Autonomic Computing for Spacecraft Ground Systems

Zhenping Li,
SGT
Cetin Savkli
Matron

NASA GMSEC Lab

Agenda

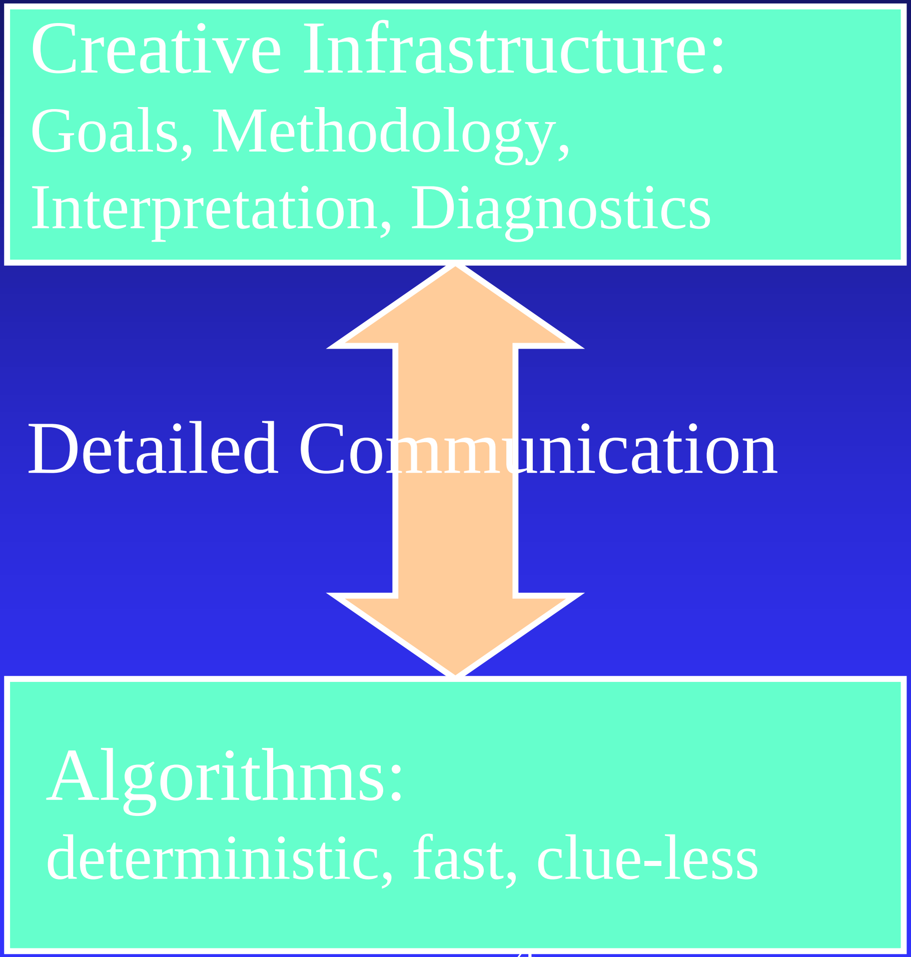
- Autonomic Computing
- GMSEC Architecture
- Criteria Action Table
  - A component approach to Autonomic Computing
- Applications
- Summary

Stinger Ghaffarian Technologies

Autonomic Computing

- Also Call Organic Computing
- A self managing and regulating computing system that enables
  - Self configuring
  - Self healing
  - Self optimizing
  - Self protection
- A new computing paradigm to meet the challenges of complex computing applications that are increasingly complex, heterogeneous, and dynamic
- For spacecraft ground systems, autonomic computing provides a systematic approach for increasing system automation and autonomy
  - Reducing the operation and maintenance costs
  - Increasing system reliability
  - Provides a rapid response to a dynamically changing environment

Algorithmic Division of Labor



Human:

