National Aeronautics and Space Administration

European Ground System Architecture Workshop (ESAW)
June 16-17, 2015
Darmstadt, Germany

Using Publish/Subscribe Messaging for System Status and Automation

Dan Smith
GMSEC Project Manager
NASA Goddard Space Flight Center
Software Engineering Division
Greenbelt, Maryland USA
Dan.Smith@nasa.gov

ABSTRACT

The NASA Goddard Mission Services Evolution Center (GMSEC) system is a message-based plug-and-play open system architecture used in many of NASA mission operations centers. This presentation will focus on the use of GMSEC standard messages to report and analyze the status of a system and enable the automation of the system's components. In GMSEC systems, each component reports its status using a keep-alive message and also publishes status and activities as log messages. In addition, the components can accept functional directive messages from the GMSEC message bus. Over the past several years, development teams have found ways to utilize these messages to create innovative display pages and increasingly sophisticated approaches to automation. This presentation will show the flexibility and value of the message-based approach to system awareness and automation.

Ground System Architecture Principles

- Different missions may have different needs
  - Do not believe in the “one size fits all” approach
  - Do believe use of common configurable products as appropriate
- Must allow for continual innovation
  - Must support missions that last many years
  - Missions should be able to upgrade, replace, or add components over time
- Must allow for industry participation
  - In the U.S., the commercial sector competes for available work
  - The commercial vendors are the source for much of the innovation
  - They work to have the most advanced capabilities for the fairest price
- Must have processes to promote broad use while controlling the common baseline
  - Configuration control boards, governance approach, open source distribution

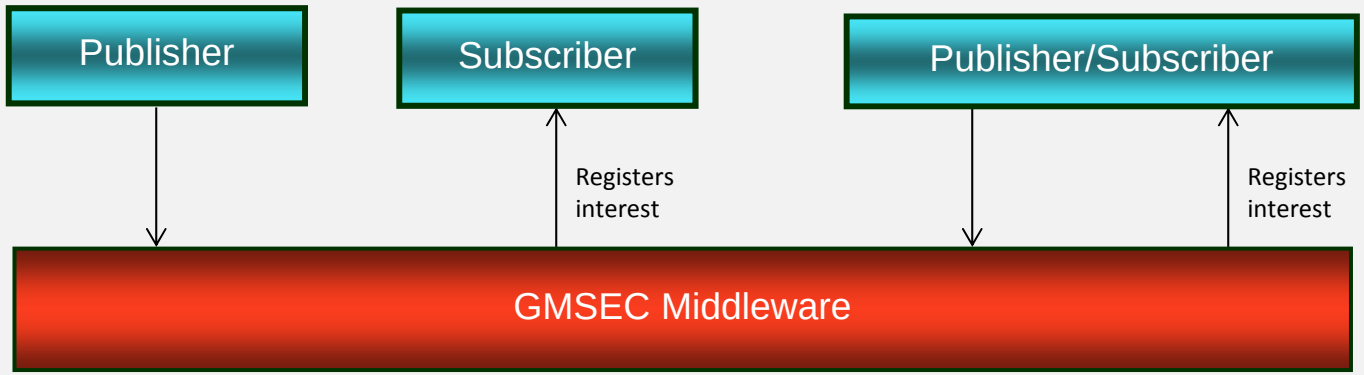
GMSEC Introduction

The Goddard Mission Services Evolution Center (GMSEC) is a proven satellite mission operations center open architecture software framework for use at the mission, fleet, or enterprise level.

