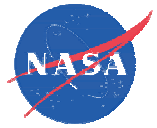
# Applying MDA to SDR for Space to Model Real-time Issues

NASA space communications systems have the challenge of designing SDRs with highly-constrained Size, Weight and Power (SWaP) resources. A study is being conducted to assess the effectiveness of applying the MDA Platform-Independent Model (PIM) and one or more Platform-Specific Models (PSM) specifically to address NASA space domain real-time issues. This paper will summarize our experiences with applying MDA to SDR for Space to model real-time issues. Real-time issues to be examined, measured, and analyzed are: meeting waveform timing requirements and efficiently applying Real-time Operating System (RTOS) scheduling algorithms, applying safety control measures, and SWaP verification. Real-time waveform algorithms benchmarked with the worst case environment conditions under the heaviest workload will drive the SDR for Space real-time PSM design.

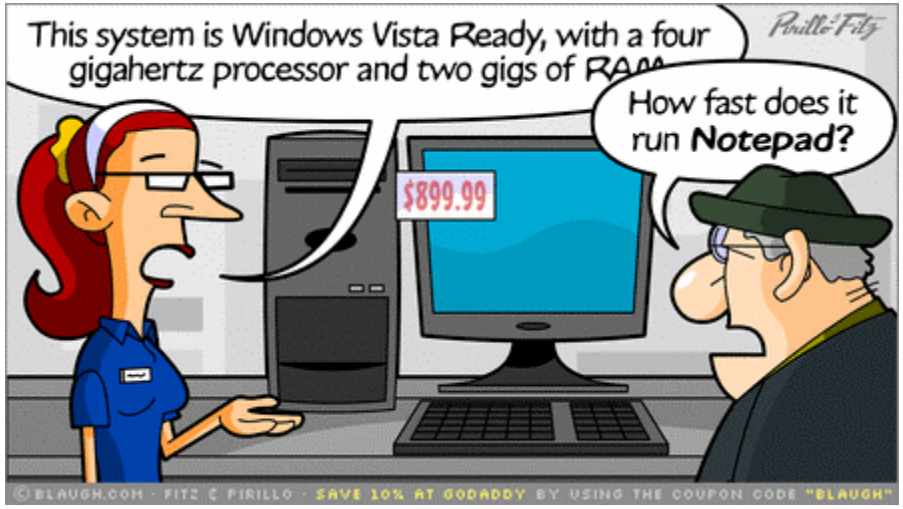



# SBC Workshop
# Applying MDA to SDR for Space to Model Real-Time Issues

OMG's Third Annual
Software-Based Communications Workshop:
Realizing the Vision
March 5-8, 2007; Fairfax, VA USA

**Tammy M. Blaser**
**Tammy.M.Blaser@nasa.gov**
**NASA's Space Telecommunication Radio System (STRS) Project**
**Software Defined Radio (SDR) for Space**
**NASA Glenn Research Center**

# Model Real-Time Issues




It is a misconception that real-time systems must be "fast“ … the real challenge is to design a predictable and reliable real-time system that can also run NASAs' high Gbps SDR Waveforms?

# Introduction

- **Project**
  - NASA Space Telecommunications Radio System (STRS)
- **Objective**
  - Conduct a preliminary (initial phase) study to access the effectiveness of applying Model Driven Architecture (MDA) Platform Independent Model (PIM) and one or more Platform Specific Models (PSMs) specifically to address NASA space domain *Real-Time* issues.
- **Goal**
  - Develop a prototype to study NASA Software Defined Radio (SDR) Real-Time issue classes and extract key SDR for Space PIM/PSM properties and performance requirements. Define a phased prototype development approach. **Complete the initial phase** prior to *this* OMG Third Annual SBC Workshop.
- **Examine, Measure and Document**
  - Specific scheduling algorithms, concurrency and resource management techniques
  - Safety control measure techniques
  - Size Weight and Power (SWaP) verification methods
  - Benchmark workloads

Goal Met

# STRS Mission-Critical Requirement Drivers

- Radiation Suitable Processing
  - Less capable than terrestrial, often lagging by a generation or two.
  - Limits both the footprint and complexity of the infrastructure.
- Spacecraft Resource Constraints
  - SWaP limitations on spacecraft.
  - Open architecture overhead must be balanced against these spacecraft constraints.
- Reliability
  - Designed to prevent single point failures.
  - Manned missions have high reliability requirements, especially for safety critical applications.
- Specialized Signal Processing (SSP) Abstraction
  - Waveforms to be deployed on specialized hardware (FPGAs, ASICS)
- Space Waveforms
  - Data rates range from low (kbps) to high (Gbps). Frequencies from low (MHz) to (GHz)

