NASA/TM—2012-217819

# Space Telecommunications Radio System (STRS) Architecture

## Tutorial Part 1—Overview

*Louis M. Handler and Janette C. Briones*
*Glenn Research Center, Cleveland, Ohio*

*Dale J. Mortensen*
*ASRC Aerospace Corporation, Cleveland, Ohio*

*Richard C. Reinhart*
*Glenn Research Center, Cleveland, Ohio*

December 2012

# NASA STI Program . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI Program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NASA Aeronautics and Space Database and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include creating custom thesauri, building customized databases, organizing and publishing research results.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at *http://www.sti.nasa.gov*

- E-mail your question to *help@sti.nasa.gov*

- Fax your question to the NASA STI Information Desk at 443–757–5803

- Phone the NASA STI Information Desk at 443–757–5802

- Write to:
  STI Information Desk
  NASA Center for AeroSpace Information
  7115 Standard Drive
  Hanover, MD 21076–1320

NASA/TM—2012-217819

# Space Telecommunications Radio System (STRS) Architecture

## Tutorial Part 1—Overview

*Louis M. Handler and Janette C. Briones*
*Glenn Research Center, Cleveland, Ohio*

*Dale J. Mortensen*
*ASRC Aerospace Corporation, Cleveland, Ohio*

*Richard C. Reinhart*
*Glenn Research Center, Cleveland, Ohio*

National Aeronautics and
Space Administration

Glenn Research Center
Cleveland, Ohio 44135

December 2012

Trade names and trademarks are used in this report for identification only. Their usage does not constitute an official endorsement, either expressed or implied, by the National Aeronautics and Space Administration.

*Level of Review*: This material has been technically reviewed by technical management.

Available from

NASA Center for Aerospace Information
7115 Standard Drive
Hanover, MD 21076–1320

National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Available electronically at http://www.sti.nasa.gov

# Space Telecommunications Radio System (STRS) Architecture
## Tutorial Part 1—Overview

Louis M. Handler and Janette C. Briones
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Dale J. Mortensen
ASRC Aerospace Corporation
Cleveland, Ohio 44135

Richard C. Reinhart
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

## Abstract

Space Telecommunications Radio System (STRS) Architecture Standard provides a NASA standard for software-defined radio. STRS is being demonstrated in the Space Communications and Navigation (SCaN) Testbed formerly known as Communications, Navigation and Networking Configurable Testbed (CoNNeCT). Ground station radios communicating the SCaN testbed are also being written to comply with the STRS architecture. The STRS Architecture Tutorial Overview presents a general introduction to the STRS architecture standard developed at the NASA Glenn Research Center (GRC), addresses frequently asked questions, and clarifies methods of implementing the standard. The STRS architecture should be used as a base for many of NASA's future telecommunications technologies. The presentation will provide a basic understanding of STRS.

