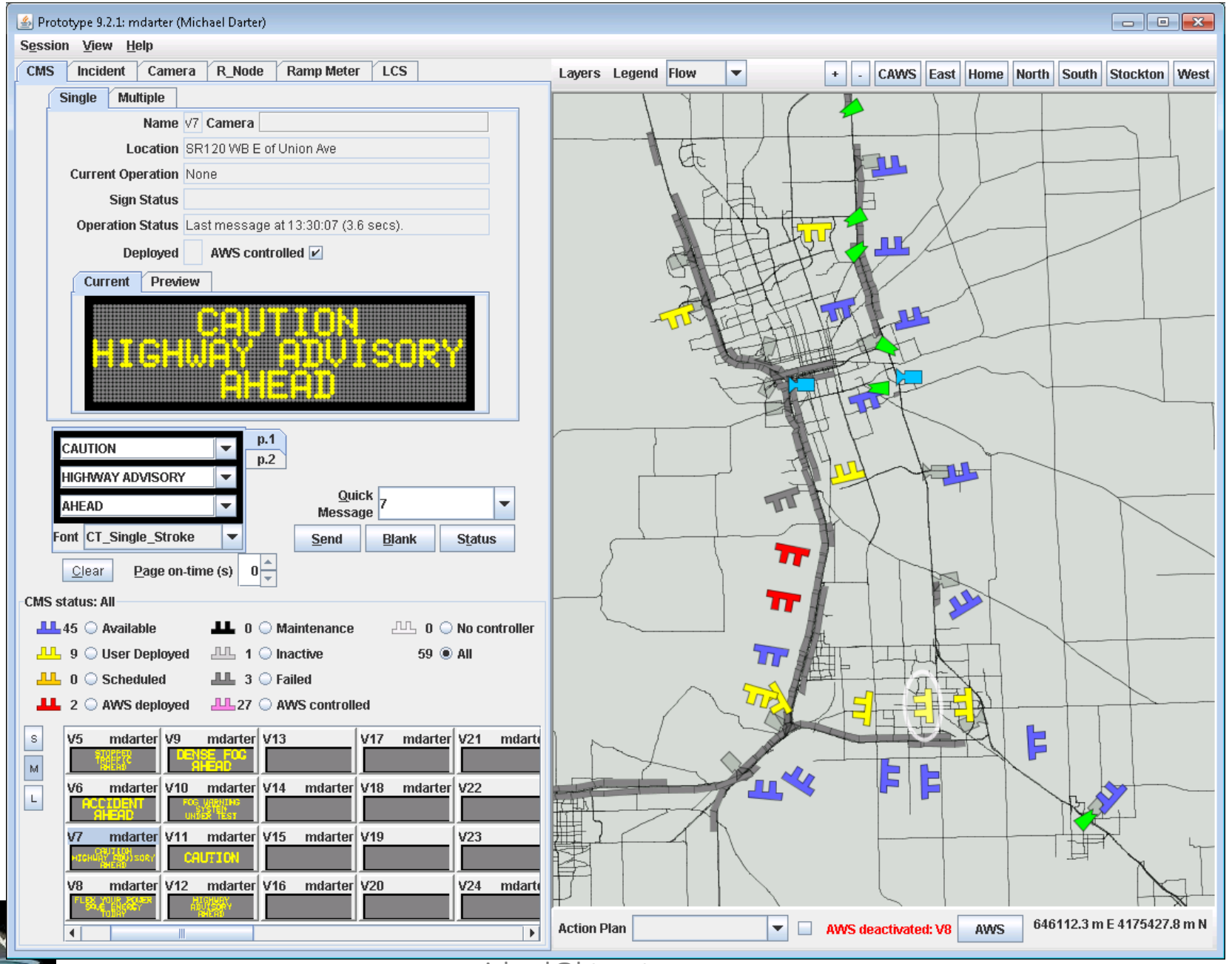
michael@bt-systems.com

IRIS Functional Overview

(demo)

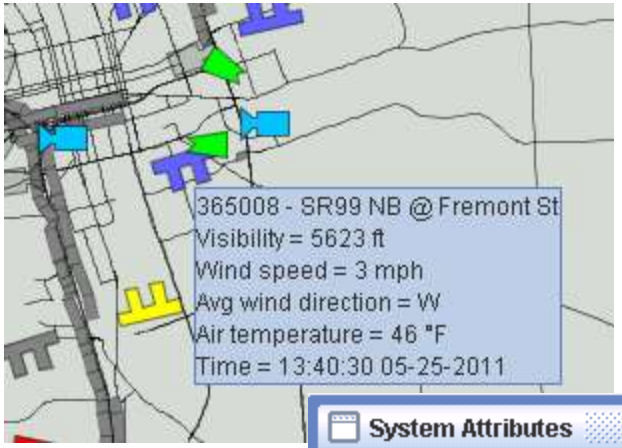
- Mapping
- AWS: operation, configuration
- Traffic
  - Real-time speed, flow, density per lane or station
  - Configuration: controllers, comm links, detectors, stations
- DMS
  - Sending messages, blanking, get-status
  - Message libraries, diagnostics, configuration
- RWIS: operation, configuration
- Video: operation, PTZ, configuration
- Configuration: drivers, system attributes, properties files
- Travel time, scheduled messages, variable speed limits
- Incidents: operation, configuration
- Ramp meters, LCS
- Security: user permissions, LDAP