# Future Space Communications Architecture
## A Network of Networks

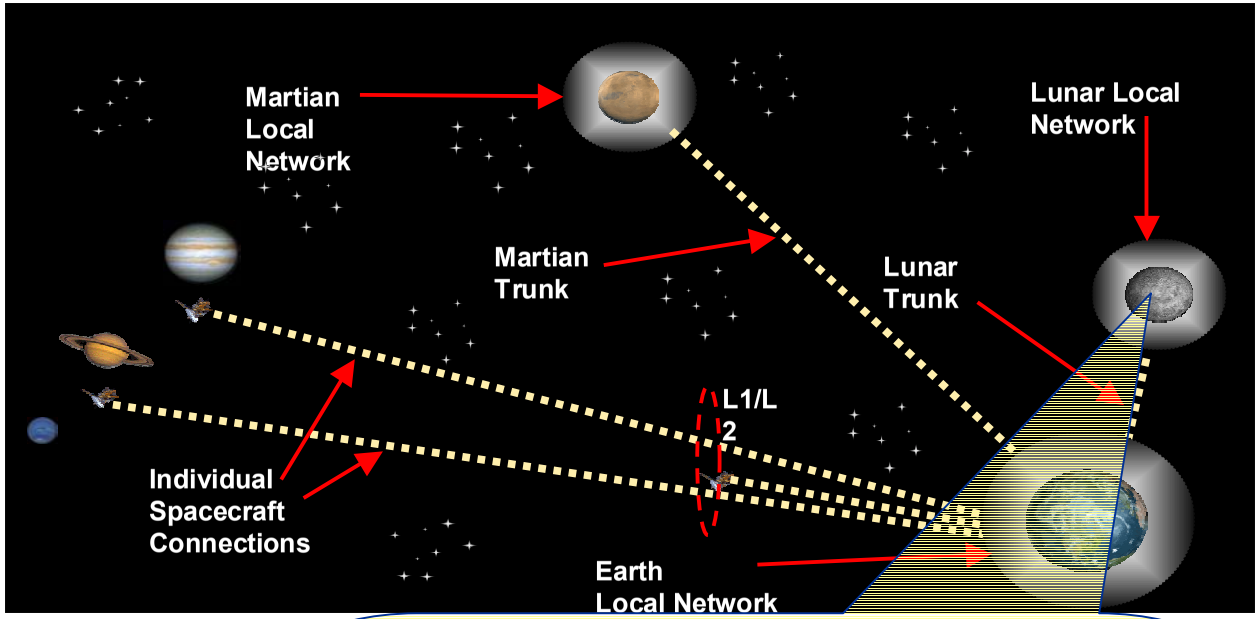
### Mission Types

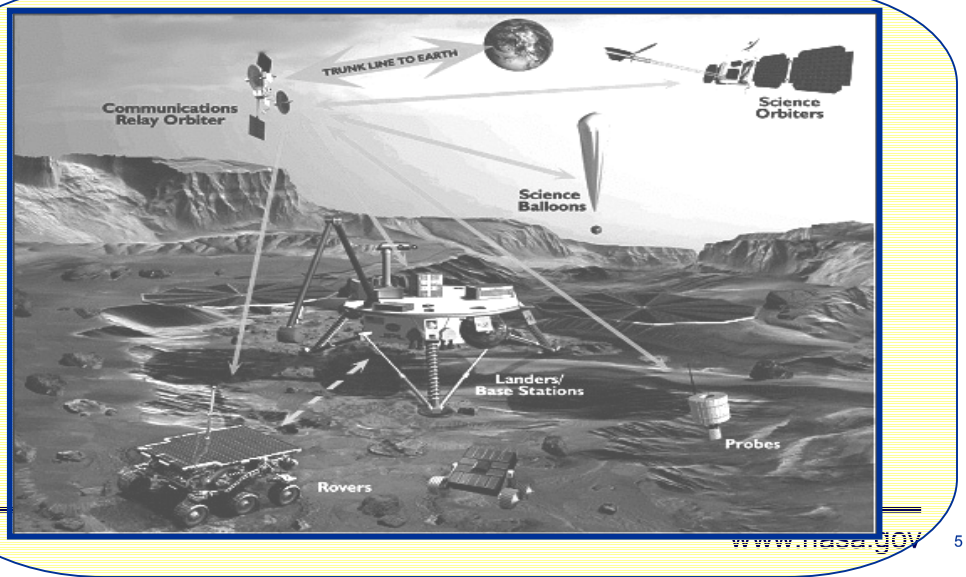
Crewed Elements
- Transport Vehicles
- Space Stations/Outposts
- Crew Activity (i.e. EVA)

Spacecraft Links
- Science Satellites
- Orbiting Relay Satellites

Surface Radios
- Rovers
- Science Elements
- EVA

Other…
- Launch Vehicles
- Sub-Orbital Vehicles
- Ground Stations



# STRS Missions Resource Capacity Constraints

- STRS mission resource capacity constraints (coding, system/subsystem interfaces, data rates, frequency bands, transponder/transceiver, etc.) vary based on mission type and classes …
  - Small Mission Class: Low intrinsic complexity and low data rate signals
  - Medium Mission Class: Moderate complexity and medium to at least one high-data-rate transmit signal
  - Large Mission Class: High functional complexity with mixture of low, medium, and high data rate signals up to at least one ultra-high data rate transmit Signal

| Mission Type/Class | Small Mission | Medium Mission | Large Mission |
|---|---|---|---|
| Transport Vehicles | | x | |
| Space Stations/Outposts | | | x |
| Crew Activity (i.e. EVA) | x | | |
| Science Satellites | | x | |
| Orbiting Relay Satellites | | | x |
| Rovers | x | | |
| Science Elements | x | | |
| Surface Radio EVAs | x | | |
| Launch Vehicles | x | x | |
| Sub-Orbital Vehicles | x | x | |
| Ground Stations | | | x |

# Approach

- **Three Phased Development**
  - Initial phase
    - Develop a prototype framework to initially study real-time STRS issues and obtain preliminary results
  - Interim phase
    - Integrate STRS Software Application Programming Interface (API) together with OMG's PIM and PSM for SWRadio
    - Perform schedulability and timeliness analysis of STRS waveform models
  - Final phase
    - Use integrated STRS/SWRadio to implement Waveforms (WFs)
      - Best Practices Guide for implement WF
      - Hardware Abstraction Layer (HAL) Recommendations
    - Integrate WF designs with prototype