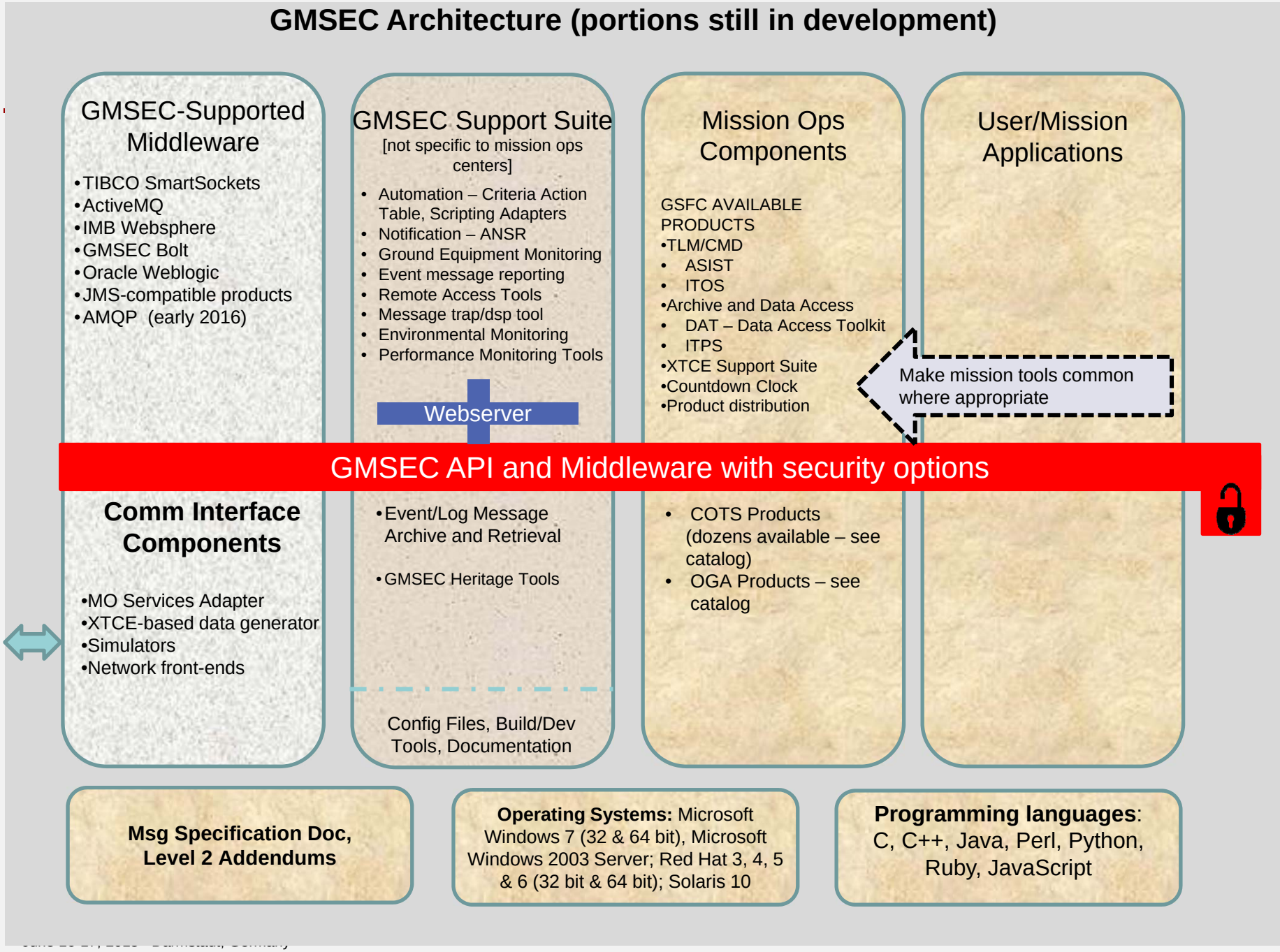
Key Features

- Publish/Subscribe approach with standardized messages
- Supports multiple middleware products via standard API
- Allows for easy integration of many commercial and in-house products
- Encourages shared software or capabilities across organizations
- Operational since 2005, used on many missions since then

The GMSEC API is released as Open Source and we are considering the release of many of the supporting software components.