IRIS Screenshot – DMS Control and Monitoring

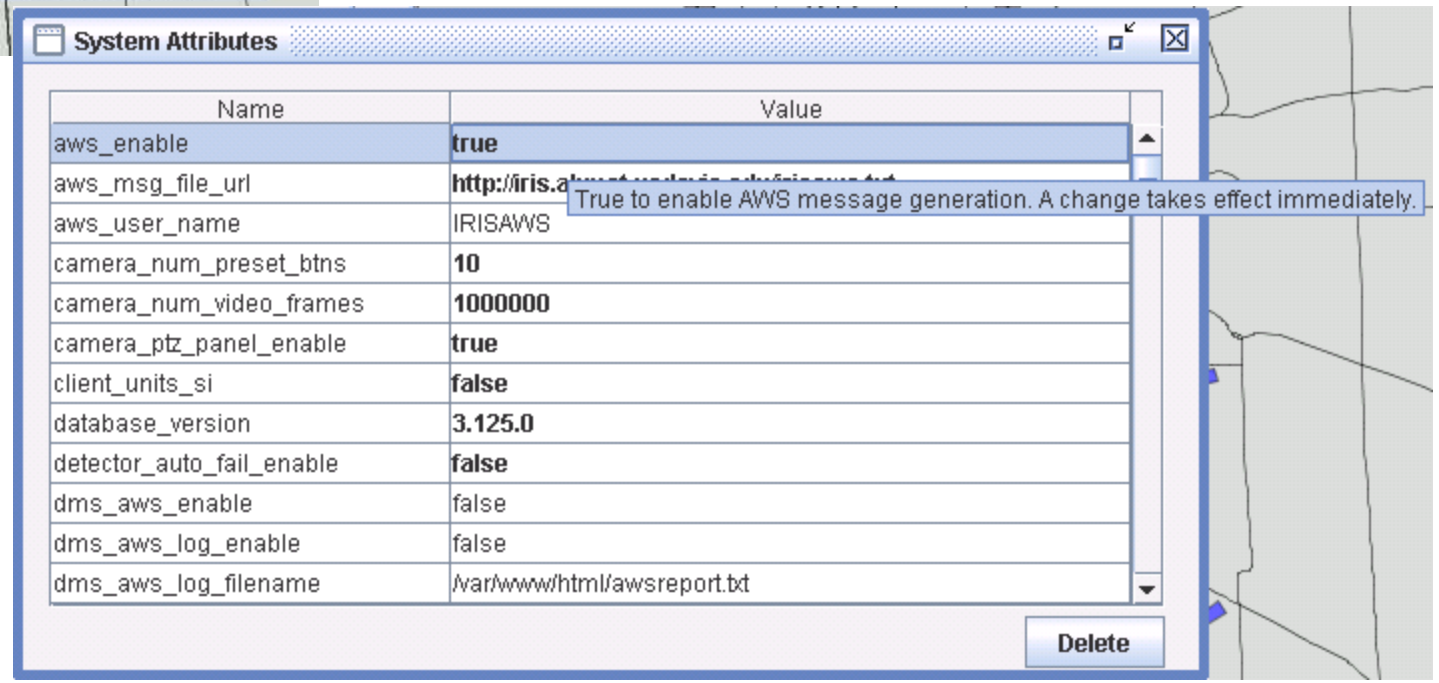


IRIS Screenshots

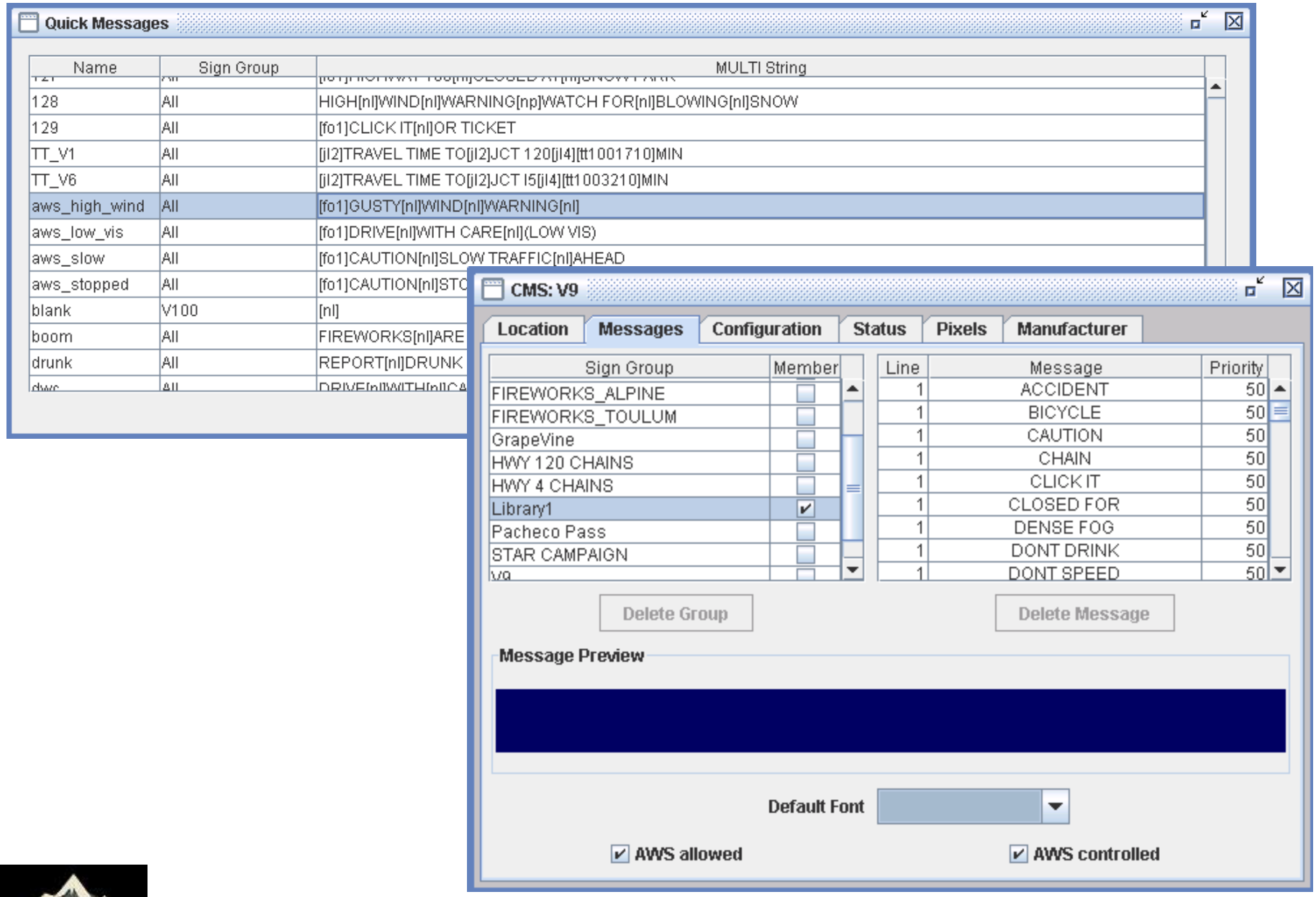
RWIS Map Icons



System Attribute Editor



IRIS Screenshots – DMS Message Libraries and DMS Definition



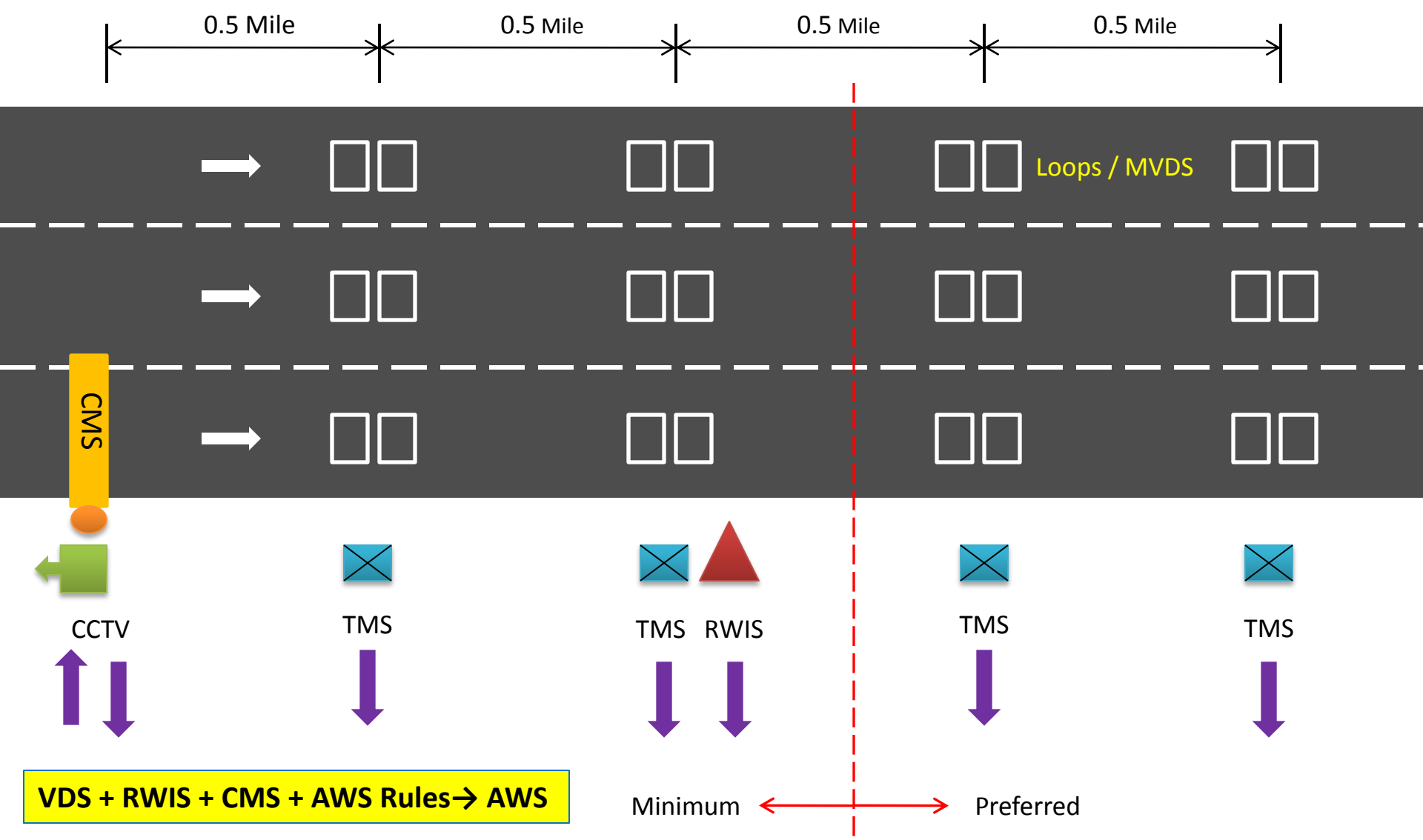
IRIS AWS Functionality

(demo)

- History – 3 steps
  1. CAWS: 1996 - 2009
  2. CAWS + IRIS: 2010
  3. IRIS AWS: 2011
     - Any agency can use this AWS functionality
- Design
  - IRIS automatically generates CMS messages
  - VDS + RWIS + DMS + AWS Rules→ AWS
- Configuration
  - Presently via editing a file, in the future via GUI
  - Associates multiple RWIS and VDS with each CMS
  - Define AWS rule parameters, e.g. max wind speed
- Verification
  - Difficult and time consuming
  - Integration test cases: +110
  - Automated unit test cases

michael@bt-systems.com

Typical Automated Warning System (AWS) Configuration



michael@bt-systems.com

What Triggers an AWS Message?

- IRIS looks for one of these 4 conditions to be present for 90 seconds at an AWS location:

1 Vehicle Speed < 11 mph

2 11 mph < Vehicle Speed < 35 mph

3 Visibility < 500 ft

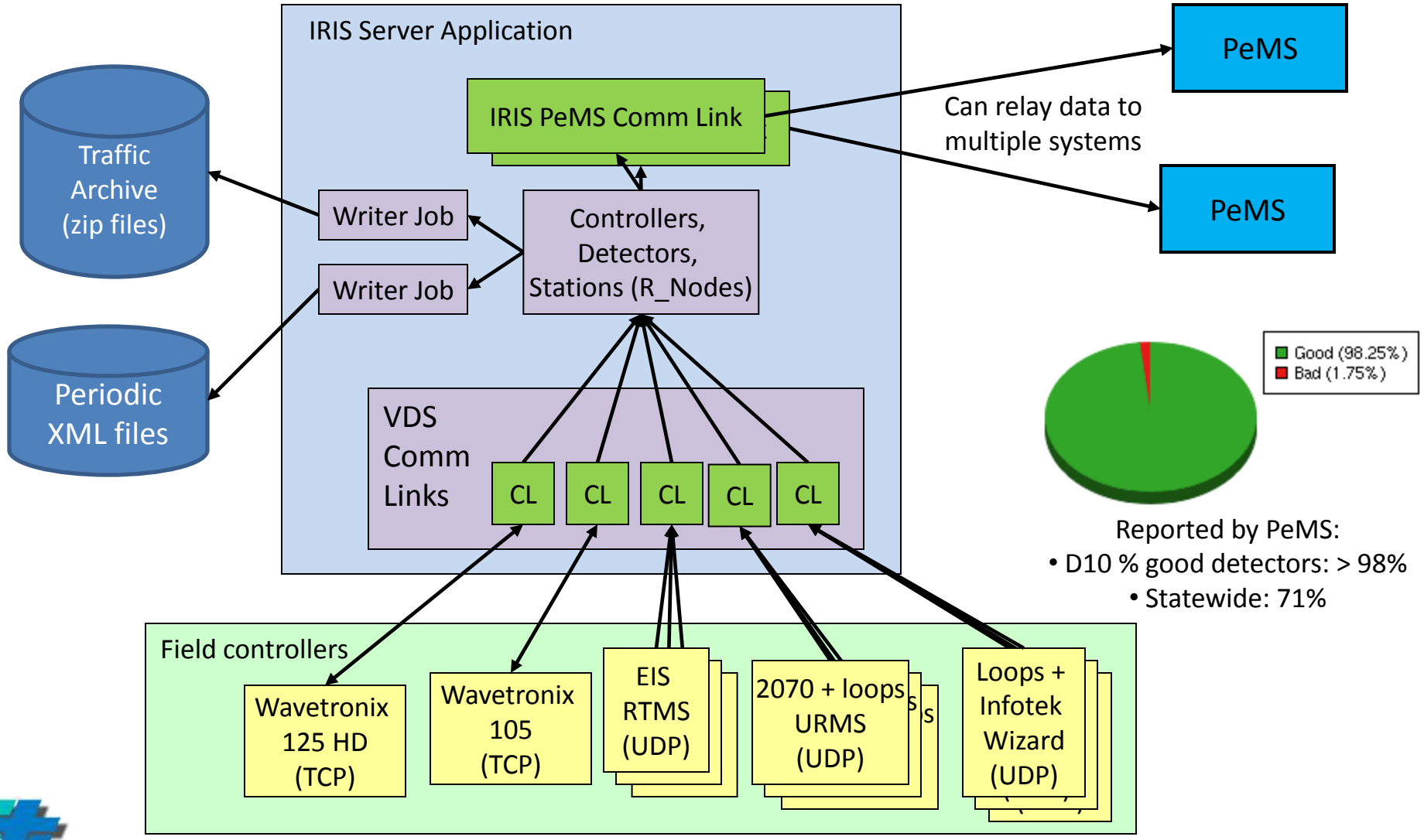
4 Wind Speed > 25 mph

Sobering Fact: In March 2011 there were 3,913 CMS messages deployed in D10
73% (2,848) of those CMS messages were automatically deployed by AWS (73%)
27% (1065) were "Custom" messages deployed by the TMC Operators

IRIS Traffic Data Collection

Open-source Data Acquisition

Field → IRIS → PeMS

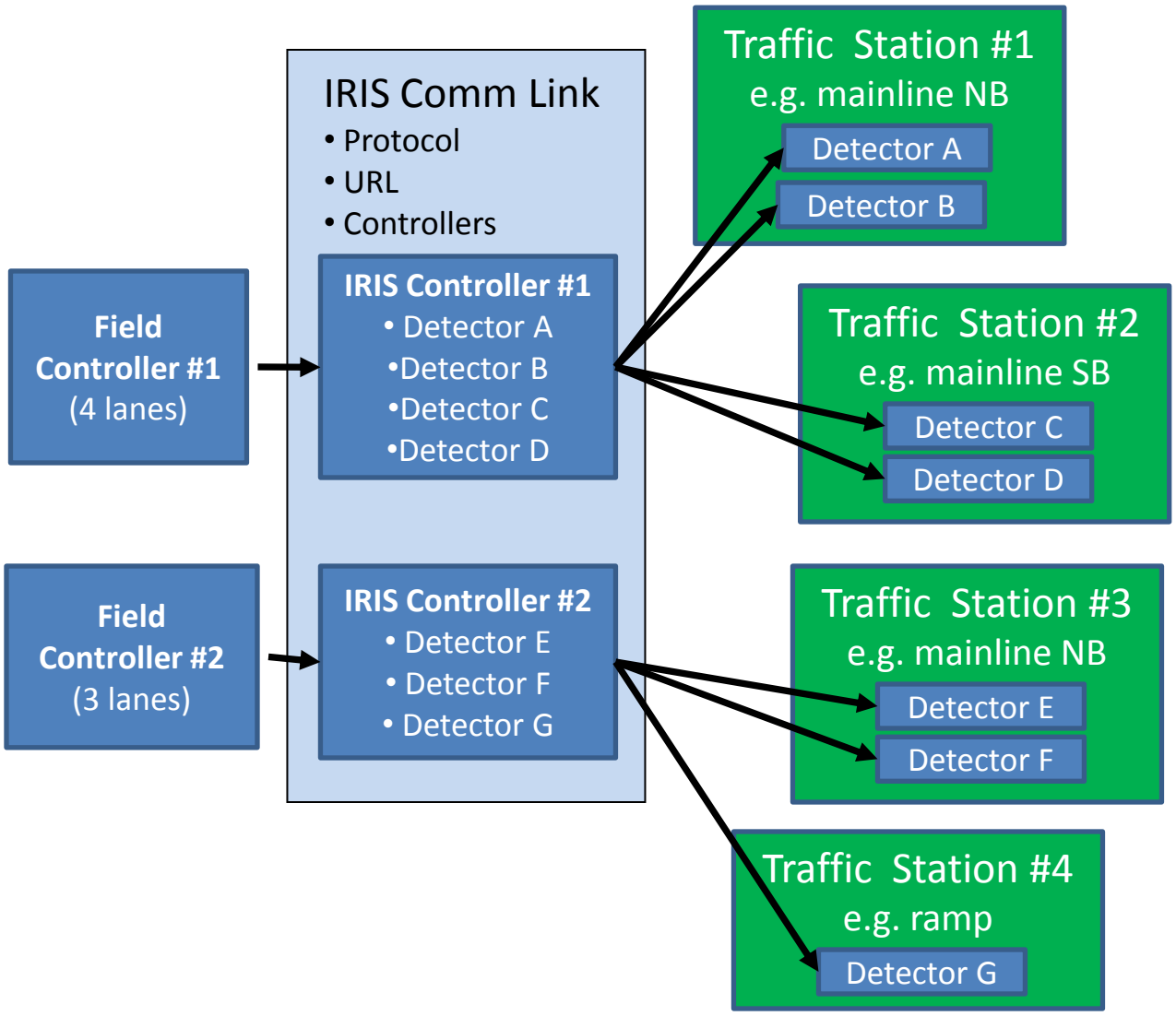


Reported by PeMS:

- D10 % good detectors: > 98%
- Statewide: 71%

michael@bt-systems.com

VDS Configuration

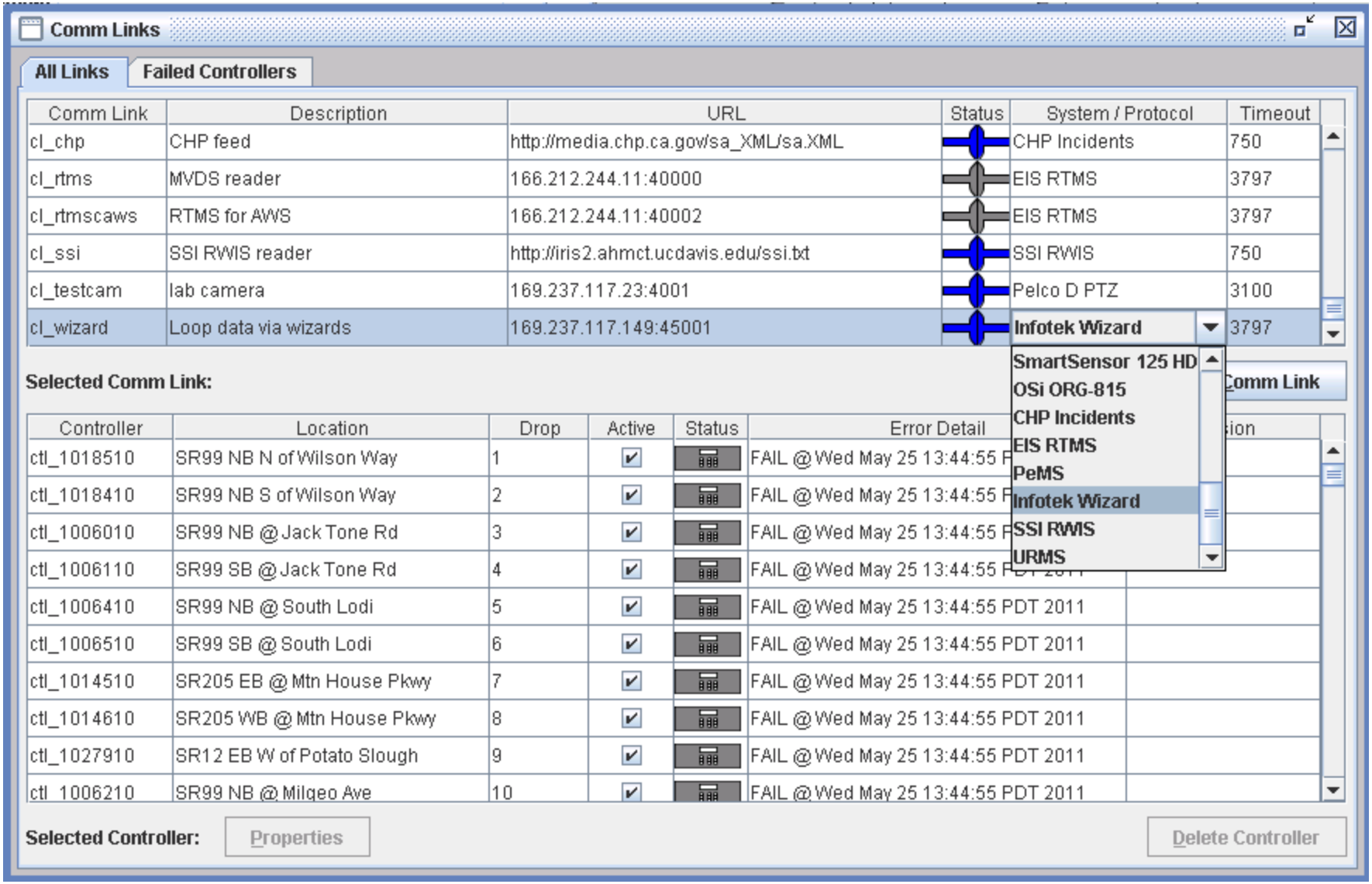


Data consumed by:
- Client maps
- Archived on server
- Real-time XML files
- PeMS
- Etc.

- Field controller lanes are logically decoupled from station lanes
- Lane order can be adjusted via specified station detector order

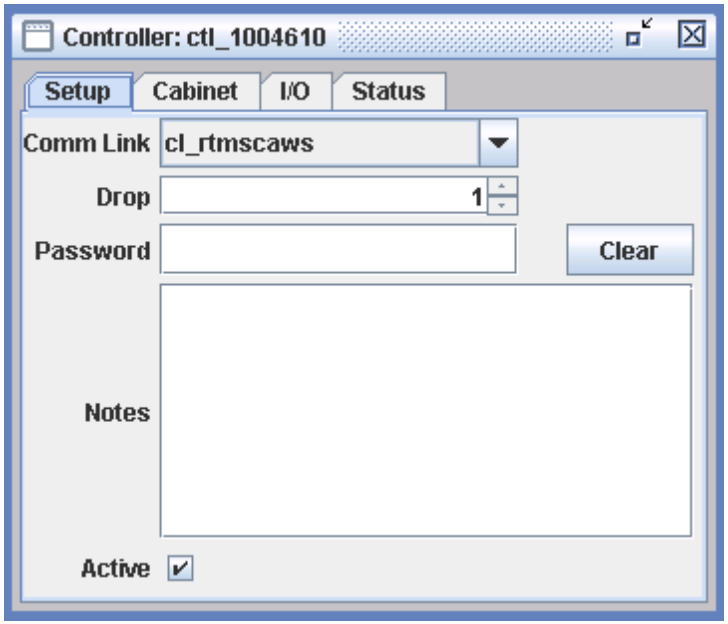
michael@bt-systems.com

IRIS Screenshot – Comm Link Configuration

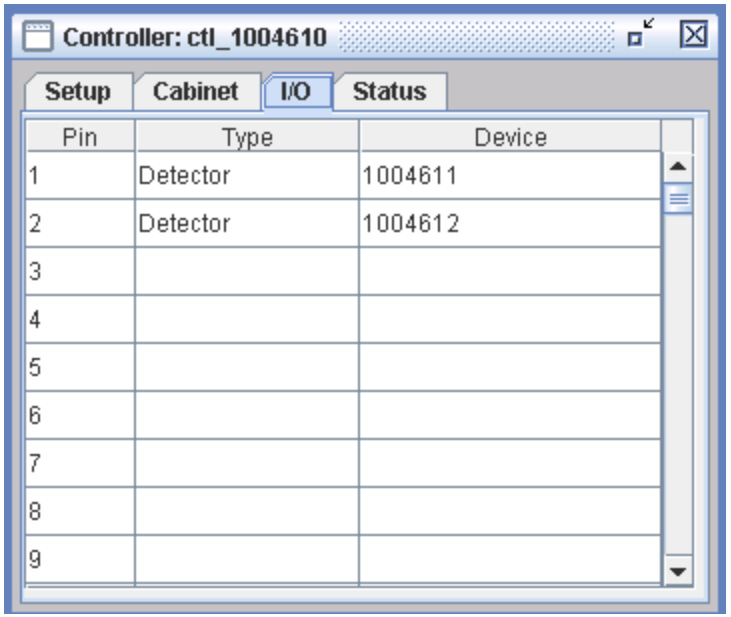


IRIS Screenshot – Controller Configuration

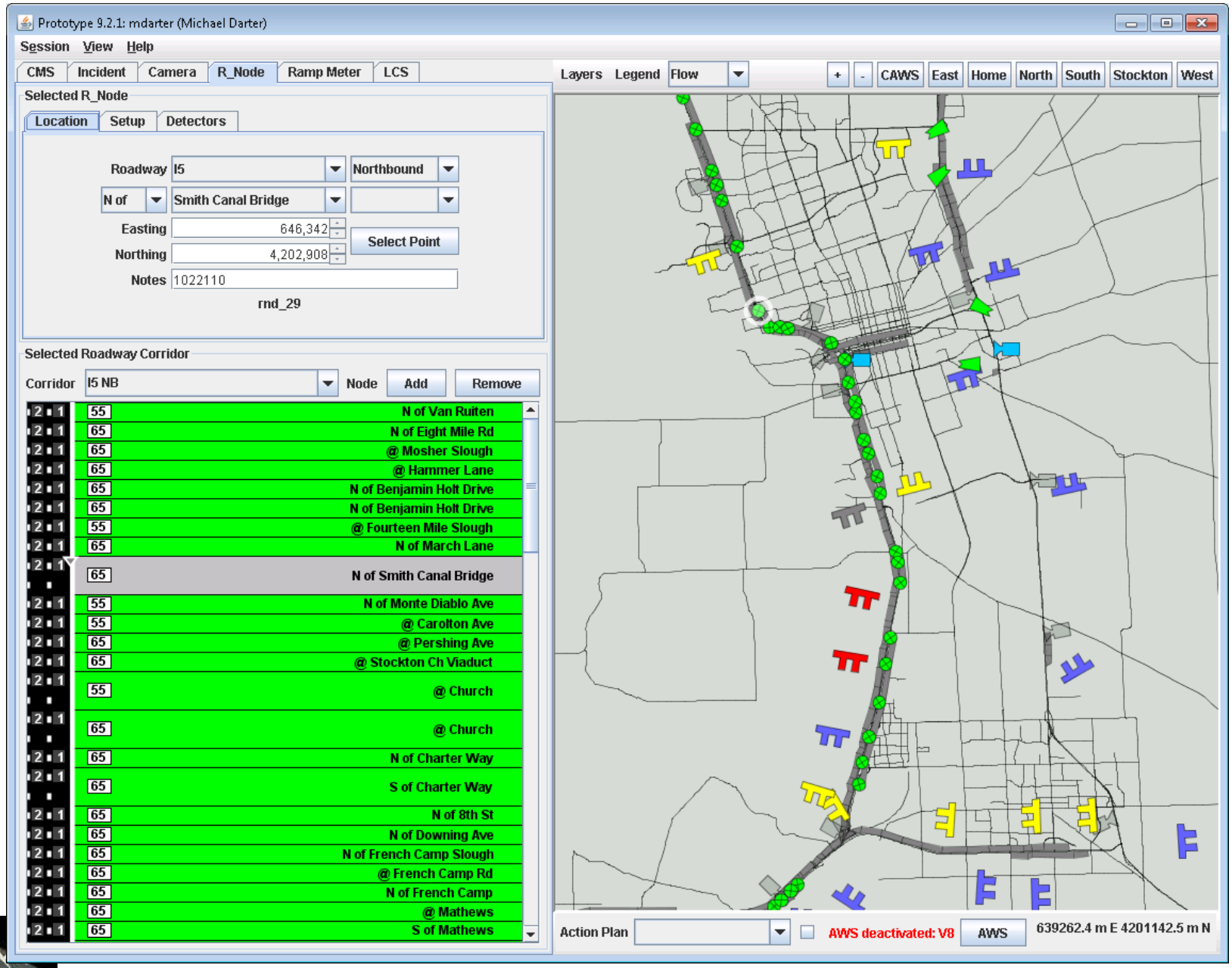
Associating inbound traffic data with a controller