# Initial Phase Approach

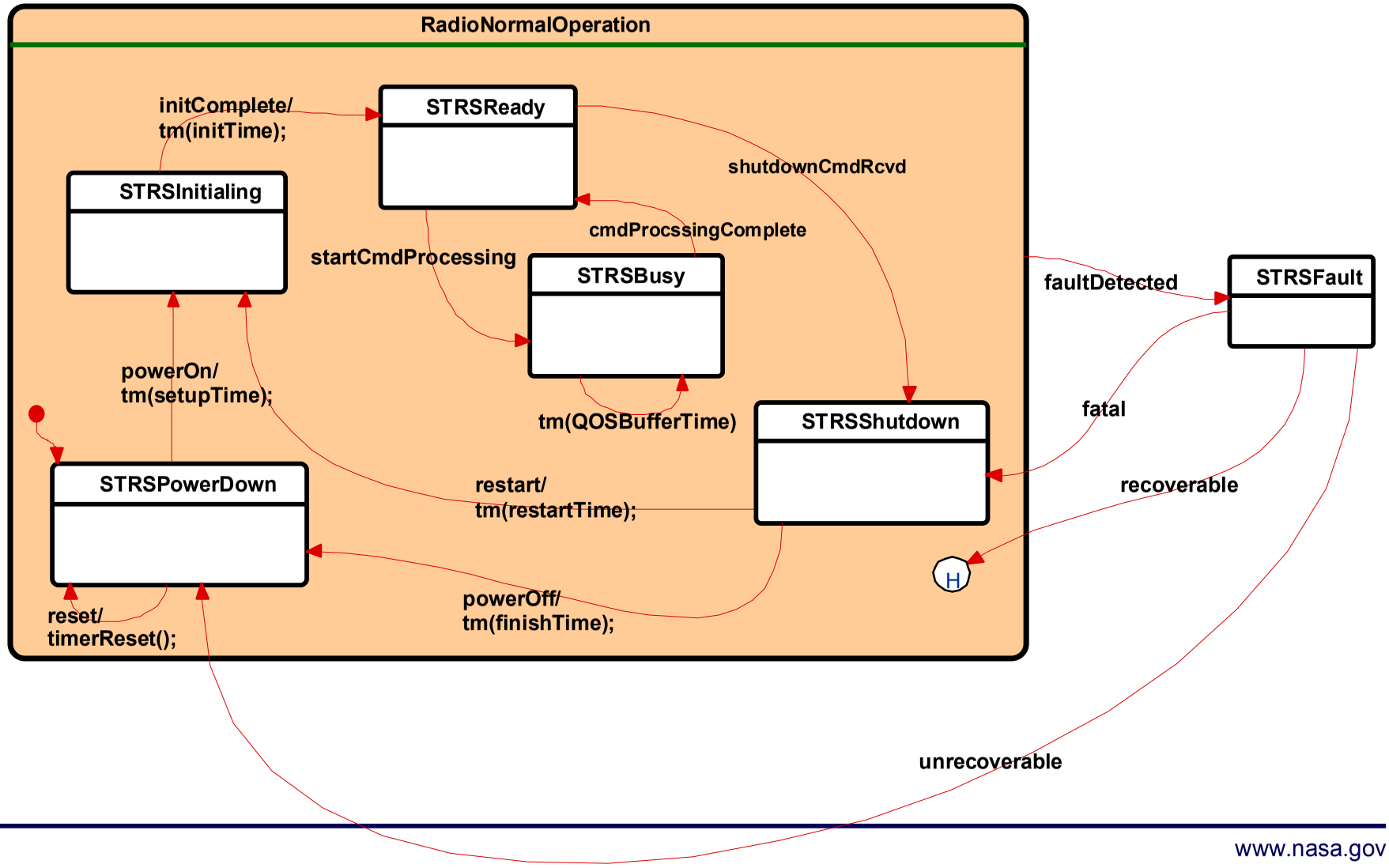
- **Tools Used for Initial Phase**
  - Rhapsody
    - Modeling
    - Design-Level Debugging via Rhapsody Animation
    - Rhapsody Object Execution Framework (OXF) Interface to INTEGRITY RTOS
  - INTEGRITY
    - INTEGRITY simulator for ARM
      - One of NASAs' STRS Reference Implementation test bed efforts uses the Davinci ARM core with a Texas Instruments (TI) Digital Signal Processor (DSP)
    - POSIX APIs (as applicable)
      - STRS requires the use of POSIX for medium and large mission classes

# Main Presentation Topics

- **Subsystems and Services Studied**
  - Real-Time Waveform Control
  - Resource Management
  - Hardware Abstraction Layer (HAL)
  - Fault Management
  - SWaP Verification
  - Performance Monitor and Benchmark Service
- **Design Patterns Analyzed**
  - Real-Time Design Patterns [Bruce Douglass]
  - Object Oriented (OO) Design Patterns [Gang-of-Four (GoF)]
- **Mapping to SWRadio PIM/PSM**

# Real-Time Waveform Control Topic

## STRS Radio State Chart - Rhapsody Animation Preparation



# Real-Time Waveform Control and Resource Management Topic

- Concurrent Waveform Control Requirements
  - Example: SDR for Space Concurrent GPS Rx operations with ISS S-band
- RTOS Specific Scheduling Algorithm
  - **INTEGRITY Partition Scheduler**
    - Partition Scheduling contains its own kernel (manages concurrency)
    - However disables the INTEGRITY Resource Manager
- Concurrent Real-Time Design Patterns [Bruce Douglass]
  - Message Queuing, Interrupt, Guarded Call, Rendezvous, Cyclic Executive, Round Robin, Static Priority (i.e. Rate Monotonic Scheduling (RMS)), Dynamic Priority
- Resource Real-Time Design Patterns [Bruce Douglass]
  - Critical Section, Highest Locker, Simultaneous Locking, Ordered Locking, Priority Inheritance, **Priority Ceiling**

Studied in Initial Phase, Test Case and Results Follows

Many Concurrent and Resource Design Patterns Available

Studied in Initial Phase, Class Diagram Follows

# Real-Time Waveform Control and Resource Management Topic

## INTEGRITY Partition Scheduler Testing

- STRS may use separate address spaces to satisfy concurrency requirements while providing a reliable and/or secure design.
  - Applicable to larger mission classes and some medium mission classes.
- The INTEGRITY Partition Scheduling library can be used to schedule tasks on a per-AddressSpace basis.
- Test Results using 3 virtual AddressSpaces, scheduled as follows:
  - P1 runs for .1 of every .5 seconds
  - P2 runs for .2 of every 1 second
  - P3 runs for .2 of every 1 second
- The tasks that run periodically display (to the console) indicating the partition number and task. Tasks in the first partition run followed by the 2nd partition (for twice as long), followed by the first partition again, etc.
- Tasks run in separate partitions enabling a more reliable and/or secure design but requires more memory due to additional virtual address space overhead
- Note: INTEGRITY working on a high-resolution partition scheduler which may deem helpful for scheduling STRS high (Gbps) Waveforms.