Creative Infrastructure:
Goals, Methodology,
Interpretation, Diagnostics

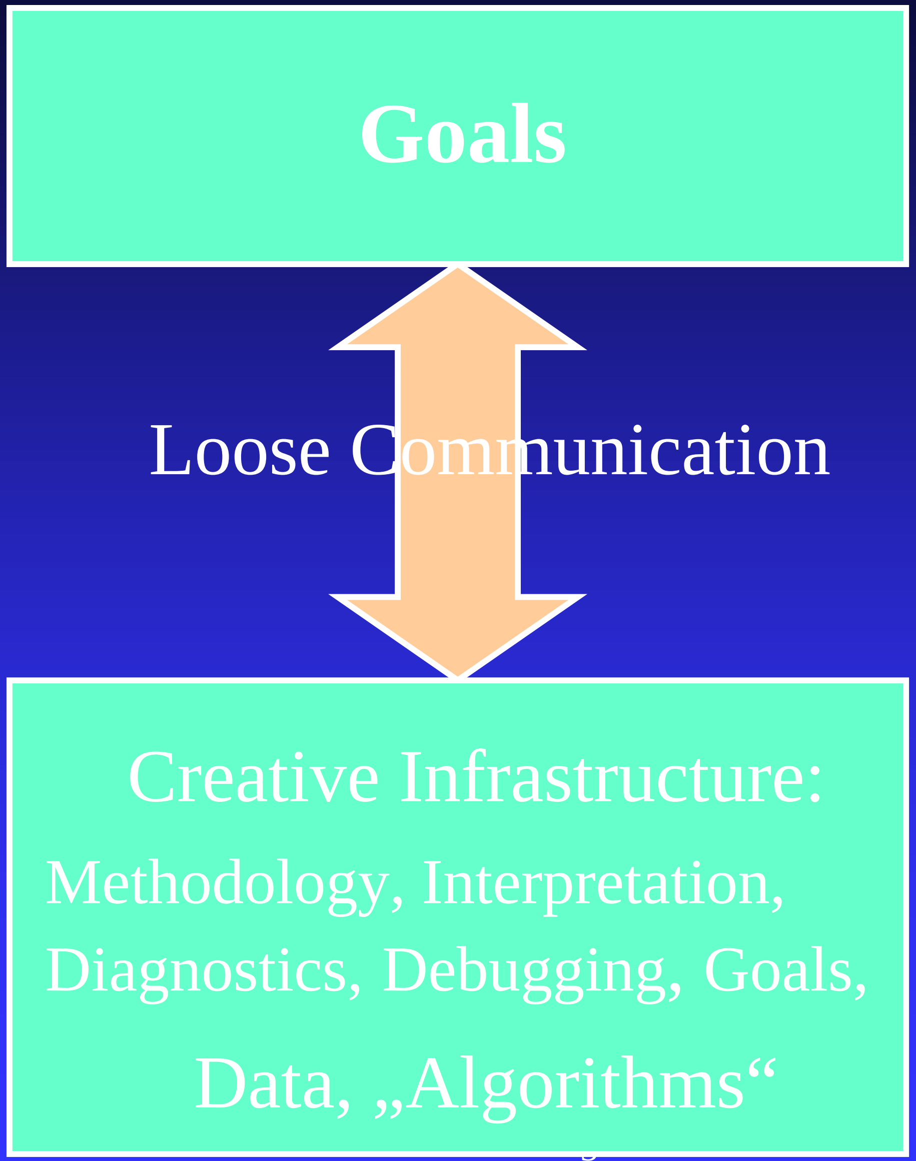
Detailed Communication

Machine:

Algorithms:
deterministic, fast, clue-less

Stinger
Ghaffarian
Technologies

Autonomic Computing: New Paradigm



Human:

Goals

Loose Communication

Machine:

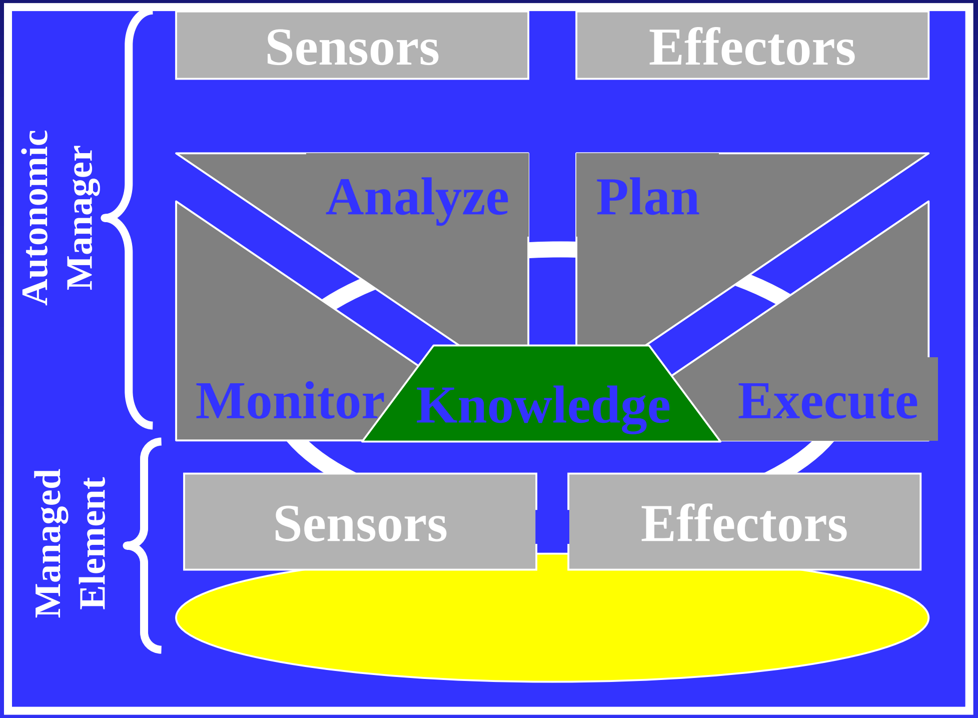
Creative Infrastructure:

Methodology, Interpretation, Diagnostics, Debugging, Goals,

Data, „Algorithms“

Autonomic Computing Structure

- Autonomic elements
  - Provides autonomic computing services
- Managed elements
  - A component or service in the system
- Autonomic elements and managed elements form a feedback control loop
  - The managed element’s behavior is based on management policies and rules



An Autonomic Element

Dynamic Data Driven Application Systems (DDDAS)

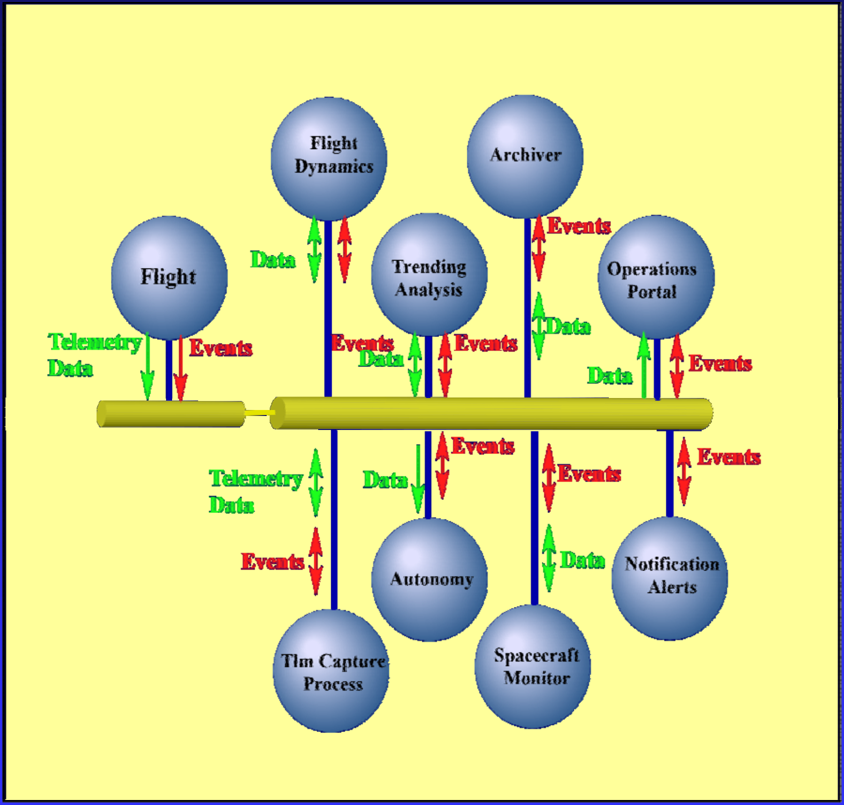
- Involves the data analysis and decision making system, scheduling and planning system, and instruments (applications) that form a symbiotic feedback control loop.
- www.cise.nsf.gov/**dddas** or www.dddas.org
- Autonomic Computing Architecture in a larger scale.
- Sensor Web is an example of DDDAS system.
    - GOES-R System has been focal point of space assets in Sensor Web architecture.