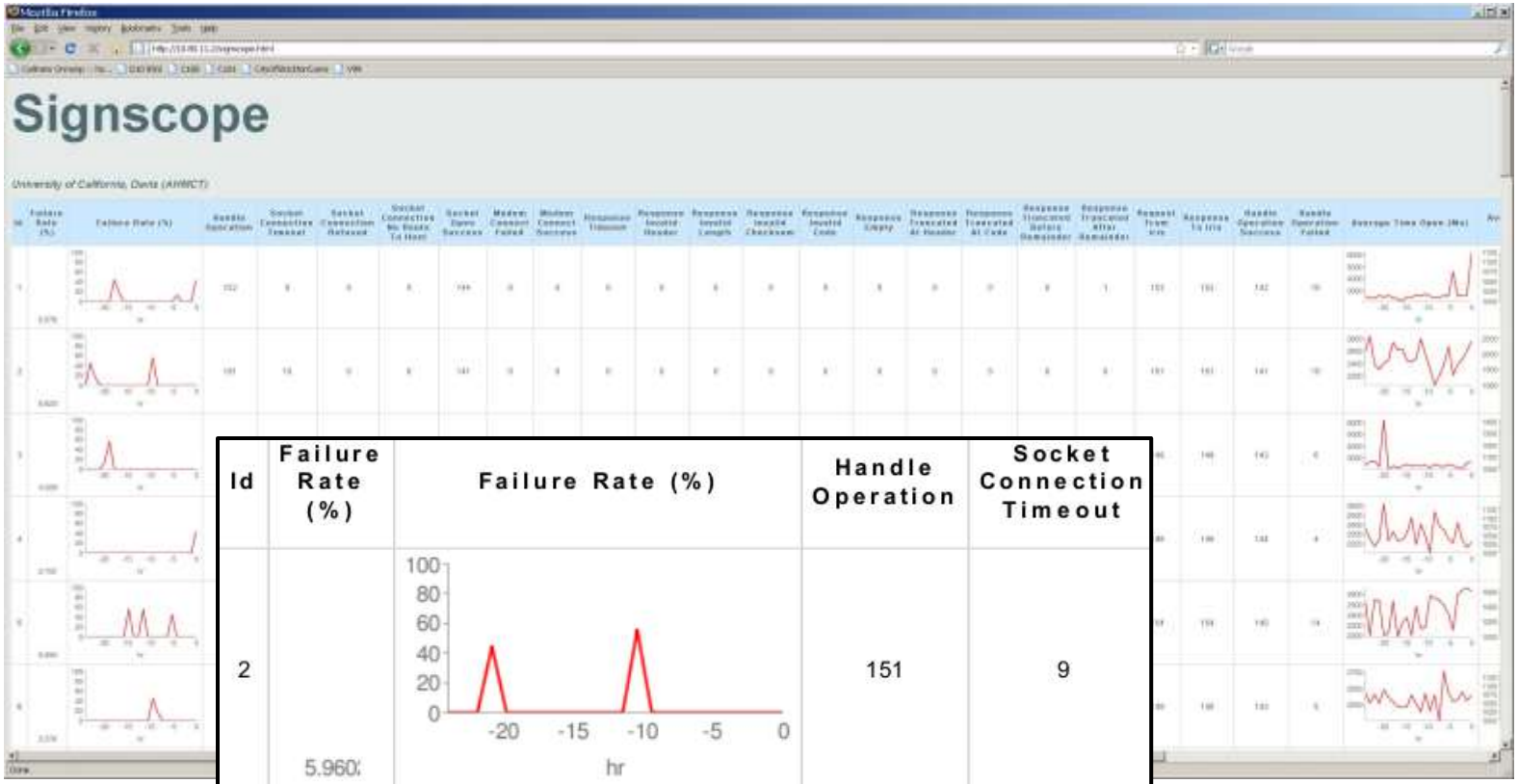
Associating detectors with a controller



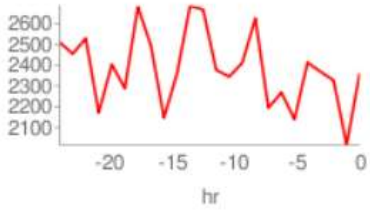
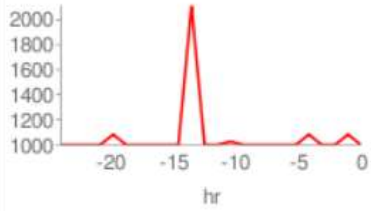
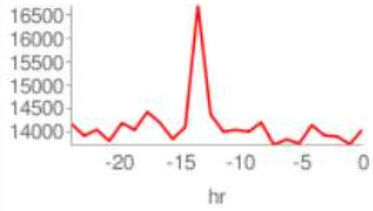
IRIS Screenshot – R_Node Definition and Editor



SignScope Screenshot – Diagnosing Comm Problems



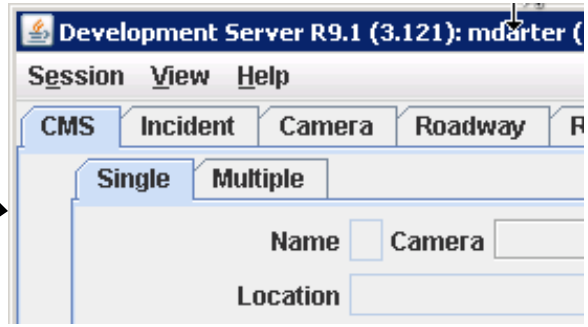
Id	Failure Rate (%)	Failure Rate (%)	Handle Operation	Socket Connection Timeout
2	5.960:		151	9

Handle Operation Success	Handle Operation Failed	Average Time Open (Ms)	Average Time Response (Ms)	Average Time Operation (Ms)	Socket Close Success	Socket Close Failed
145	0				145	0

michael@bt-systems.com

System Configuration

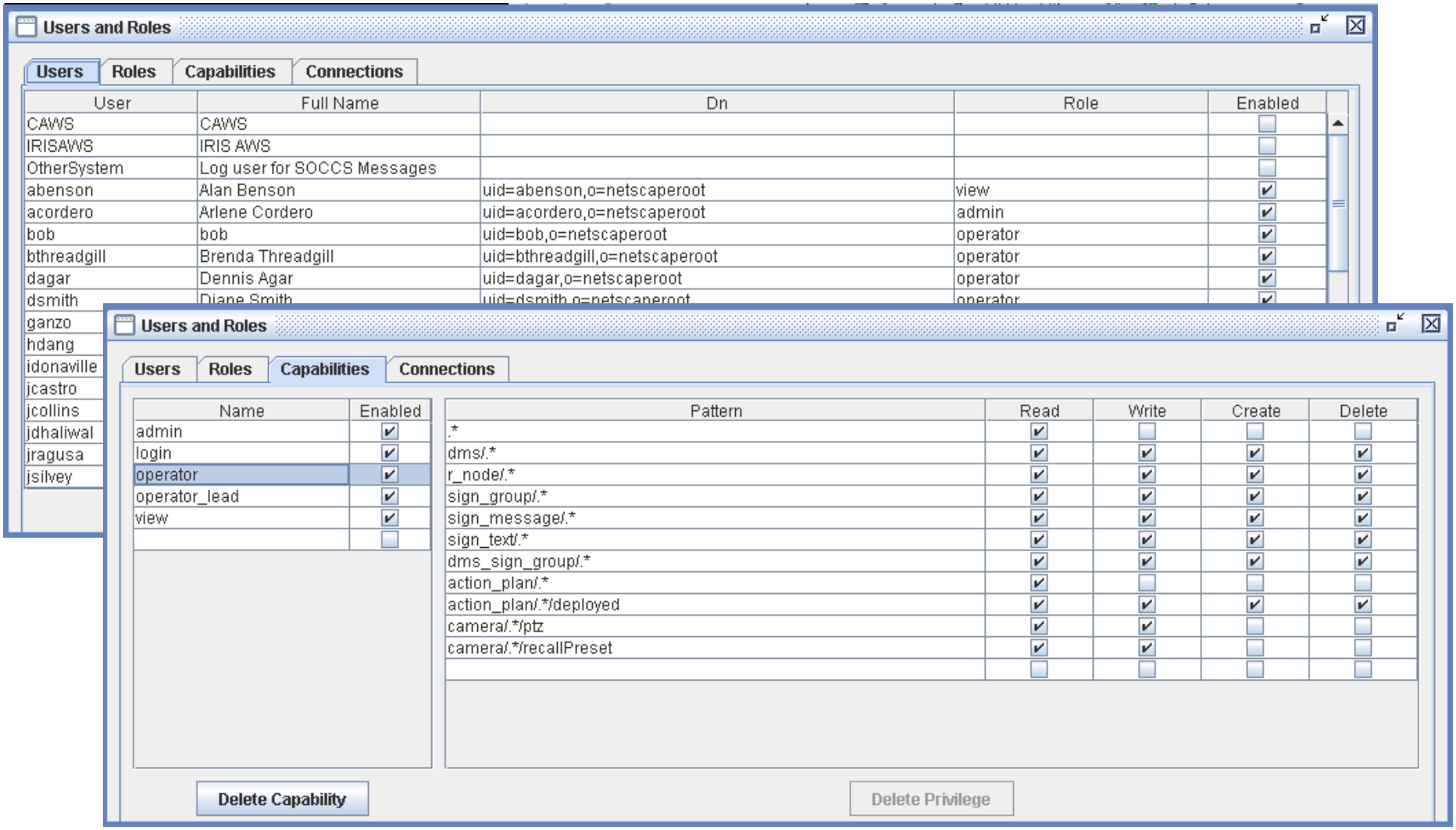
- One code base across multiple districts, agencies
- Configuration determines agency-specific behavior
  - System attributes
  - Property files
  - Agency specific code
    - Integrated
    - Non-integrated
  - System configuration stored in IRIS database
    - Comm Links, Controllers, Detectors, R_Nodes, etc.
    - System Attribute values
  - Internationalization,
    - e.g. “DMS” versus “CMS”



michael@bt-systems.com

User Permissions Screenshots

User → Role → Capabilities → read, write, create, delete permissions per object



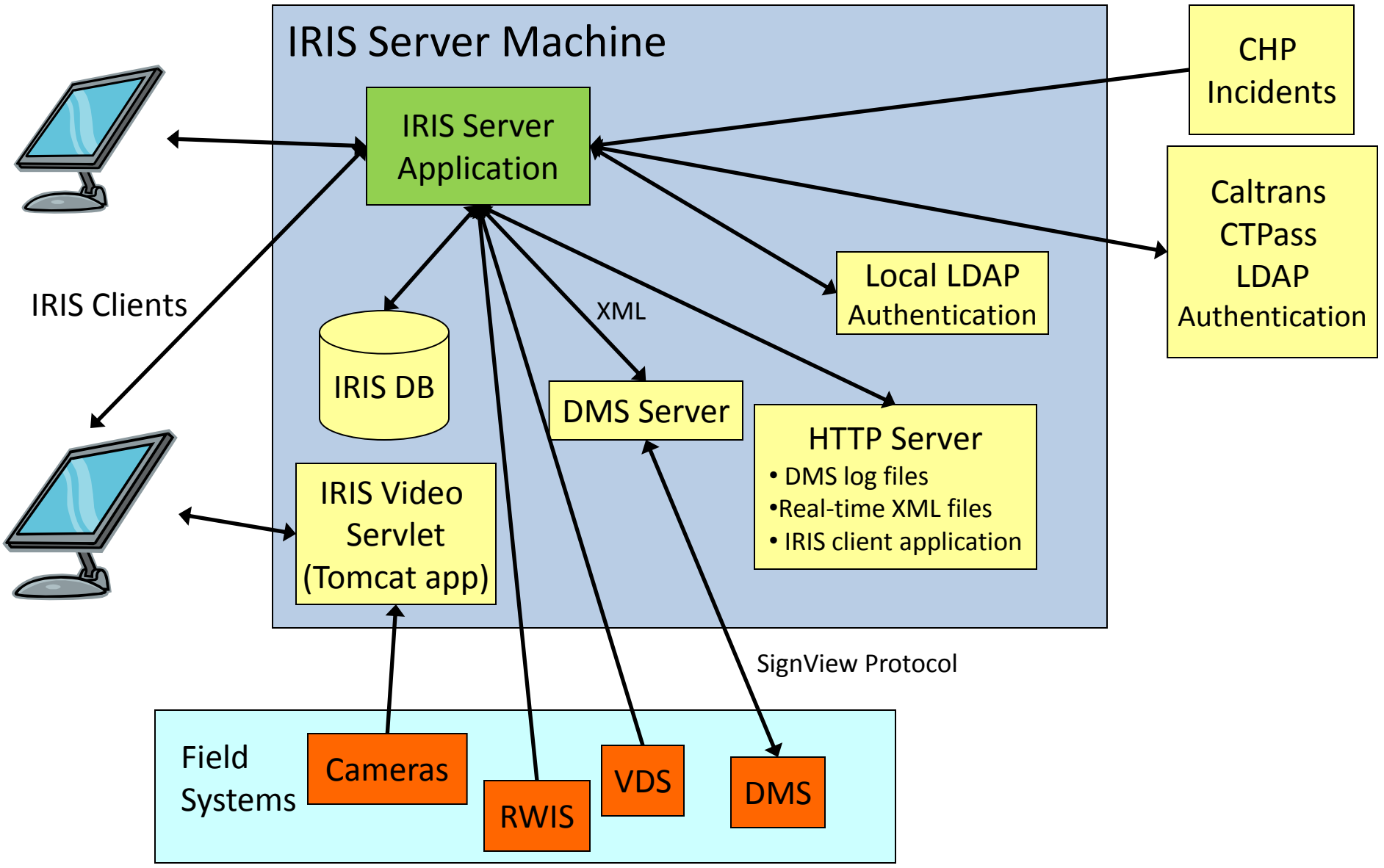
michael@bt-systems.com

IRIS Development Environment

- Client / Server architecture
- All dependent software packages are open-source
  - Free, no purchase requisition form
  - No NDA
- IRIS written in Java
  - ~100K lines of code
  - Heavily object oriented ~1000 classes
  - Learning the code base is a non-trivial
- Client
  - JWS (Java Web Start), nothing to install on client
  - Running on Windows, Linux
- Built with…
  - OpenJDK, Linux, Tomcat, Ant (all free and open source)
  - Database: PostgreSQL
  - Distributed source repository: Mercurial
- User authentication via LDAP
  - Integrates with CTPASS