# Resource Management Topic

## Priority Ceiling Pattern Class Diagram - Resource Management Candidate

**Problem Context:**

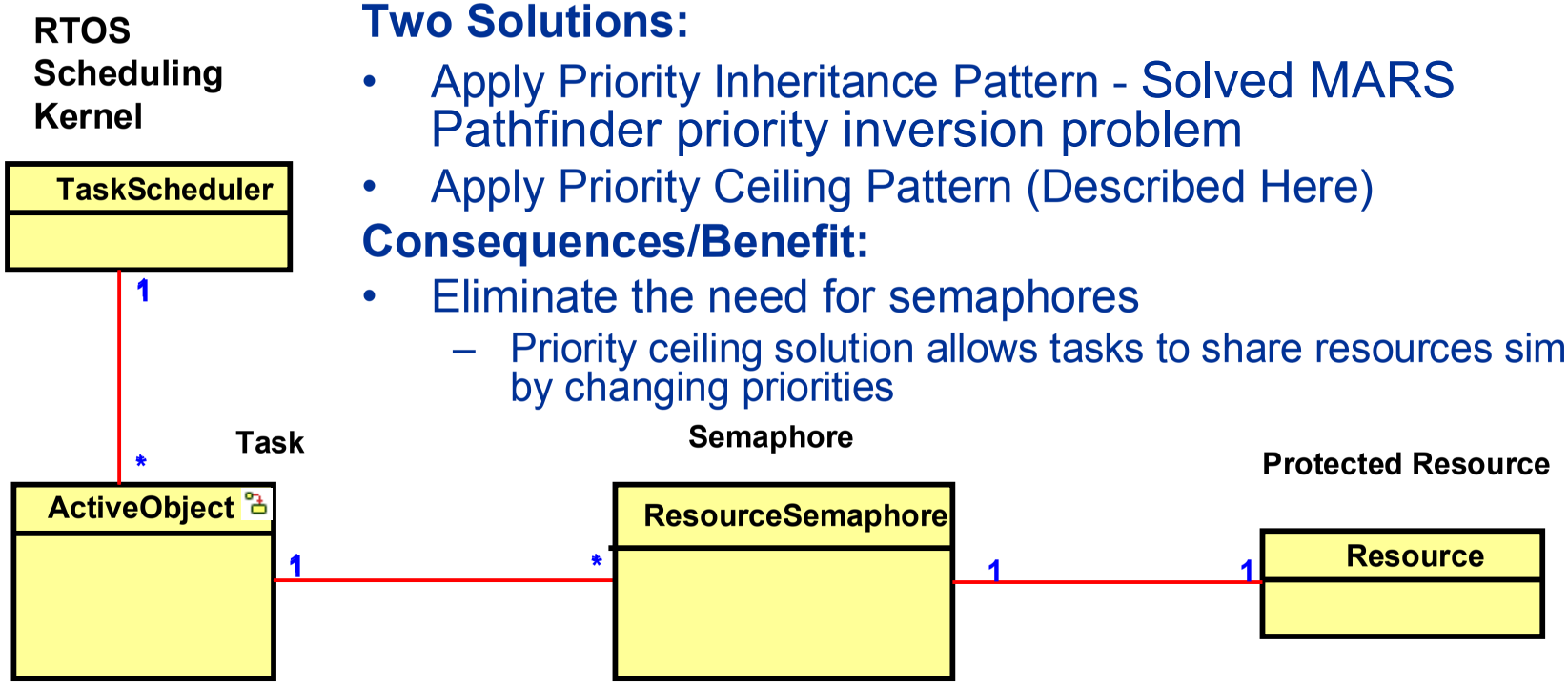
- Highest priority, ready tasks may not run when needed, due to events outside the RTOS scheduler's control, due to a lower priority task running locking a shared resource
  - Priority Inversion
- Critical deadline could be missed, causing the system to fail

**Two Solutions:**

- Apply Priority Inheritance Pattern - Solved MARS Pathfinder priority inversion problem
- Apply Priority Ceiling Pattern (Described Here)

**Consequences/Benefit:**

- Eliminate the need for semaphores
  - Priority ceiling solution allows tasks to share resources simply by changing priorities