Autonomic Computing for Spacecraft Ground Systems

- Components and services already exist in ground systems
  - How to develop the autonomic computing elements without modifying the existing components significantly?
- Require solutions at both the architecture and component levels
  - Architecture level:
    - Use open standard for the interaction between the autonomic elements and managed elements
    - Create system awareness
      - System event messages that contain all necessary monitoring information for the autonomic elements
  - Component level:
    - Provide autonomic computing services
    - Configurable
      - Provide flexibility to evolve and adapt, and to add additional management rules and policies

GMSEC Ground System Architecture

- Provides flexible and cost effective approach to meet the operational needs of current and future missions
- Middleware provides the services common to all components, and is a standard for the interaction and communication among the components
  - Message routing and delivery
  - Security and guaranteed delivery
  - Open standard for the interfaces between the component and middleware
    - Easy integration
  - Open standard for the messages
    - Standard communications among the components
    - Message Standard defined for the event message, directive request & response, heartbeat, and other types of messages
- Higher interoperability and portability for components
  - Missions select different components and middleware based on their requirements



Stinger Ghaffarian Technologies

System Awareness in GMSEC Architecture

- Message Standard goes beyond “time, type, fixed length String format”
  - More information available for data analysis and correlations
- Every component within the ground and flight system publishes its status
- Every component within the ground system should accept GMSEC directive messages to change its behavior
- Creates an autonomic computing environment to manage any component or service

Header	Message Type
	Version Number
	Mission ID
	Constellation ID
	Spacecraft ID
	Facility ID
	Device Node
	Component ID
	Subcomponent ID
	Process ID
Content	Type
	Severity
	Spacecraft Time
	Event Logging Time
	Reference ID
	Message Text
	Detailed Information
	Attachment File Reference