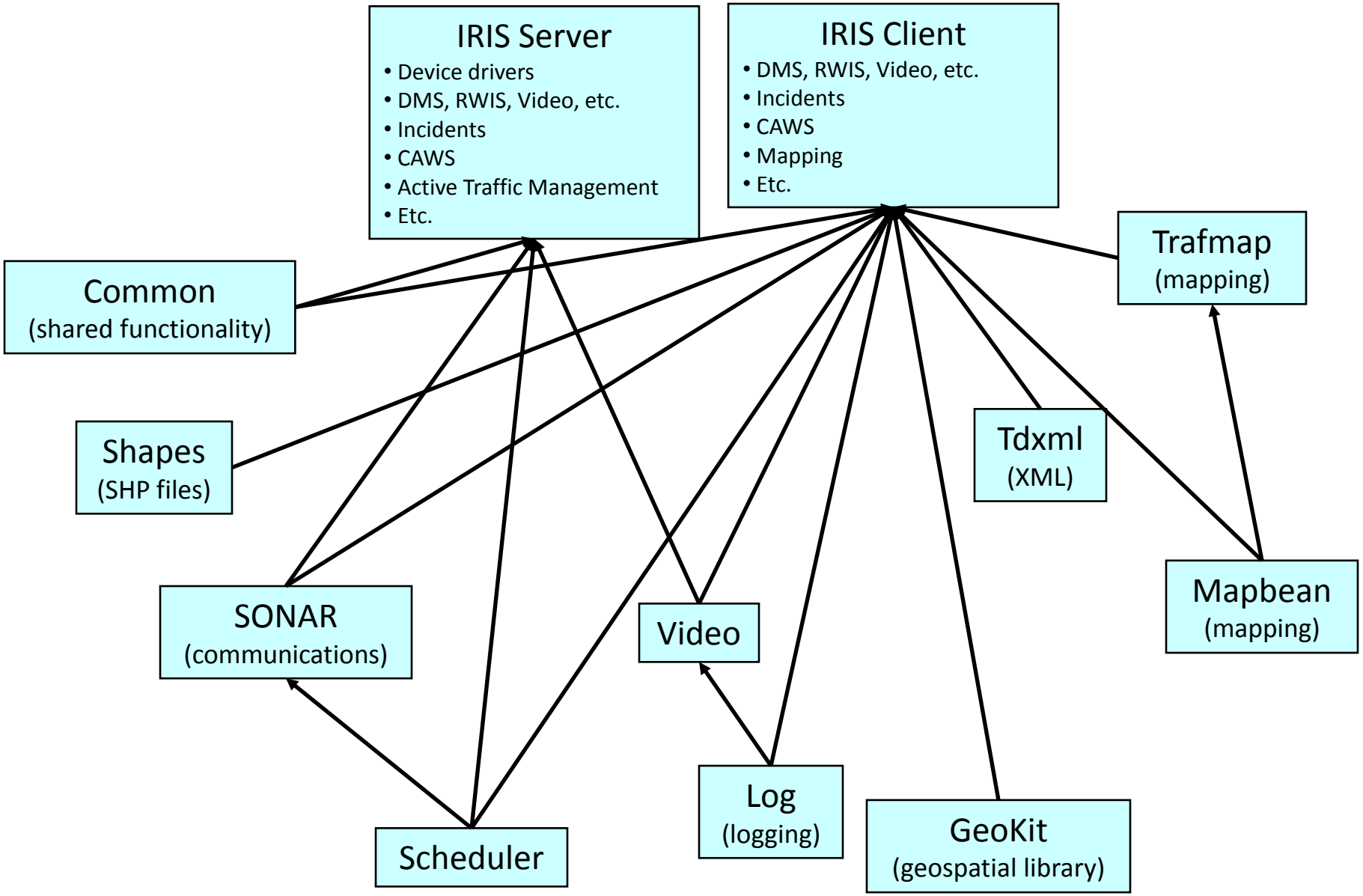
michael@bt-systems.com

Data Flow Among IRIS Applications



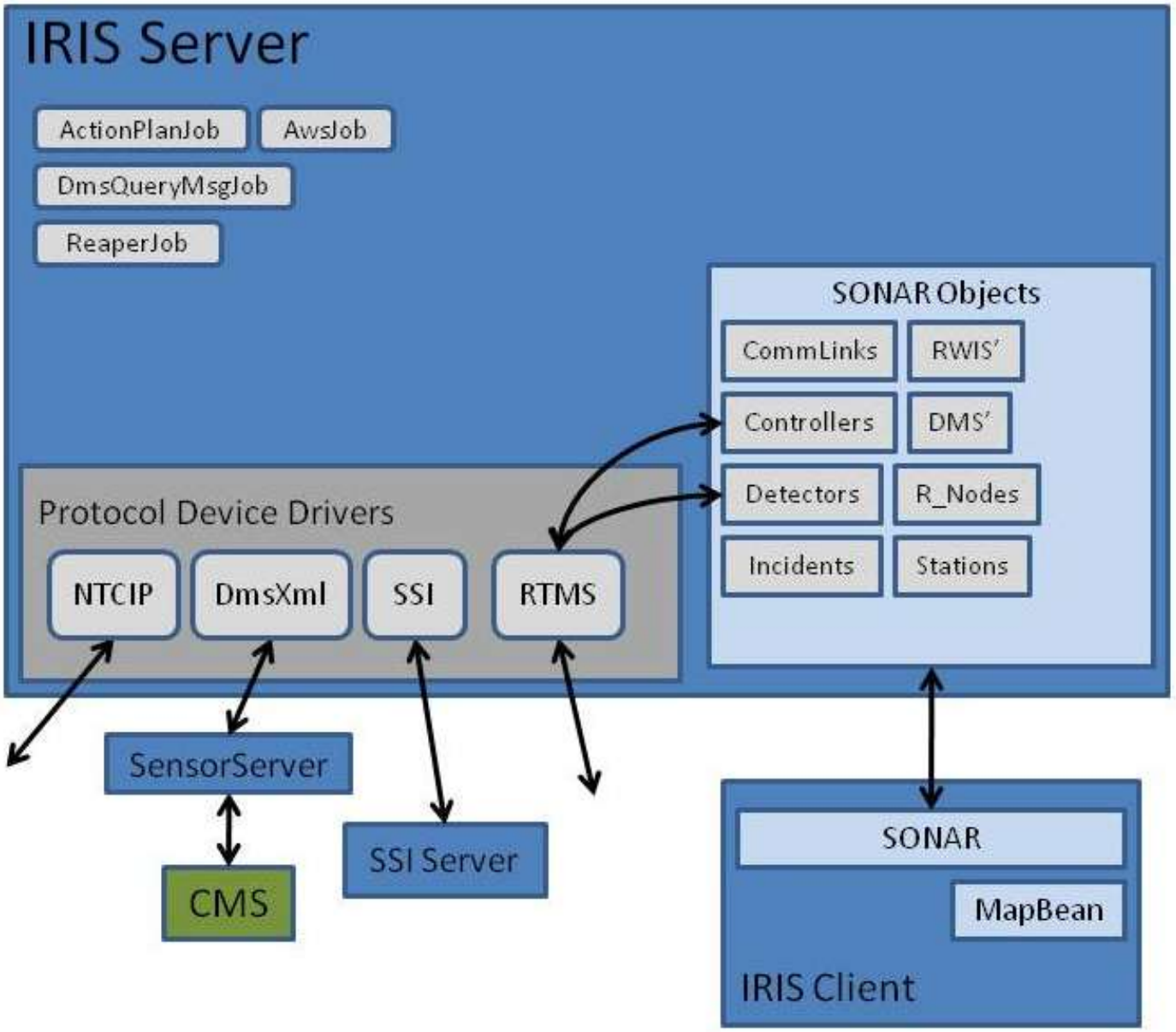
michael@bt-systems.com

IRIS Binary Modules



michael@bt-systems.com

IRIS Server Architecture



michael@bt-systems.com

IRIS Architecture

Scheduler, Jobs, Threads

Comm Links

- Are threads

Scheduler objects

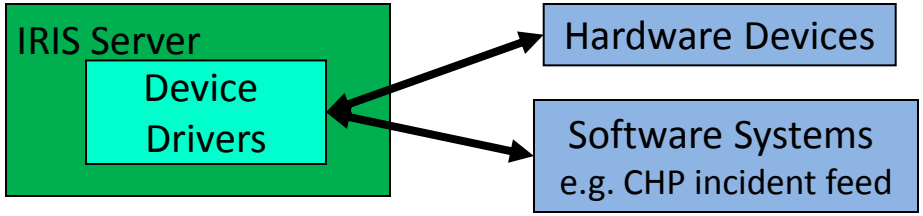
- Is a thread, executes jobs
- TIMER: runs periodic jobs
- AWS: runs AwsJob
- FLUSH: runs XML, KML jobs

IRIS has dozens of simultaneously executing jobs

- DmsQueryStatusJob: periodically queries all DMS (TIMER)
- SampleQuery30SecJob: query all pollers periodically (TIMER)
- WeatherQueryJob: queries all weather pollers every 60 seconds (TIMER)
- SendSettingsJob: send settings to devices 4AM (RampMeters, etc.) (TIMER)
- ReaperJob: garbage collector for SONAR objects (TIMER)
- AwsJob: makes AWS decisions every 20 seconds and sends AWS messages to signs (AWS)

michael@bt-systems.com

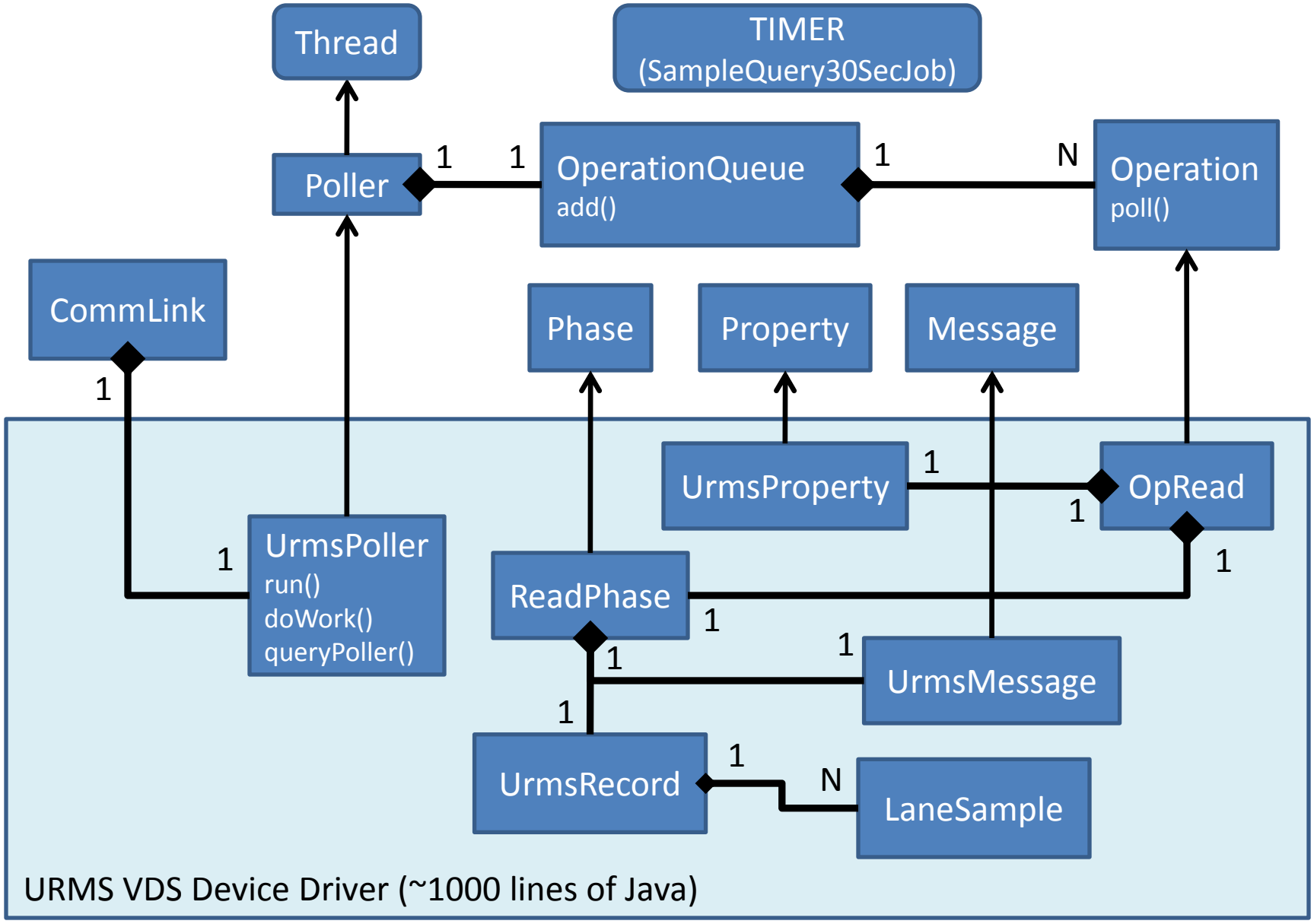
IRIS Architecture + Design
Protocol Device Drivers