# Hardware Abstraction Layer (HAL) Topic

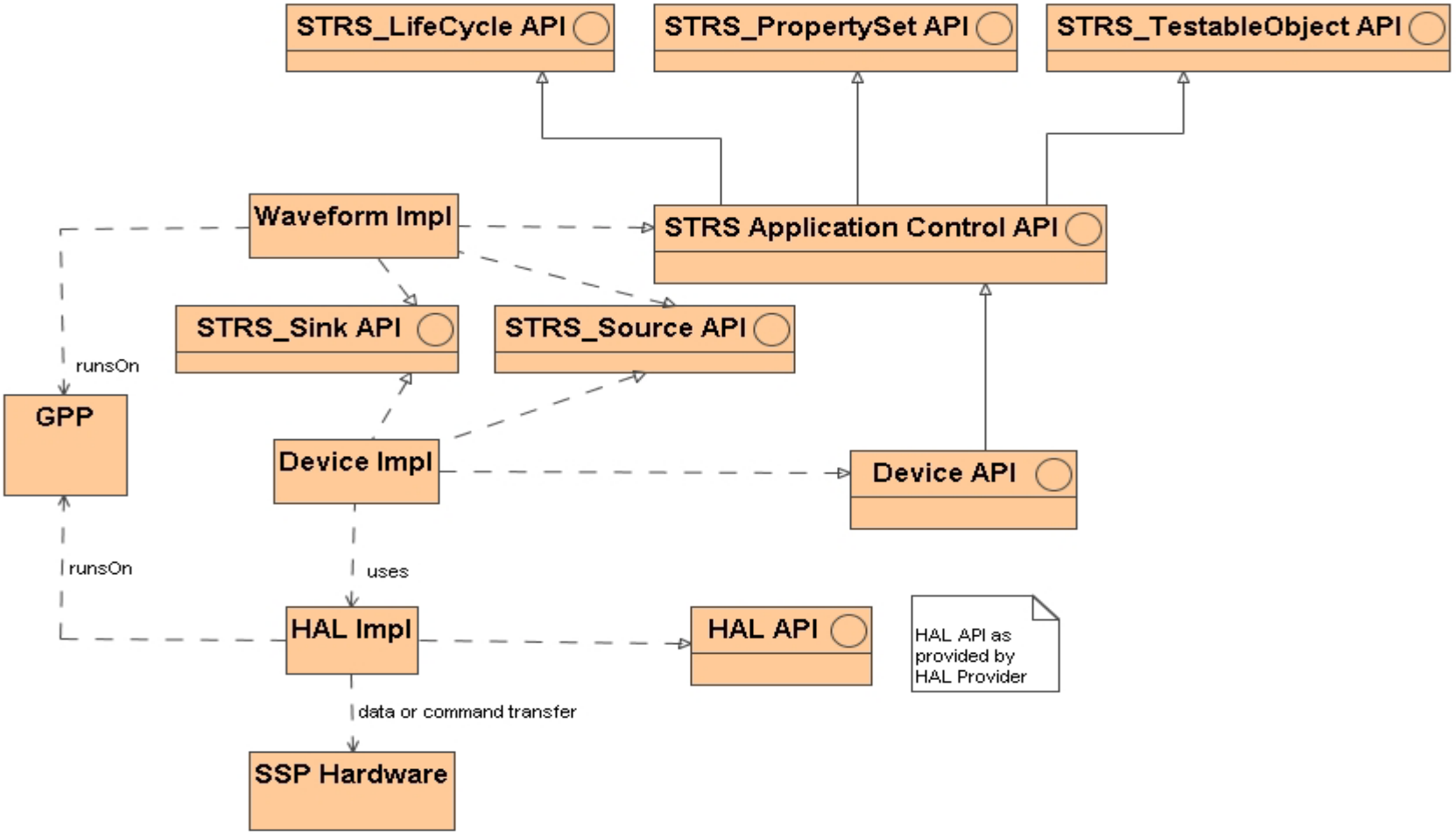
## HAL Design Approaches

- The HAL design has been very challenging, no standard available
  - Several approaches have been initiated
    - DoD: M-HAL
      - JTRS HAL standard but not available
    - CP289
      - Component Portability Specification
    - Integrated Circuit ORB (ICO)
      - FPGA middleware technology
- STRS needs a non-proprietary standard HAL design
  - Candidate OO Design Patterns [GoF]:
    - Strategy
    - Adapter
    - Bridge
      - 2 levels: Proxy at the GPP, and Data/Command Transfer for (SSP) devices

# Hardware Abstraction Layer (HAL) Topic

## STRS Preliminary HAL Class Diagram

### Bridge Pattern - Design Applied for HAL



# Fault Management Topic

## Safety and Reliability Techniques

- Safety and Reliability Real-Time Design Patterns [Bruce Douglass]
  - Protected Single Channel, Homogeneous Redundancy, Triple Modular Redundancy, Heterogeneous Redundancy, Monitor-Actuator, Sanity Check, Watchdog, **Safety Executive (AKA Safety Kernel)**

Studied in Initial Phase, Class Diagram Follows

# Fault Management Topic

## Safety Executive Class Diagram Pattern

**Problem Context:**

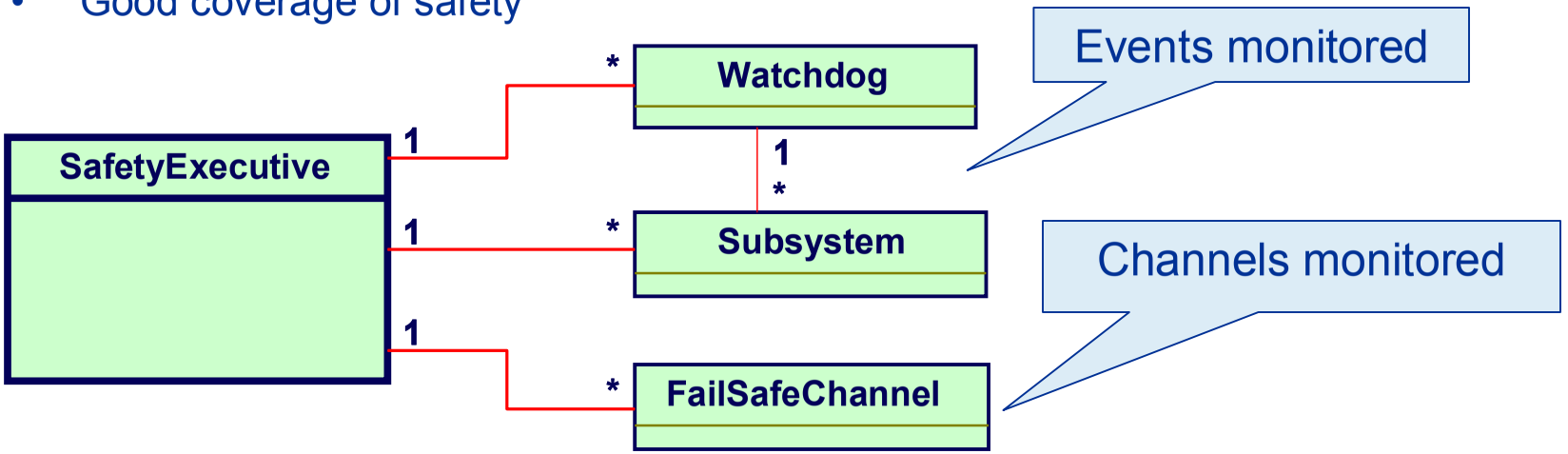
- Provide safety in complex multi-actuator systems or systems which require complex fault Recovery behavior

**General Solution:**

- Use a safety executive to oversee fault identification, isolation, and recovery processes.