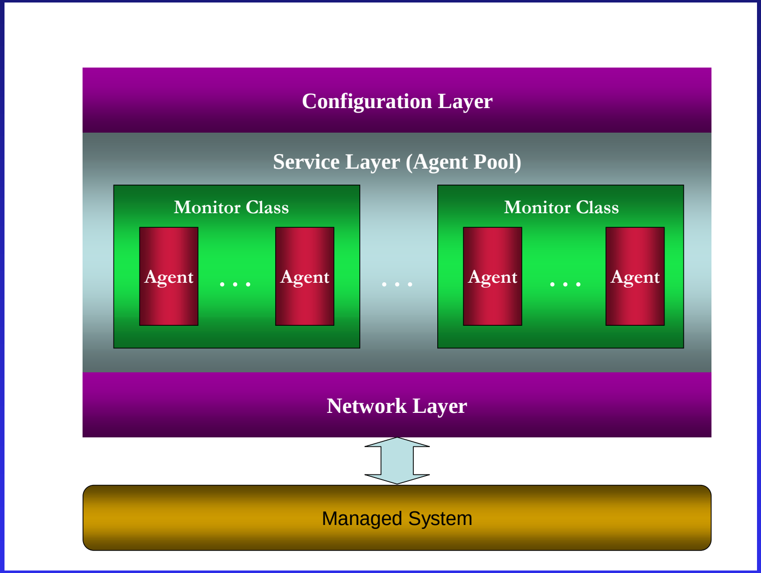
Stinger
Ghaffarian
Technologies

Autonomic Component: Criteria Action Table

- The managed elements are services provided by components within a system
- Service Oriented:
  - A component generally provides more than one service
    - Health and safety status service
    - Domain Specific services
  - The messages associated with a service and published by a component generally have a fixed template
    - Enables easy extraction of the relevant information from the messages
- Scalable:
  - Monitor class
    - A group of autonomic agents that manage the same service provided by different components or entities
    - Follows the same management policies and rules
    - Have the same life cycle
- Adaptive:
  - Lifecycle of autonomic agents management
  - Agents are created dynamically
    - Generally triggered by an event
    - Can be terminated if it is no-longer needed
- Extensible:
  - Provides a platform for more sophisticated data analysis and decision making capabilities
- Collaborate:
  - Allow the agent collaborations

CAT Architecture

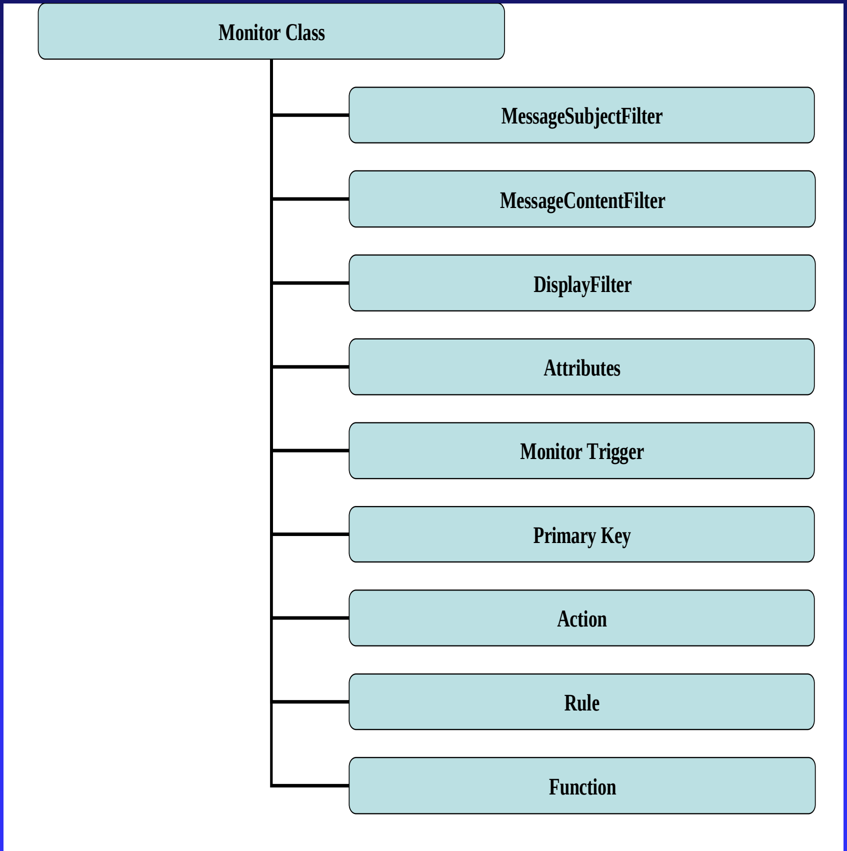
- Configuration Layer
  - XML configuration files that contain the domain specific instructions on how information is processed, the rules to apply to that information, and the responses that are required
- Service layer:
  - Agent pool manages the lifecycle of agents
  - Routes the relevant message to agents
  - Agents interact among each other within the agent pool
    - One agent can access the attribute values in the same or different monitoring classes
- Network Layer
  - Interface with GMSEC middleware
  - Monitor the system
  - Generate the directive or event messages that are sent to pre-defined destinations