- A key ATMS function is to interface with external systems
- IRIS has a powerful device driver interface
- For reading and/or writing to external systems
- Drivers are open-source
- Long-term goal: as many device drivers as possible
- Existing drivers
  - VDS: Wavetronix 105, 125, EIS RTMS, URMS, Wizard, Canoga, others pending
  - DMS: NTCIP A, B, C, SignView
  - Video: Pelco D PTZ, Pelco switcher, Vicon PTZ, Vicon switcher, Manchester PTZ
  - RWIS: Optical Scientific ORG-815, SSI
  - CHP incident feed
  - PeMS
  - External AWS, e.g. CAWS
  - Etc.

michael@bt-systems.com

IRIS Class Structure for URMS Device Driver



michael@bt-systems.com

Software Engineering / Testing

Motivation for doing it right…

- Across industries, 32% of software projects are successful (Standish)
  - On time
  - Within budget
  - With desired features
- 50% of projects are rolled back out of production (Gartner)
- Annual software project cost overruns are $50 - $80 billion
- Software development it risky
- Question: why not write IRIS from scratch?

Open Source

What 'open source' means…

1. Legal definition, `http://opensource.org`
2. Development method

Open-source (GPL) requires…

- Source code is freely available
- Modified code is open-source

Results in…

- Creation of knowledge communities
- Cooperative development model
- Network effect for enhancements / defect repairs
- Reduces legal complexity, no NDAs

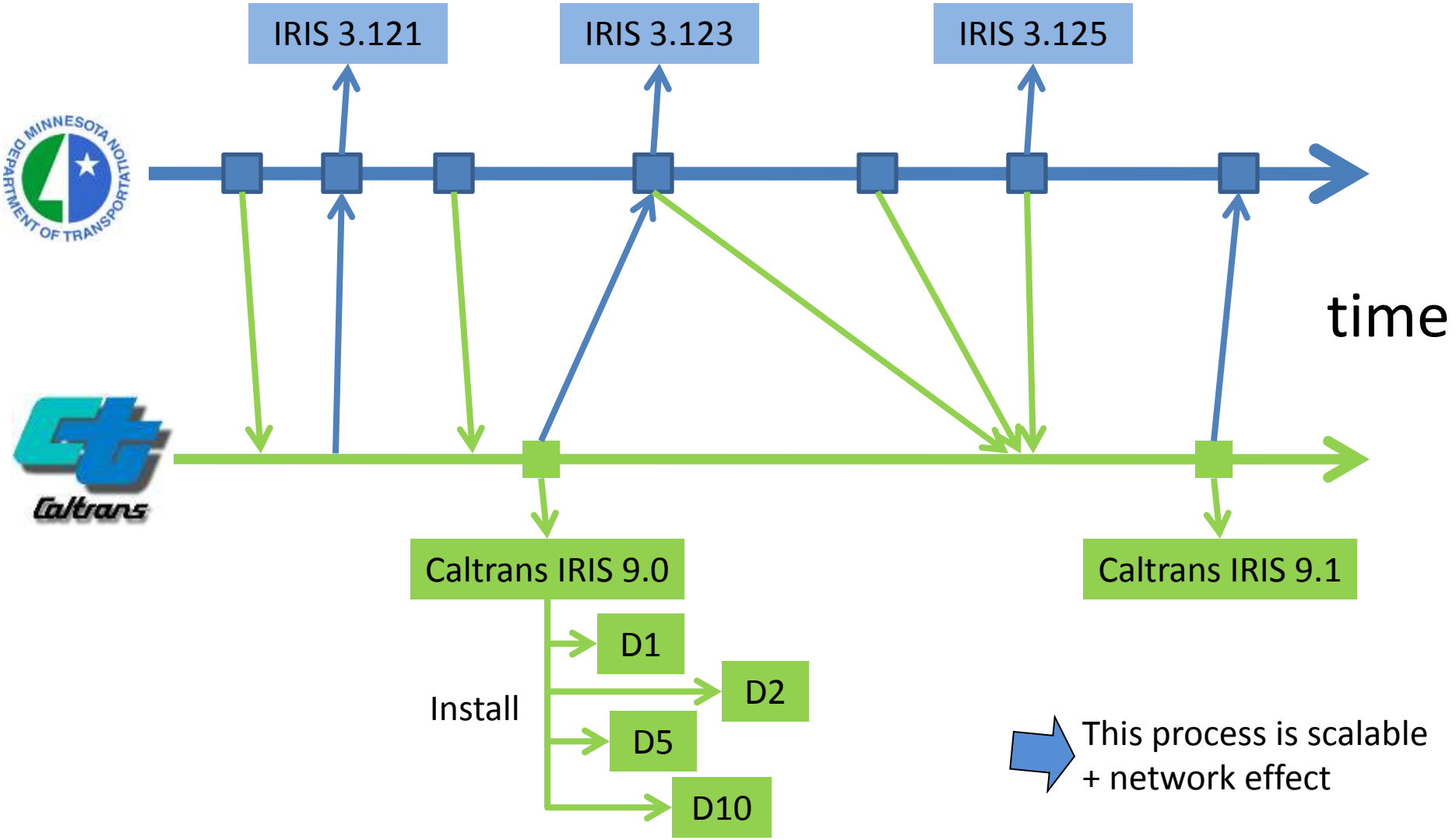
michael@bt-systems.com

Integrating with Proprietary Systems

1. Proprietary protocol (e.g. SignView)
   - If GPL 'distribution' criteria not triggered, write driver integrated with IRIS, but don't distribute source.
   - Write stand-alone server, e.g. SensorServer
     - Can use any license you want
     - Write and publish a public interface, e.g. DmsXml
2. Proprietary system with public protocol
   - E.g. almost any hardware device
   - Add new protocol to IRIS
3. Proprietary system with proprietary protocol
   - Same as case #1

michael@bt-systems.com

IRIS Release and Collaboration Process



michael@bt-systems.com

Testing

- Continuous Integration →
  - Test early and often
  - Automated test cases (600+)
    - Integrated w/ development process
    - Enables Test Driven Development (TDD)
  - DMS Simulator (CASPER)
- Validation
  - User needs → requirements → design → prototype
  - Users experiment with prototype
  - User acceptance test cases
- Verification
  - Automated test cases
  - Integration testing, test cases + ad hoc
  - Multi-agency testing
  - Acceptance test cases
- Future
  - Automated integration tests
  - Simulated traffic
  - Capture and replay of traffic and RWIS data

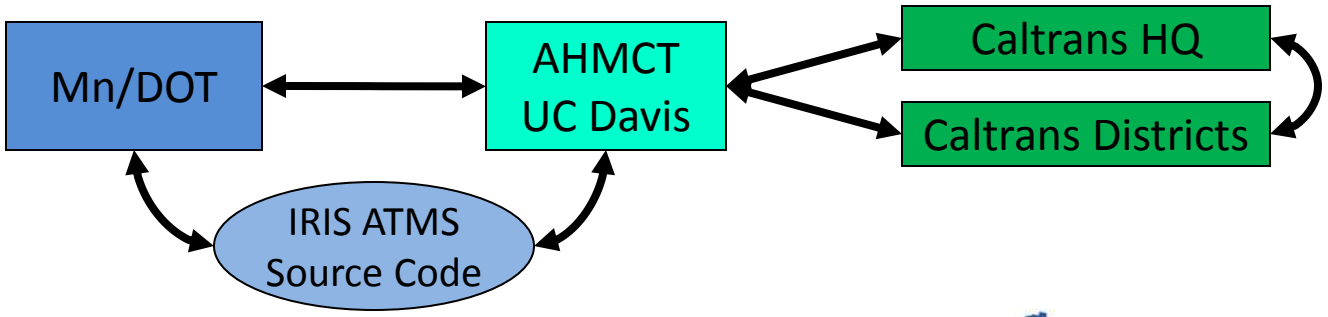
michael@bt-systems.com

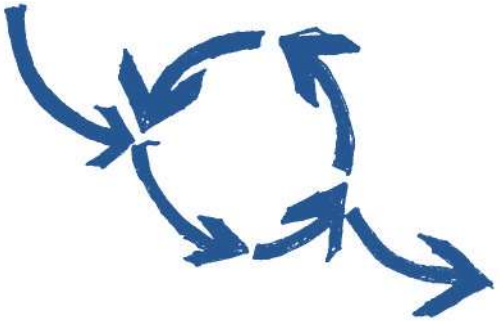
Example Automated Test Case MULTIString junit test cases

```
void equals() {
        MultiString t1 = new MultiString("x");
        MultiString t2 = new MultiString("x");
        MultiString t3 = new MultiString("x");
        // reflexive
        assertTrue(t1.equals(t1));
        // symmetric
        assertTrue(t1.equals(t2) && t2.equals(t1));
        // transitive
        assertTrue(t1.equals(t2) && t2.equals(t3) && t1.equals(t3));
        // simple cases
        assertTrue(new MultiString("").equals(new MultiString("")));
        assertTrue(new MultiString("").equals(""));
        assertTrue(new MultiString("XXX").equals("XXX"));
        assertTrue(new MultiString("XXX").equals(new MultiString("XXX")));
        assertFalse(new MultiString("XXX").equals("XXY"));
        assertFalse(new MultiString("XXX").equals(new MultiString("XXY")));
        // verify normalization used
        assertTrue(new MultiString("[fo1]abc").equals("[fo1]ABC"));
}
```