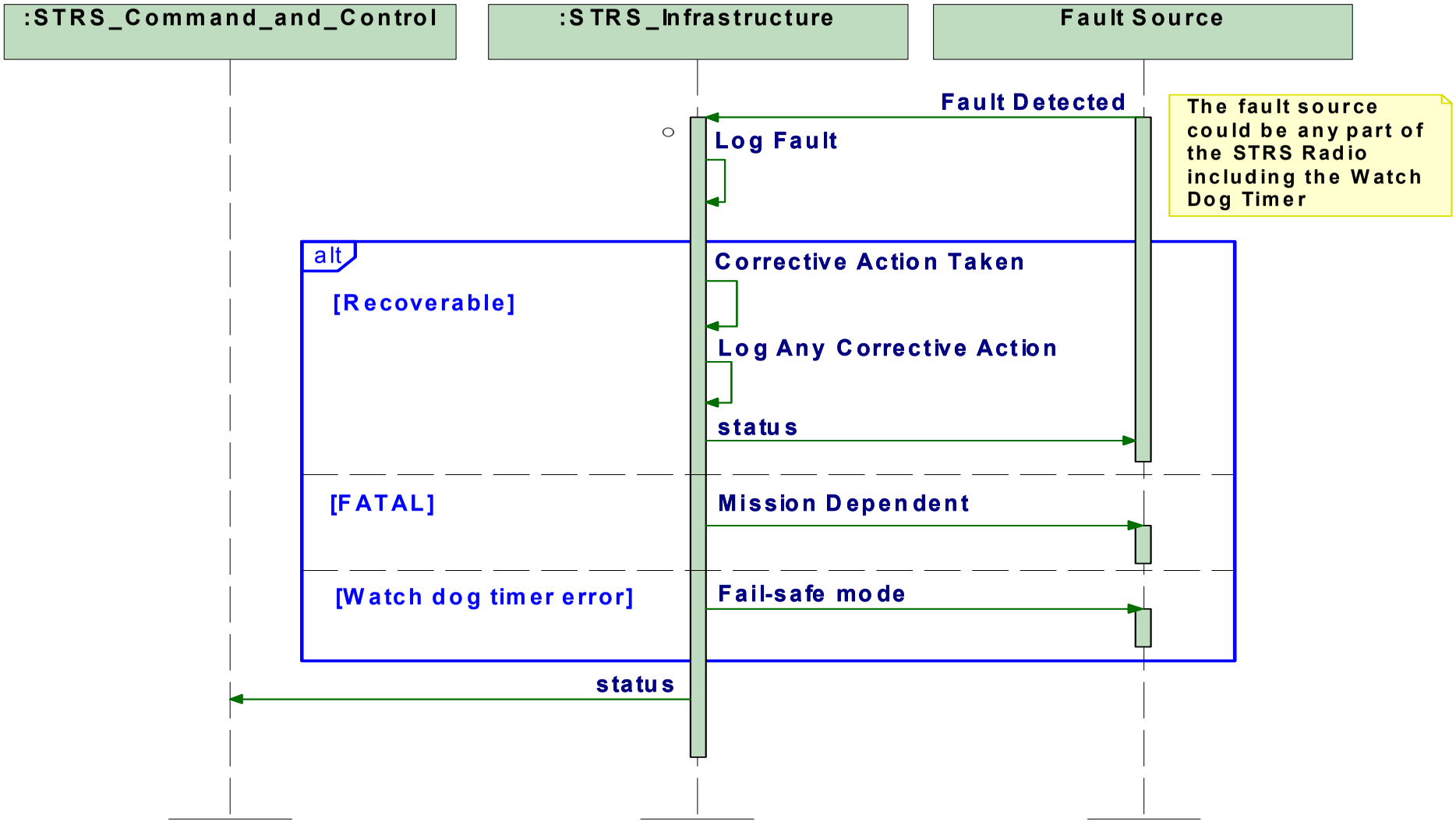
**Consequences:**

- Increased complexity.
- Good scalability to complex systems and complex recovery.
- Good coverage of safety



# Fault Management Topic

## STRS Fault Management Sequence Diagram



# SWaP Verification Topic

## SWaP Verification Techniques

- Approach:
  - Apply Distributed and Memory Management Patterns and benchmark techniques (applied in the final phase) to test and verify NASA STRS SWaP limited Resource Constraints
- Distributed Real-Time Design Patterns [Bruce Douglass]
  - Shared Memory Pattern
    - With or Without INTEGRITY MemoryRegions API
    - **With INTEGRITY POSIX Shared Memory API**
  - Remote Method Call, Observer, Data Bus, Proxy, Broker
- Memory Management Real-Time Design Patterns [Bruce Douglass]
  - Memory Management, Static Allocation, Pool Allocation, Fixed-Sized Buffer, Smart Pointer, Garbage Collection, Garbage Compactor
- **Shared Libraries**

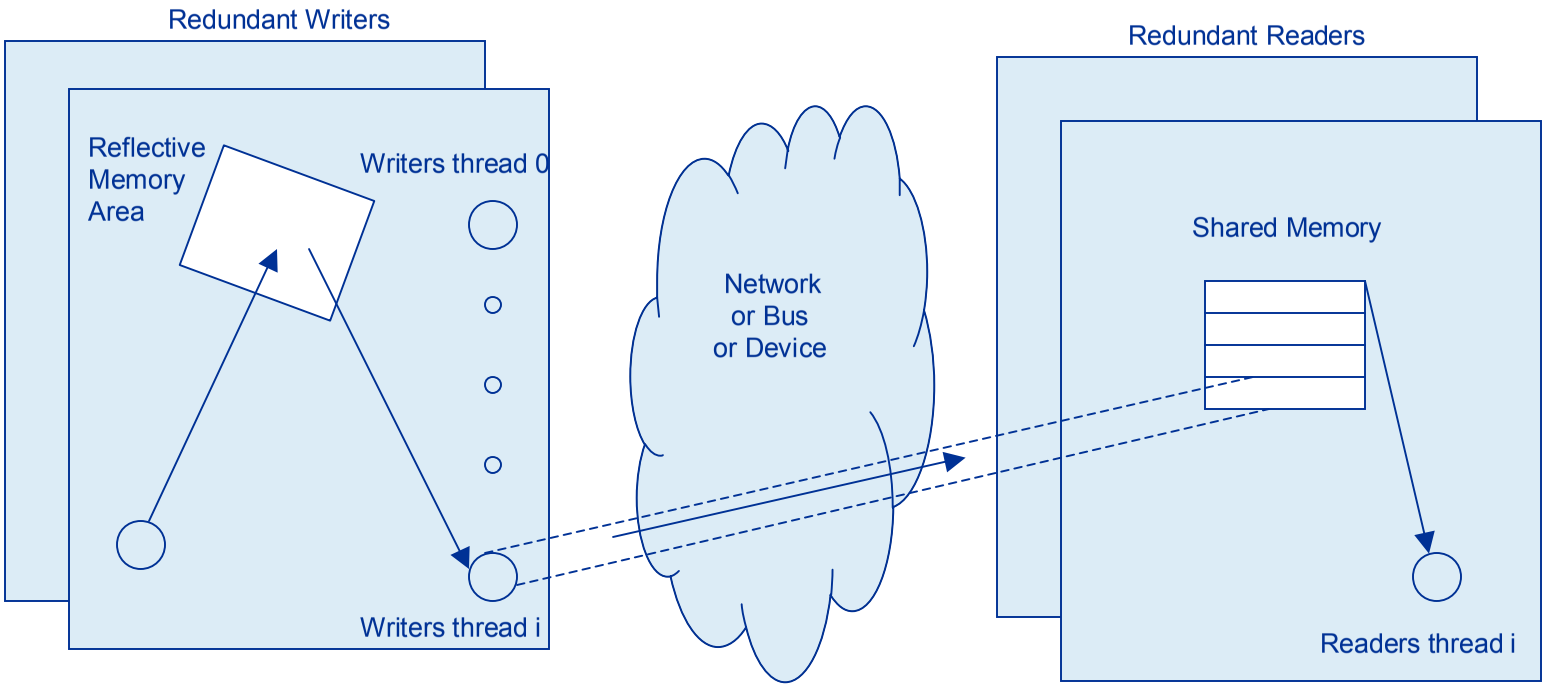
Studied in Initial Phase, Diagram Follows