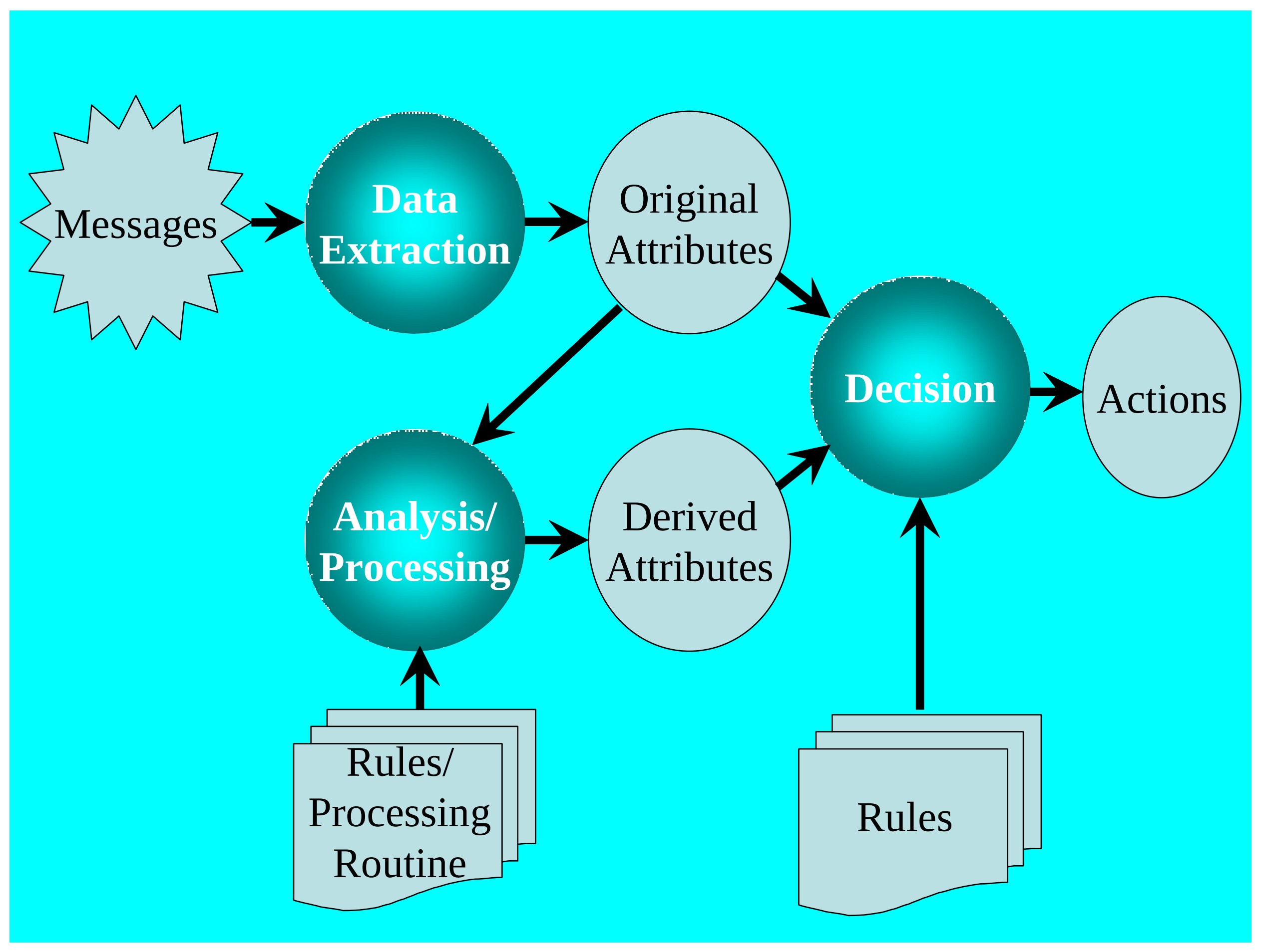
Stinger Ghaffarian Technologies

The Agent Representation

- Basic representation for data processing is the "attribute"
- Filters ensure only relevant messages are routed to the agents
- Monitor Trigger enables agents to be created dynamically
- Primary Key provides the unique identifier for an agent within a monitor class
- Function provides a platform for data analysis
  - Additional data analysis routines can be easily added
- Rules, using specific criteria, determine if pre-determined actions are to be taken
  - Actions, associated with the rules, provide the information on how the messages are to be constructed



Data Processing



Stinger Ghaffarian Technologies

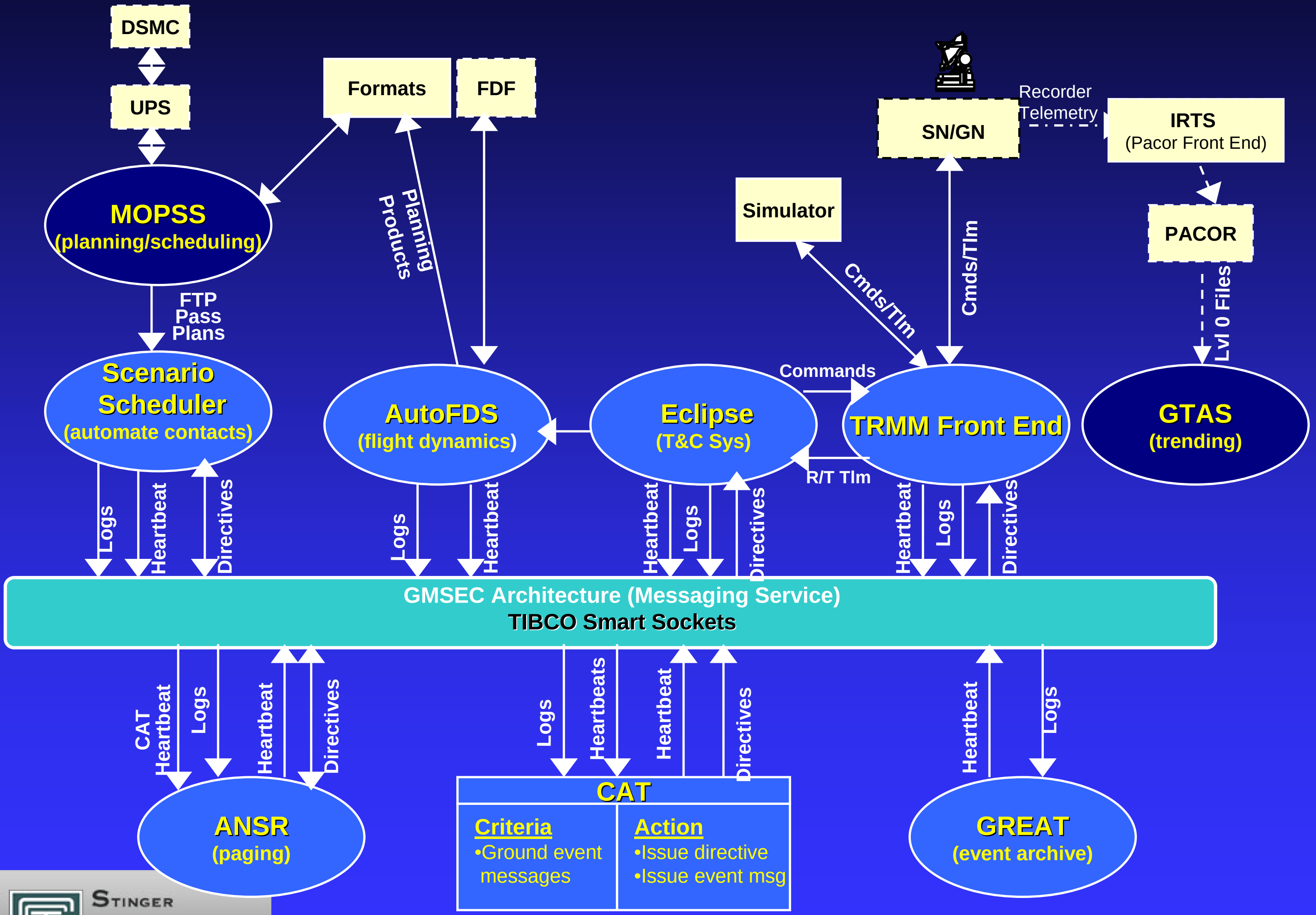
Examples: HeartBeat Configurations

```
          <monitor-class name="HeartBeatMonitor" enabled = "true">
          <subject-constraint>
          <requirement attribute="SUBJECT" operator="~" value=".*C2CX.*"/>
          </subject-constraint>
          <class-constraint>
          <requirement attribute="MESSAGE-SUBTYPE" operator="~" value=".*C2CX.*"/>
          <requirement attribute="COMPONENT" operator="!~" value="CAT"/>
          </class-constraint>
          <primary-key>
          <key order="0">component</key>
          </primary-key>
          <attributes>
          <attribute name="component” type="String” field="COMPONENT" pattern="(.*)"/>
          </attributes>
          <rule name=“GIVE_UP" enabled=“true">
          <act>GIVE_UP</act>
          <requirement attribute="t_sinceReceivingLastMsg" operator=">" value="5"/>
          </rule>
          <action name="GIVE_UP">
          <destination type="LOG">GMSEC.DEMO.LOG.CAT</destination>
          <text field="SEVERITY">4</text>
          <text field="MSG-TEXT">frequency=${t_sinceReceivingLastMsg} component=${component}
          Heart beat missing </text>
<text field="COMPONENT">CAT</text>
          </action>
          </monitor-class>
```