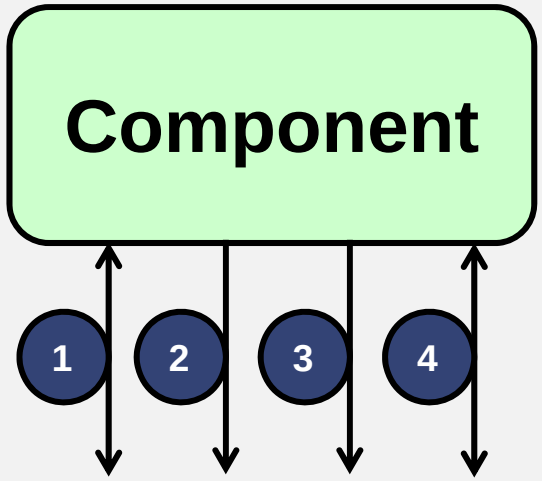


GMSEC Architecture (portions still in development)



Simple Requirements for Each Component

1. Must perform its designated function, using standard messages as appropriate



2. Must publish a keep-alive message on a regular interval
3. Must report actions and events as log messages
4. Must accept directive messages and generate responses

If it is a commercial component, do not create a special “GMSEC-compliant” version. Instead, develop an adapter to convert existing tool interfaces to GMSEC interfaces.

A special component also resides on each node to report status, receive directives.

GMSEC Message Subjects – Key Routable Fields

	Subject Standard	Mission Elements		Message Elements		*Miscellaneous Elements*			
Subject Elements	**Specifi-cation**	**Mission**	**Sat ID**	**Type**	**Subtype**	***me1***	***me2***	***me3***	***me4…***
	FIXED PORTION					***VARIABLE PORTION***			
	Required Elements					***Message Definition Determines Whether a Miscellaneous Element is Required or Optional***			

Telemetry Message Subject Example:

GMSEC. EOS .TERRA . MSG.LOG. *TLM.RT.RED.4.TLMSYS1.357*

Fixed, required portion

Message dependent, variable portion

(Body of the message follows the above header)

Using the Pub/Sub Messaging Approach

The architecture enables new approach for monitoring and automation

- Can "listen" for status from all components → situational awareness
- Can direct actions of component → system-wide control
- Recognize status and respond → event-driven automation

Many types of tools can be developed to provide these capabilities

Situational Awareness

- Filterable event/log message displays
- Query and report tools for analyzing consolidated events logs
- Spacecraft or ground system monitoring

System-Wide Control

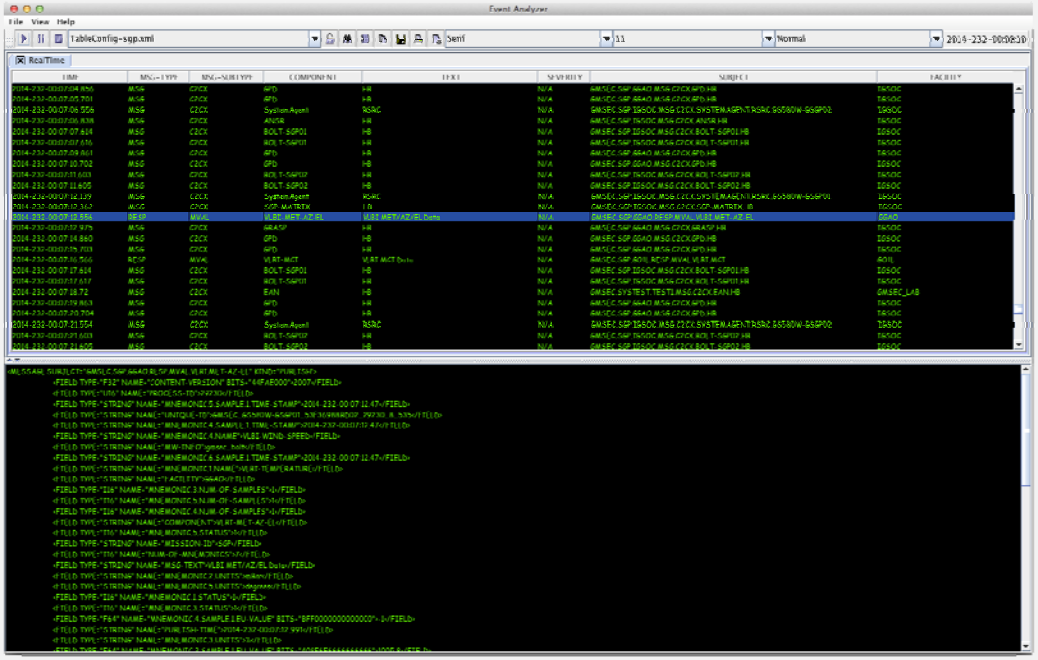
- System-wide scripting; one component can send a directive to another