Many Distributed and Memory Management Design Patterns Available

Commonly Used Technique, Detail Follows

# SWaP Verification Topic

## Shared Memory Pattern
## INTEGRITY POSIX Shared Memory API Example




INTEGRITY POSIX shared memory API can be used to allow two sets of readers and writers to share memory using independent pools.

- ***Efficiently managing memory is key to managing STRS SWaP.***

# SWaP Verification Topic

## Shared Library

- Shared libraries can provide a large memory savings by enabling the multiple virtual address spaces in a system to share the same code.
  - Uses a single code space image with independent data spaces for each process/thread

# Performance Monitor and Benchmark Service Topic

## Benchmark Technique Preliminary Plans

- Benchmark Technique Definition
  - Interim Phase Work
- Benchmark Testing
  - Final Phase Work
- Multi Purpose
  - Performance Knowledge Capture
    - Memory Budgets
    - Task Timing Models
    - Throughput Statistics
    - Failure Tolerance Models
    - Etc.
  - Drives Test and Verification Plans
  - Can be included in STRS Platform and Radio Provider API Guidance Documentation

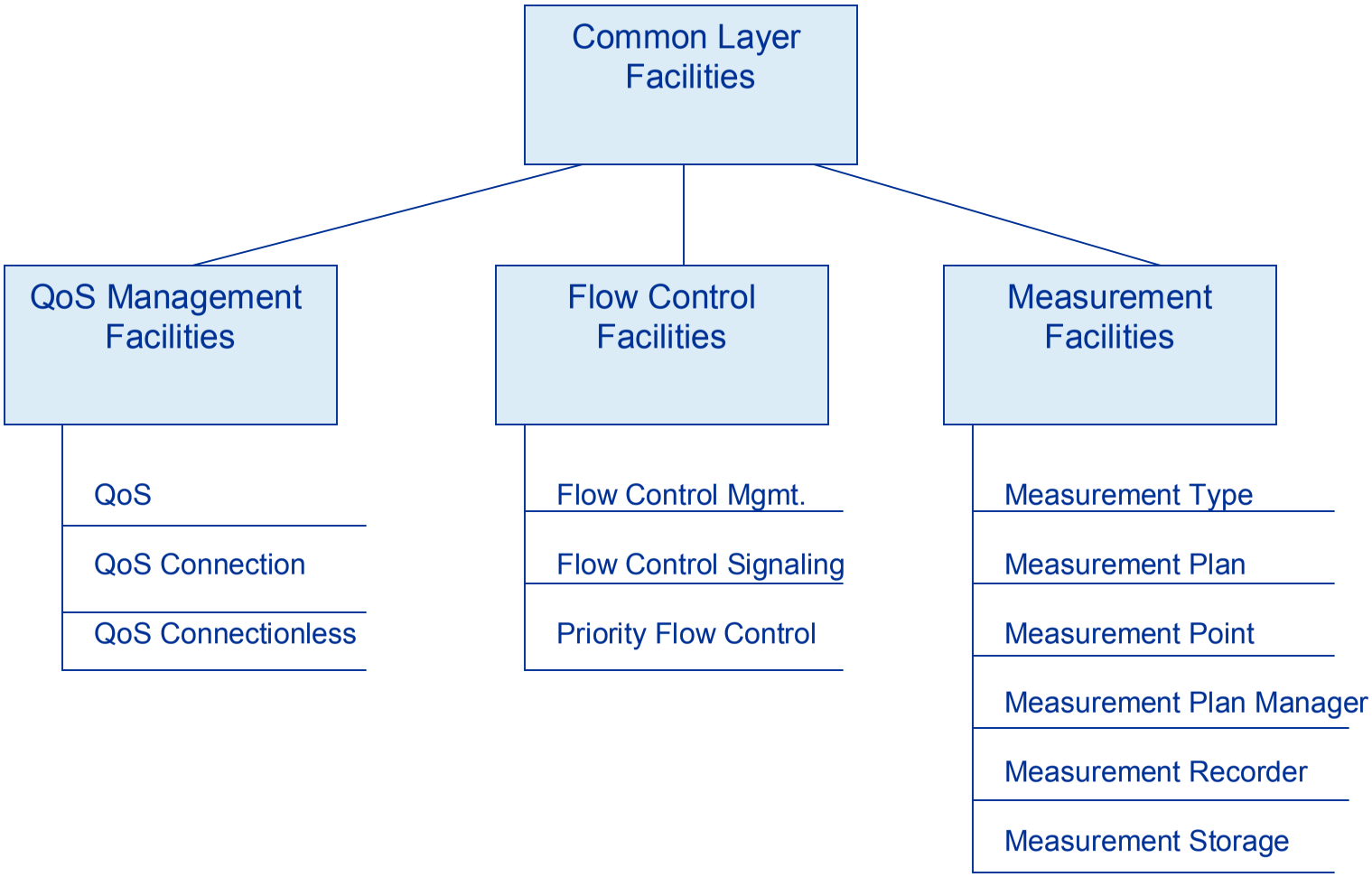
# Mapping to SWRadio PIM/PSM Topic

## SWRadio **PIM**/PSM

- **Component Framework Facilities**
  - Used by majority of radio components. Examples: waveform, device, and platform management interfaces.
- **Physical Layer Facilities**
  - Bidirectional transformation of digitized signals into a propagating RF wave
  - Includes frequency tuning, filters, interface cancellation, analog digital conversion, up/down conversion, gain control, synthesizer, baseband I/O support etc.
- **Radio Set/*Control* Facilities**
  - Manages the radio domain and channels within the radio.
- **Common Radio/*Platform* Facilities**
  - Used by majority of radio components. Examples: logging, naming, and event services.
- **Common Layer Facilities**
  - Used by majority of radio components. Examples: flow control, packet, and stream interfaces.
- **Data Link Layer Facilities**
  - Composed of Link Layer Control (LLC) and Media Access Control (MAC) layer functionality for communication needs.
- **PSM** – Transformation.