michael@bt-systems.com

Adding an IRIS Enhancement



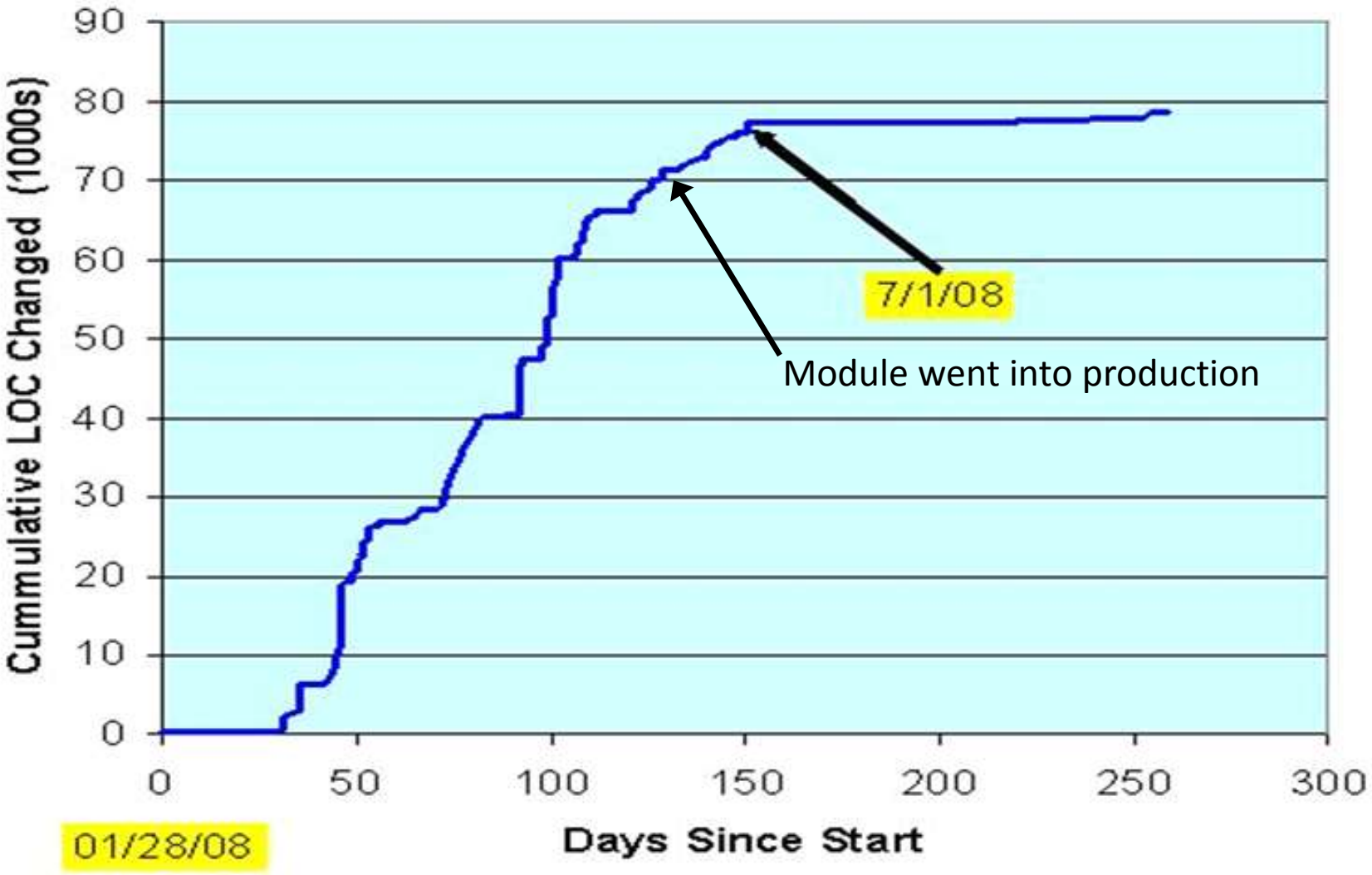


1. Request for improvement…
   - Caltrans operators “we need…”
   - Organizational need
   - Defect discovered
2. Ticket created
3. Requirement written + discussion
4. Design, discussion with Mn/DOT
5. AHMCT implements, discusses as necessary
   - Automated test cases
   - Integration testing
6. AHMCT publish new feature in public IRIS repository
7. Mn/DOT…
   A. Read new change sets
   B. Review
   C. Merge into Mn/DOT IRIS repository
8. Mn/DOT publishes a new version of IRIS w/ new feature

michael@bt-systems.com

Development Process Transparency

Cumulative SLOC vs. Time for a Module



michael@bt-systems.com

Estimating the Dollar Value of IRIS

What is the cost to develop IRIS from scratch?

Module	Lines of Code (filtered)	Est. Years to Develop	Est. Cost to Develop
IRIS	83,792	20.9	$3,797,360
Caltrans IRIS apps	11,410	2.4	$441,940
Total	**95,202**	**23.3**	**$4,239,300**

The total dollar value of the Caltrans IRIS project.

SLOCCount uses COCOMO (COnstructive Cost Model)

michael@bt-systems.com

High-level Cost Comparison
(IRIS compared with existing proprietary ATMS)

Goals of comparison

- Quantify “low cost”
- Approximate life-cycle cost comparison between IRIS and the existing (proprietary) system
- See the final report for details, assumptions, etc.

Assumptions

- Actual costs are used, except in 1 place
- Apples versus apples?
  - Each system has different features
    - IRIS has data acquisition, active traffic management, lane management, AWS, tiled mapping, etc.
    - The existing ATMS system has adaptive ramp metering, incident detection, & response
  - The comparison is for core features (DMS control, mapping)
- Customization costs are not expected to differ between approaches
- Caltrans funded one-time generalization features, e.g. SA, I18N

michael@bt-systems.com

Five Year Cost Comparison

(for one Caltrans district)

Cost Component	Open ATMS	Existing ATMS
1. Acquisition	$17,800	$1,054,500
2. Customization & configuration	$350,000	$350,000
3. Year 1 maintenance	$150,200	$540,000
Year 2 maintenance	$150,200	$540,000
Year 3 maintenance	$150,200	$540,000
Year 4 maintenance	$150,200	$540,000
Year 5 maintenance	$150,200	$540,000
Total 5 Year Cost	**$1,118,800**	**$4,104,500**
Budget spent on development (%)	**98%**	**27%**

72% cost reduction +

More development hours per dollar spent

michael@bt-systems.com

1. Acquisition ATMS Costs

(for one Caltrans district)

Component	IRIS Cost	Existing ATMS Cost
Server hardware	$4,500	$150,000
Backup server	$4,500	$150,000
5 client workstations (optional)	5x $1,500	5x $1,500
OS support, 24x7 (optional)	$1,300	Included
Server software licenses	$0	$239,000
Backup server licenses	$0	$239,000
Developer software licenses	$0	$269,000
Total Acquisition Cost	**$17,800**	**$1,054,500**

This is a 98% cost reduction compared with the existing ATMS system

michael@bt-systems.com

2. Customization & Configuration Costs

Four data points…

Traffic Management Installations	Cost
D1, rural, IRIS installation, DMS functionality (integration w/ video, RWIS is extra)	2 weeks
D5, rural, IRIS installation, DMS functionality (integration w/ video, VDS is extra)	2 weeks
D10, urban/rural, IRIS install + customization (Phase 1 Caltrans IRIS UC Davis research project)	$350,000
D4, urban, existing ATMS installation + customization	$750,000

Customization and configuration costs are not expected to differ between approaches.

michael@bt-systems.com

3. Annual Maintenance Costs

(for one Caltrans district)

Annual Cost Component	IRIS Cost	Existing ATMS Cost
Server hardware maintenance, 24x7 response	$200	$30,000
Software license cost	$0	$60,000
Maintenance cost, personnel*	½ - 1 FTE	2.5 - 3 FTE
Total Annual Maintenance Cost	**$75,000 - $150,200**	**$465,000 - $540,000**

This is a 68% - 86% cost reduction

*Based on assumed $150,000 per year contract price per FTE

michael@bt-systems.com

Kin Yen
Development Engineer • AHMCT University of California

- District status update
- Video
- System maintenance
- Mapping
- Scalability
- Current and future enhancements

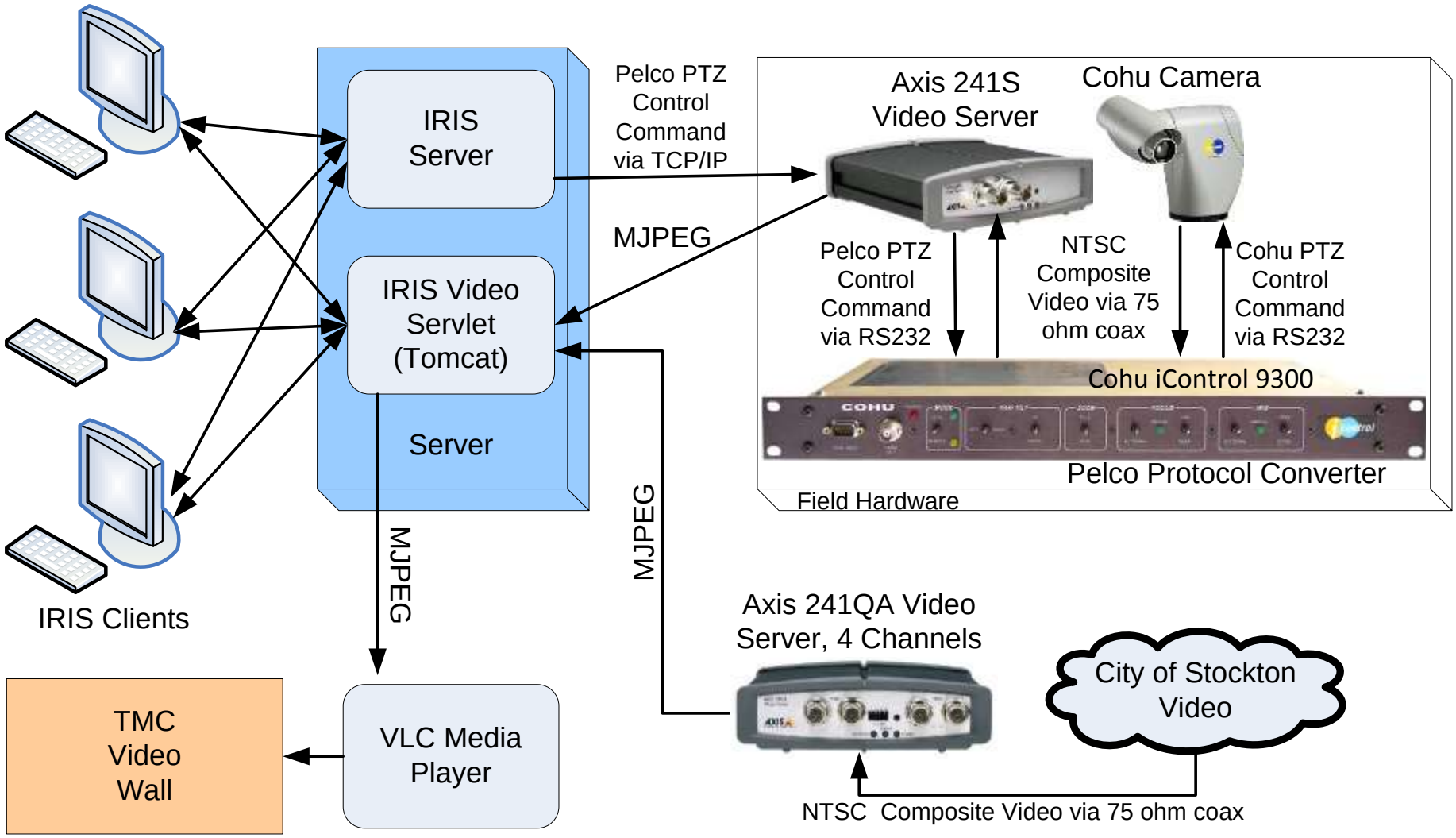
ksyen@ucdavis.edu

Status of IRIS in Caltrans Districts

- District 10
  - Installed 05/2008
  - Used as main ATMS since 05/2009
  - Using DMS, video, incidents, VDS, RWIS, AWS
  - 02/2011
    - IRIS AWS active
    - Reading all VDS
    - Forwarding traffic to PeMS
    - Forwarding loop data from 2070s (URMS)
- District 5
  - Installed 08/2009
  - At present, light use of DMS
  - Experimented with traffic, video, scheduling
  - Full deployment fall 2011
- District 2
  - Installed 12/2010
  - Not used (waiting for improved mapping)
  - Full deployment fall 2011
- District 1
  - Installed 09/2009
  - Light use of DMS
  - Full deployment fall 2011