Event-Driven Automation

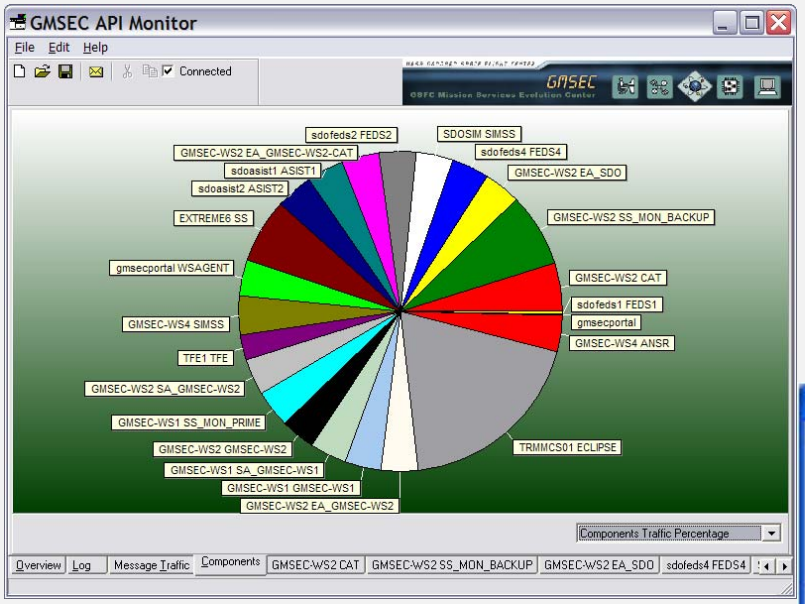
- Rule-based analysis and action tool

GMSEC Event Message Display and Analyzer

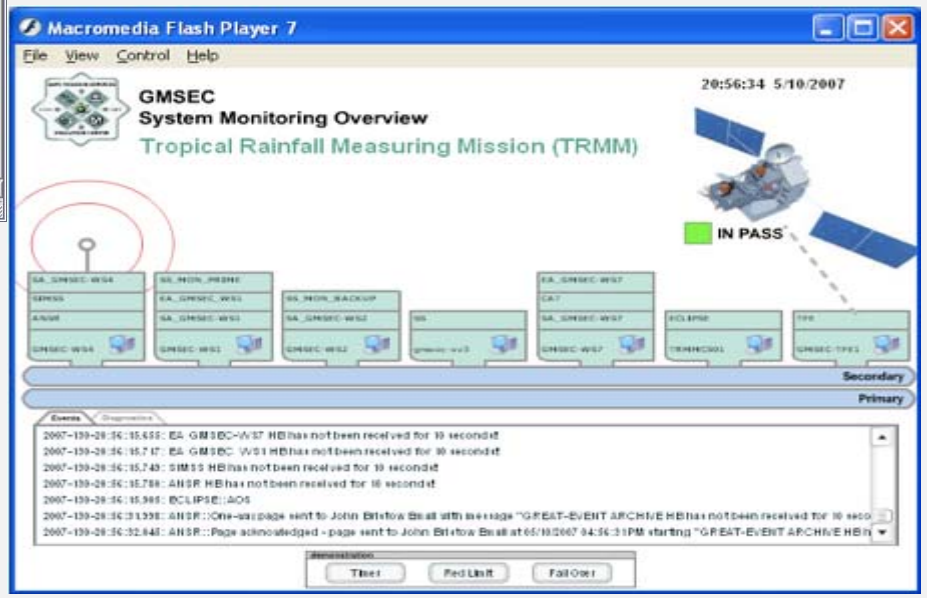
- Real-time view of message traffic on the bus
- Inspect contents of any message
- Filter view of messages
  - Declutters display
  - Focus on messages of interest
- Search messages based on topic or content
- Retrieve historical messages from GREAT database



New Tools that “Listen” to the GMSEC Bus



First steps. Subscribe to all messages and show their distribution.