# Mapping to SWRadio PIM/PSM Topic

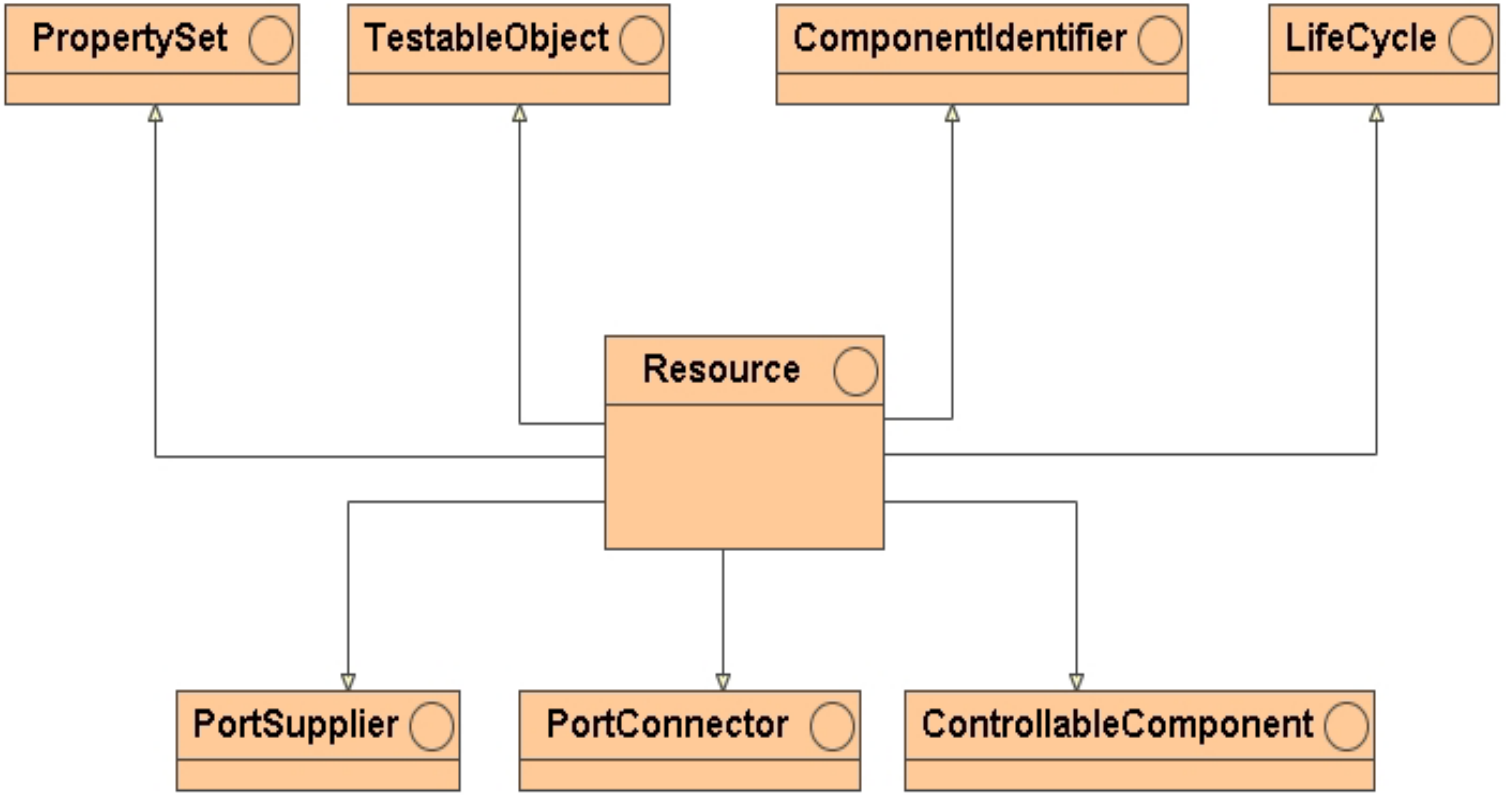
## Real-Time Waveform Control with SDR for Space **PIM**/PSM Applied



# Mapping to SWRadio PIM/PSM Topic

## Resource Management with SDR for Space PIM/**PSM** Applied

- Resource Infrastructure - TestableObject (BIT)
  - Observer/Publish-Subscribe/Delegation Event Design Pattern [GoF]
  - Observer Real-Time Design Distributed Pattern [Bruce Douglass]



# Mapping to SWRadio PIM/PSM Topic

## Initial Phase Study Results SDR for Space PIM/PSM

- Specific Real-Time Design Patterns [Bruce Douglass] and OO Design Patterns [Gang-of-Four (GoF)] **map and correspond** to SDR for Space **PIM** for medium and large NASA SDR mission classes.
- POSIX and RTOS specific Real-Time techniques **map and correspond** to NASA SDR **PSM** for medium and large NASA SDR mission classes.
- RTOS specific Real-Time techniques, that require minimum resources, **map and correspond** to NASA SDR **PSM** for small NASA SDR mission classes.
- SDR for Space PSM **extensions need to be developed** to safeguard Radiation Suitable Processing requirements.
- Further analysis needed to map the following requirements to the SWRadio PIM Component Framework
  - Reliability (fault tolerance, guaranteed delivery) and Availability
    - Fault Management
  - SSP Abstraction
    - HAL

# Summary and Conclusion

- STRS mission systems must use very limited processor and memory resources optimally to meet mission-critical and high reliability requirements.
- Design patterns will continue to be leveraged as applicable to STRS requirements.
- PIM/PSM extensions will continue to be refined in the interim and final prototype phases.
- The Interim phase will focus on integrating STRS APIs together with OMG's PIM and PSM for SWRadio
- The Final phase will use the integrated STRS/SWRadio to implement WFs, integrate WF designs with the matured prototype, conduct review, and incorporate into formal documentation.

**Thanks for your interest!**