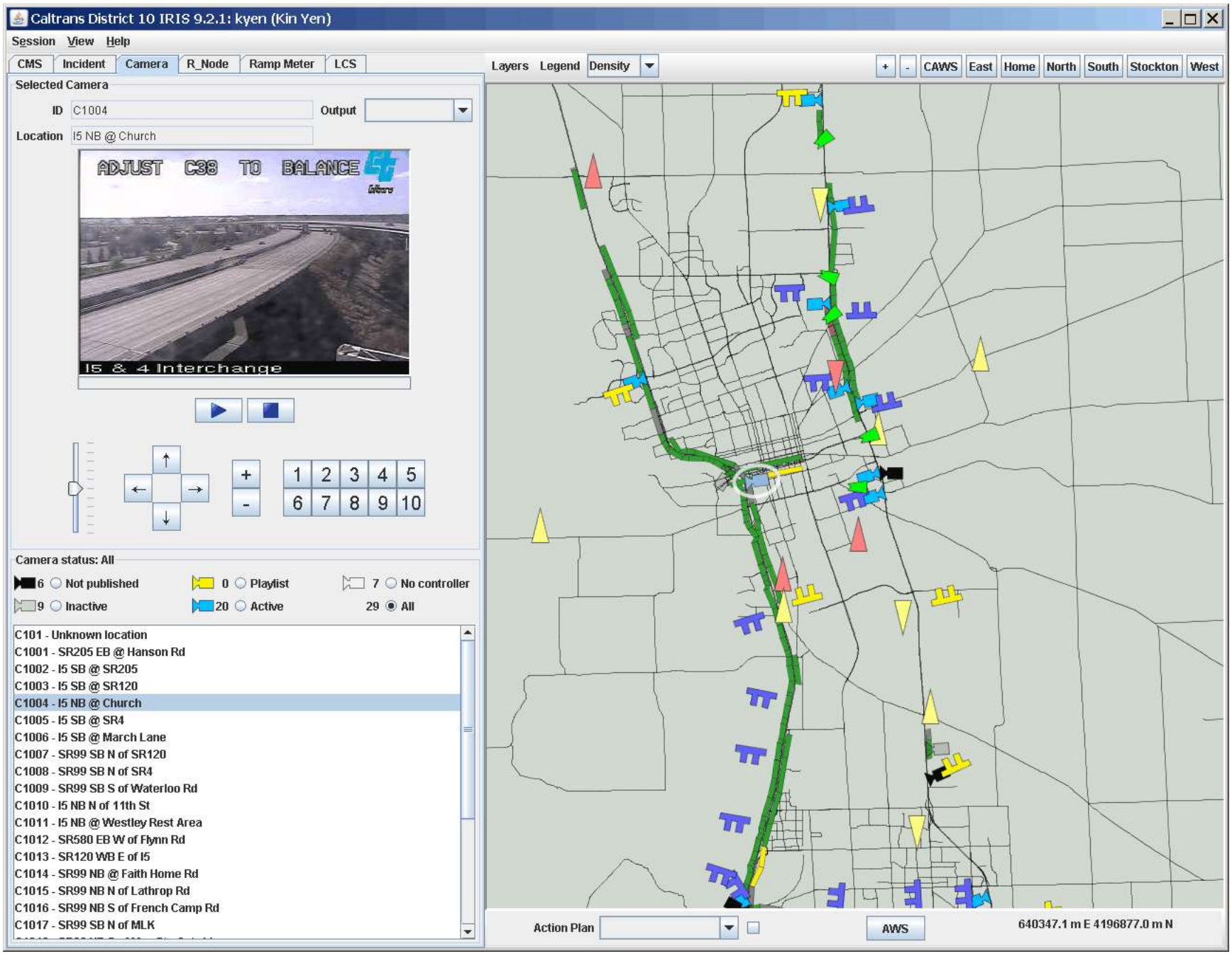
ksyen@ucdavis.edu

D10 Video Architecture



ksyen@ucdavis.edu

Video Screenshot



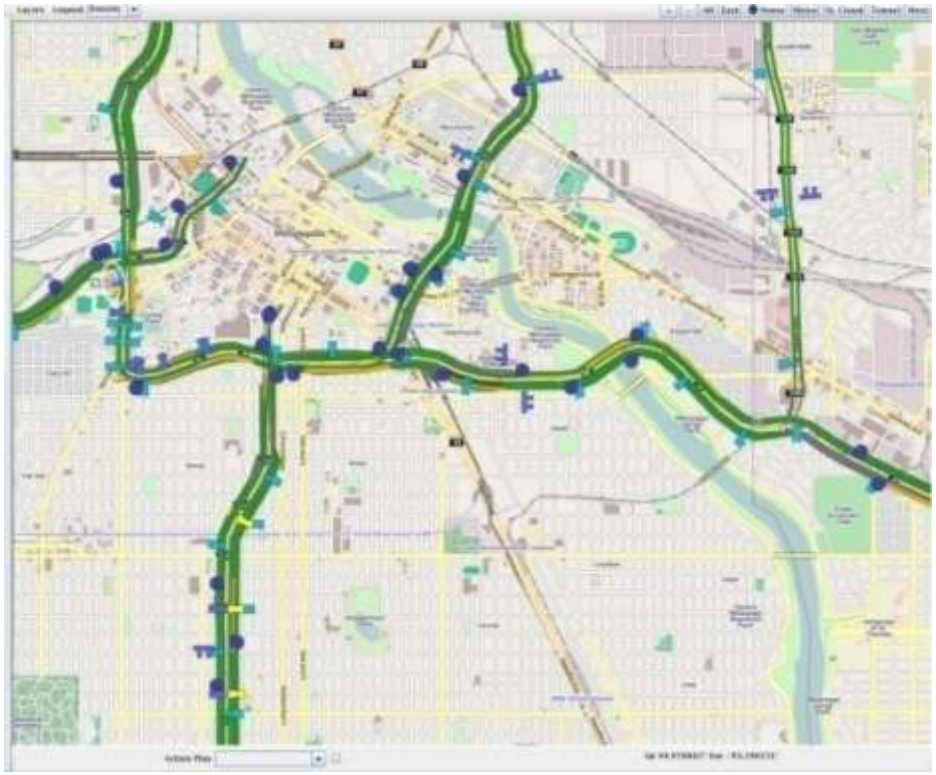
System Maintenance

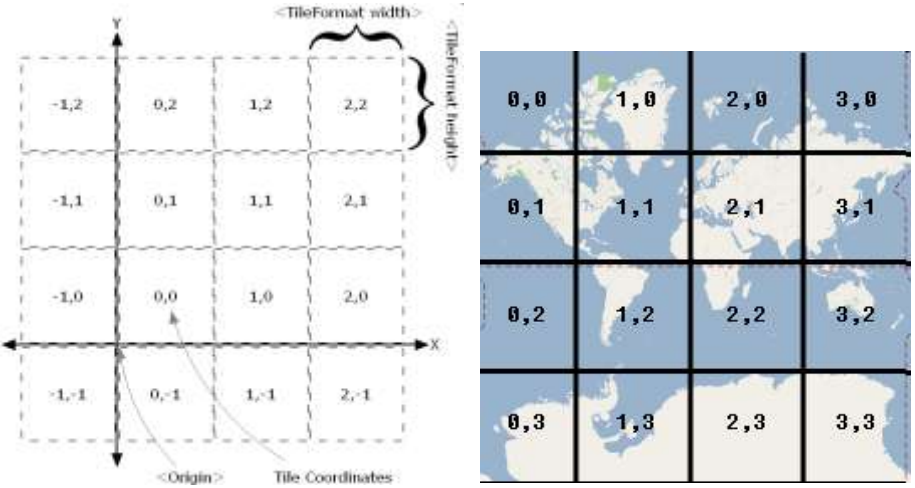
- There is very little IRIS system maintenance
  - Update IRIS configuration if field elements change
    - E.g. VDS, CMS, cameras
  - Delete contents of some log database tables
    - Comm_events, detector_events
- Normal server maintenance
  - Check disk space (clear log files if necessary)
  - Update operating system packages
  - Backup
    - IRIS PostgreSQL database
    - LDAP database
    - IRIS configuration files

ksyen@ucdavis.edu

IRIS OpenStreetMap (OSM) Mapping

- Tile format
  - Spherical Mercator
  - 256x256 png
  - Zoom level
  - Folder and file name
  - Zoom\X\Y.png
- Pro: Fast
- Con: Large data size (GB)
- Limit area, zoom level, skip zoom level





OSM Tile Coordinate vs Google Tile Coordinate

- Less license Restriction
- Many open-source software support (viewer engine, tile server, tile maker)
- Custom look with template
- Support aerial photo / raster file

Zoom Level

Zoom level 16

Zoom level 17

Zoom level 18

Zoom level 19

Zoom level 20

Zoom level 21

Scalability

- Scalability testing for simultaneous users has been performed.
- Scalability of adding field elements has **not** been tested.

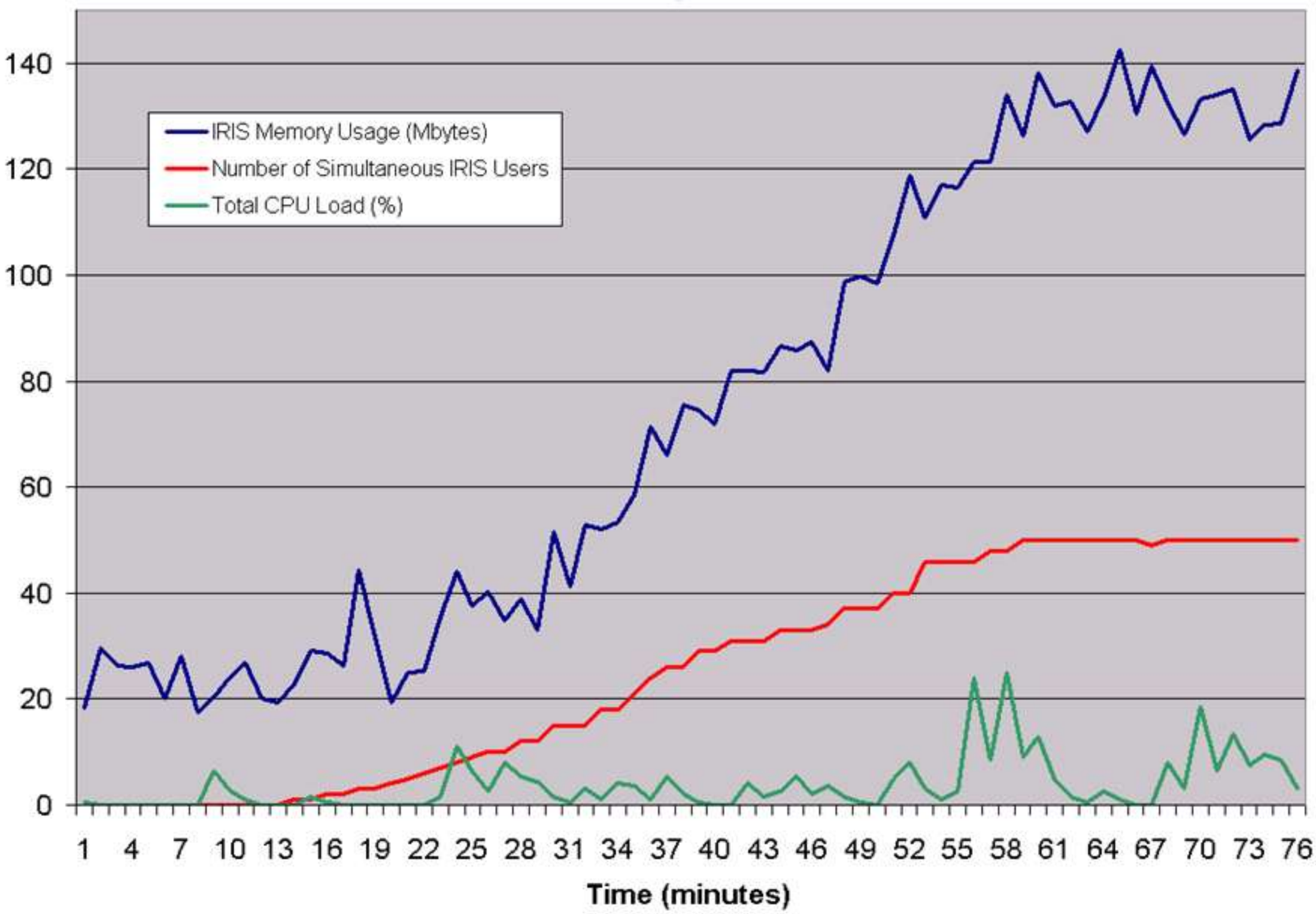
On a $1000 dual-core server…

- Running 50 simultaneous IRIS clients
- With 60 DMS, 250 VDS
- Running IRIS, database, all other apps

Resulted in…

- Processor load was < 20% on 1 core
- IRIS memory usage was 140 MB
- Total server memory usage was 818 MB

IRIS Scalability



Ongoing + Future Enhancements

- Enhanced IRIS PeMS driver
  - Configuration updates IRIS ↔ PeMS
- Video interface to D1, D2, D5 hardware
  - In progress
- Upgrade IRIS client to OpenStreetMap
  - In progress
- Ramp metering (partial complete)
- Integration with TMCAL
- Lane closure
- Inter-district travel time generation

Resources

- IRIS discussion group
  - http://groups.google.com/group/irisits
- Caltrans
  - HQ: Stan Slavin, stan_slavin@dot.ca.gov
  - D10: John Castro, john_castro@dot.ca.gov
  - D10: Mohammad Battah, mohammad_t_battah@dot.ca.gov
- Mn/DOT
  - Web site: http://iris.dot.state.mn.us
  - Jim Kranig, jim.kranig@state.mn.us
  - Doug Lau, doug.lau@state.mn.us
- UC Davis, AHMCT
  - Web site: http://iris.ahmct.ucdavis.edu
  - Kin Yen, ksyen@ucdavis.edu
  - Ty Lasky, talasky@ucdavis.edu
  - Bahram Ravani, bravani@ucdavis.edu
- Berkeley Transportation Systems
  - Web site: http://iris.bt-systems.com
  - Rob Hranac, rob@bt-systems.com
  - Michael Darter, michael@bt-systems.com