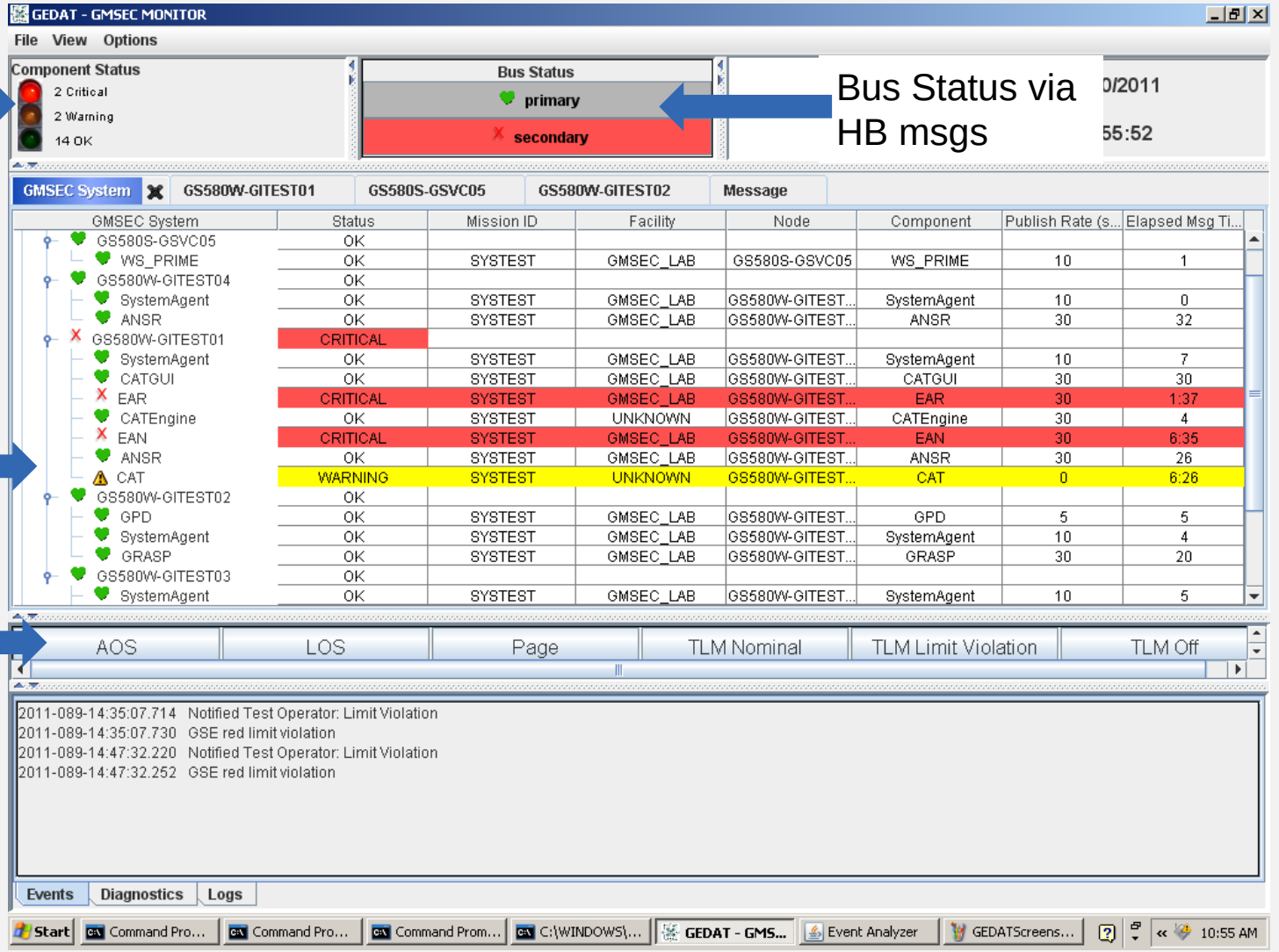
Learning more. Subscribe to keep-alives and log messages and show actual configuration and activities as they occur.