Current Applications

- GMSEC Architecture has become open standard for GSFC ground systems
  - Autonomic computing approach is the key for increasing automation and reducing the cost of operation and maintenance
- Existing Re-engineering efforts
  - Tropical Rainfall Measuring Mission (TRMM)
    - Reducing operational costs by 50 percent
    - Enabling lights out operations
  - Small Mission Explorer (SMEX)
  - Current EOS automation efforts on Terra, Aqua, and Aura
    - More complex configurations are being implemented
- New GSFC Missions
  - Discussion/Planning: LRO, GLAST
  - Sharing Tools and approach: JWST
  - Implementing: ST5
  - Co-developing: MMS and GPM
- Other NASA centers using GMSEC architecture approach as well

TRMM Reengineered Architecture



DSMC

UPS

MOPSS (planning/scheduling)

FTP Pass Plans

Scenario Scheduler (automate contacts)

Formats

FDF

Planning Products

AutoFDS (flight dynamics)

Eclipse (T&C Sys)

Simulator

Cmds/Tlm

SN/GN

Recorder Telemetry

IRTS (Pacor Front End)

PACOR

Lvl 0 Files

Commands

R/T Tlm

TRMM Front End

GTAS (trending)

Logs

Heartbeat

Directives

GMSEC Architecture (Messaging Service)
TIBCO Smart Sockets

CAT Heartbeat

Heartbeats

ANSR (paging)

CAT

Criteria	Action
•Ground event messages	•Issue directive •Issue event msg

GREAT (event archive)

Summary

- Autonomic Computing requires solutions at both architecture level and component level
- Increasing the autonomic maturity requires a service oriented architecture
  - Service monitoring becomes standard
    - More adaptive
  - System awareness is still required
- A new paradigm for increasing automation at the system level