GMSEC System Data Analysis Tool

Quick-look overall status

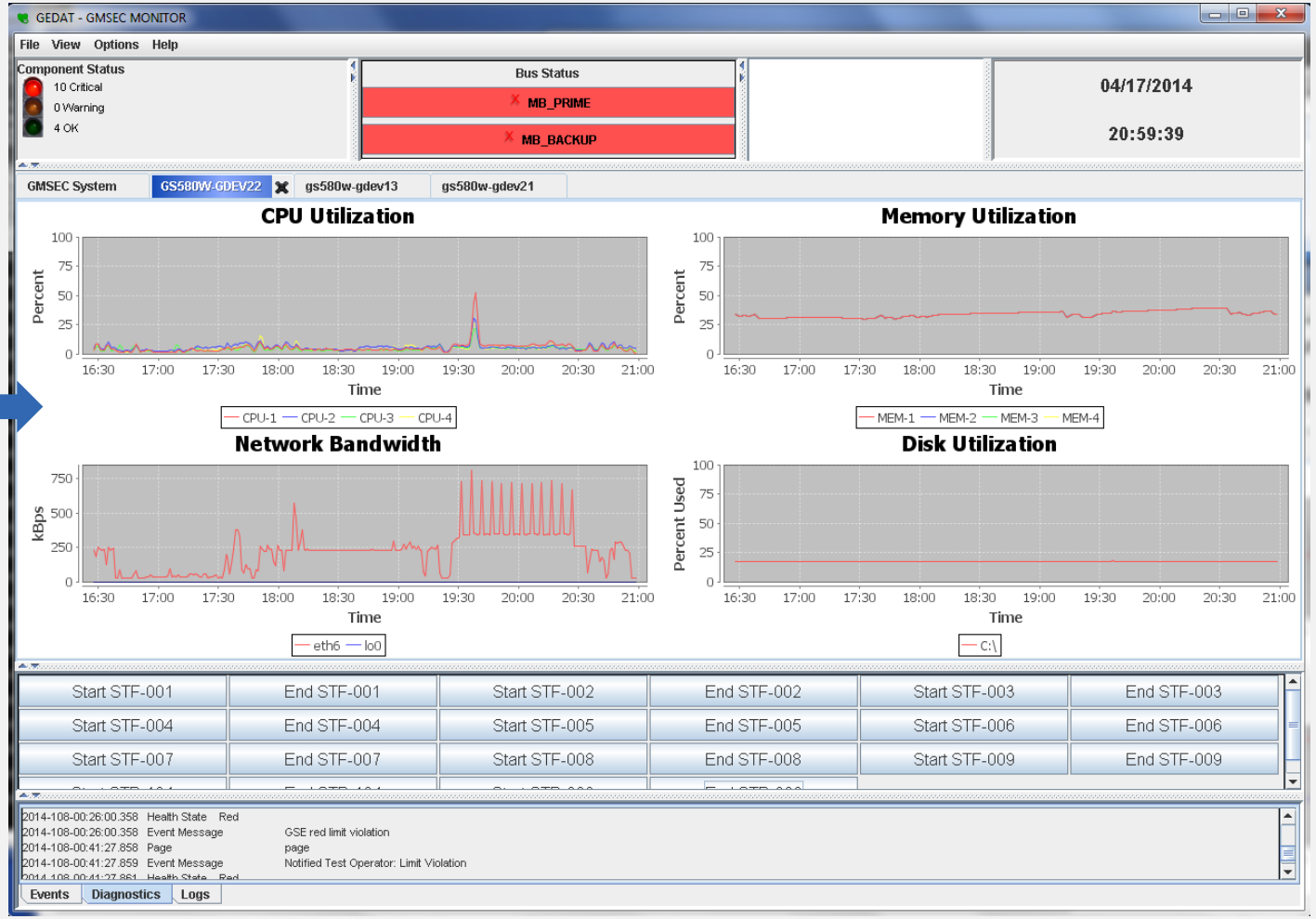
Monitor processes via heartbeat msgs

Customizable message publish buttons



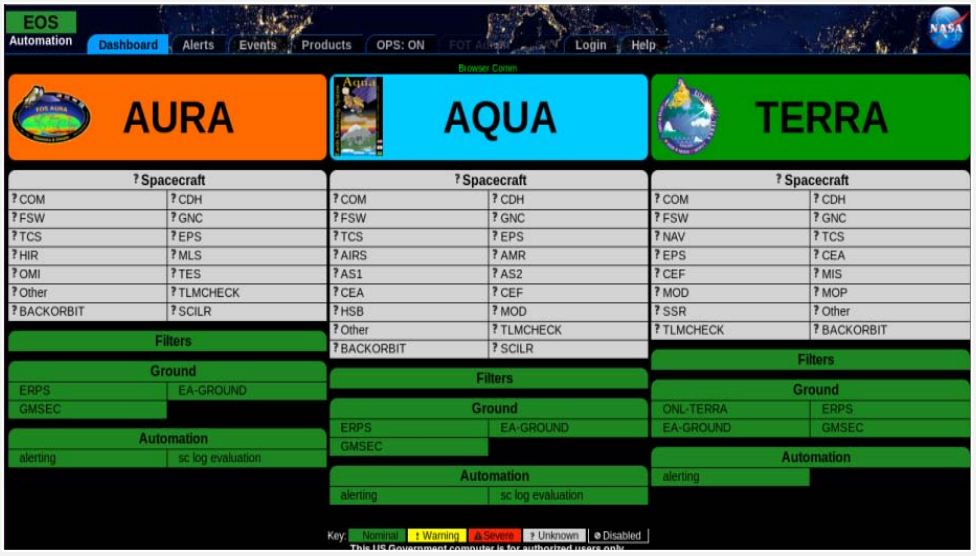
Node Resource Monitoring

Monitor resources for selected node



Framework-enabled capabilities are still maturing

Use GMSEC messages and automation rules to show progression of flight dynamics processing steps and automate the sequence of activities and notification.



Today. Process 1,000's of messages to provide drill-down status of multiple satellites and the ground system. Fully configurable, available on mobile apps. Tested at up to 5,000 messages per second.

Automation

- Script-Driven Automation
  - Since each component can accept directives, scripts can be written to send action requests to the different software components
  - Scripts can generate or subscribe to any GMSEC message
  - Can support PERL, PYTHON, and soon will be JavaScript compliant and support RUBY and other tools
  - Example. Script to run a demo that configures simulator, telemetry system, trending system, etc.

- Event-Driven Automation
  - Can combine ability to monitor event/log status with ability to send directives.
  - Criteria Action Table (CAT) tool developed to provide event-driven automation.
    - Supports timers, variables, etc.
    - Rules can be developed by Operations Team
  - Example. Scheduler says a pass should start now, if we do not get the start of telemetry within 30 seconds, then page the shift supervisor, run the diagnostic routine, and send a message to the remote site to try again to acquire the satellite.

Automation

- Another Example
  - Once a pass starts, keep a list of the first 10 parameters shown to be out of limits. After the pass, wait 30 seconds, then submit an archive retrieval request for the parameters for the last 24 hours and display the plot on the large screen in the front of the room

- Types of Rules
  - Controlling daily activity plan
  - Helping with the simple daily tasks
  - Watching for anomalous behavior and responding accordingly

- Advice
  - Start simple
  - Have several people on the team that can develop the rules
  - Only automate what is repeatable and completely understood
  - Rules should be configuration managed like other critical software

Final Notes

- At NASA/GSFC, missions are gaining sophistication in their use of the open, message-oriented GMSEC architecture
  - Most new missions find it much easier to integrate their new systems
  - Automation capabilities are getting more and more complex
  - Situational awareness tools are now evolving

The GMSEC architecture and software is enabling new levels of collaboration between government and industry to efficiently meet the long-term goals we all share.

The benefits of simplified integration, a broader set of available components, increased status and automation capabilities and the enabling of new operations concepts are realized through the open GMSEC architecture.