# Space Telecommunications Radio System (STRS) Architecture

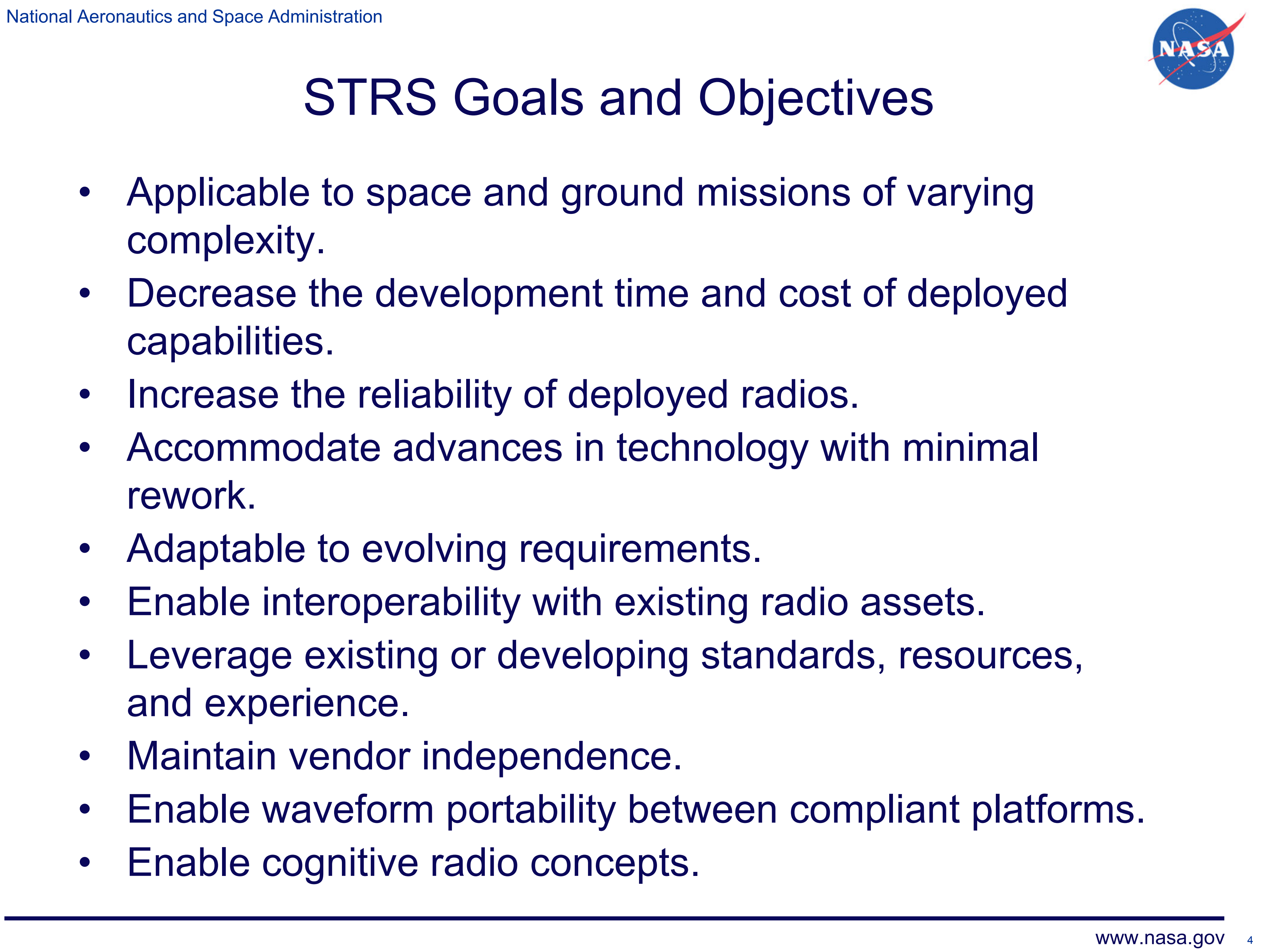
## Tutorial Part 1 - Overview

Glenn Research Center
November 2011

# STRS Architecture

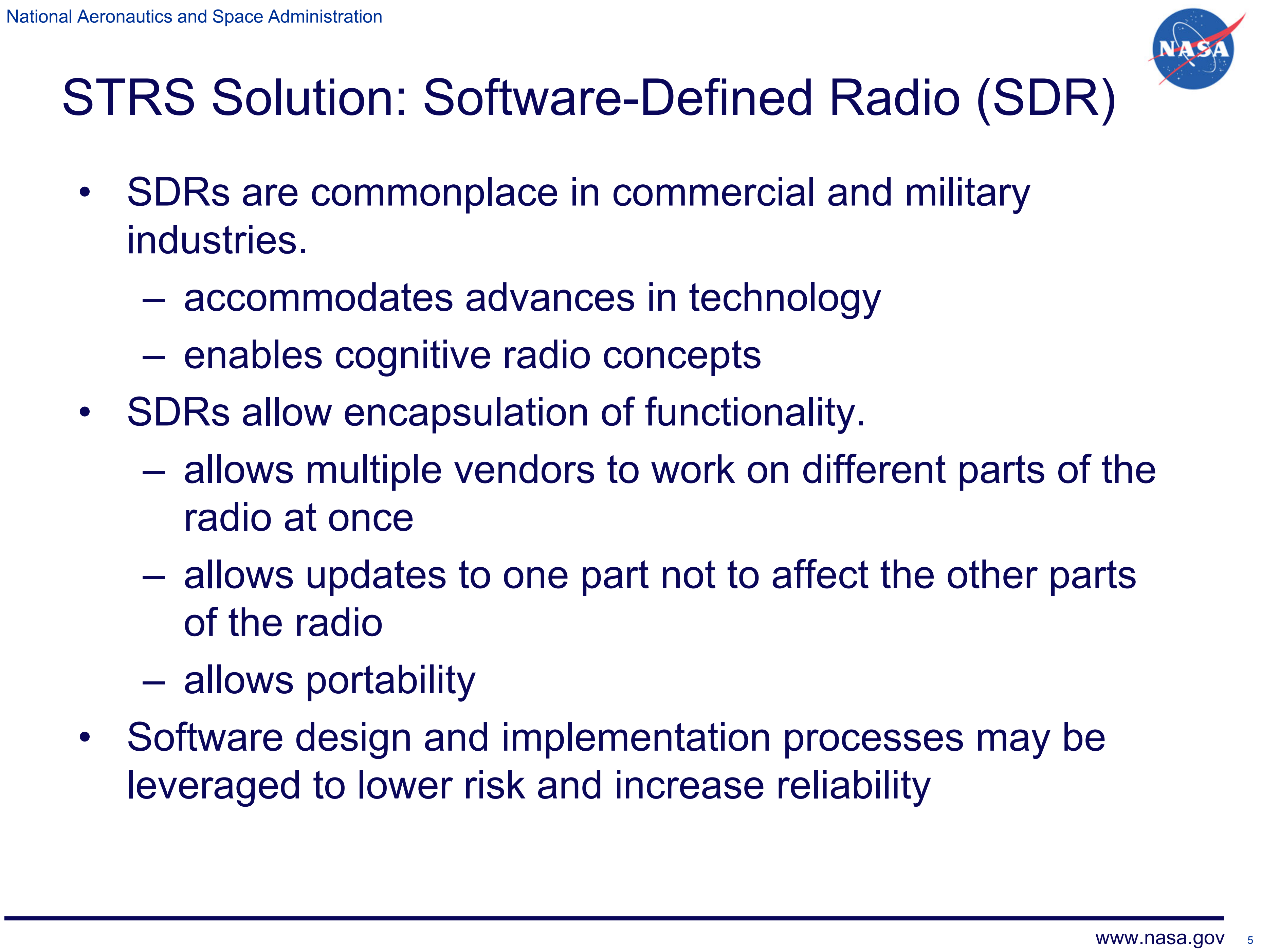
- STRS Background
- STRS Hardware & Software Structure
- STRS Infrastructure APIs
- STRS Application APIs
- STRS Configuration Files
- STRS Reference Documents

# STRS Background



# STRS Goals and Objectives

- Applicable to space and ground missions of varying complexity.
- Decrease the development time and cost of deployed capabilities.
- Increase the reliability of deployed radios.
- Accommodate advances in technology with minimal rework.
- Adaptable to evolving requirements.
- Enable interoperability with existing radio assets.
- Leverage existing or developing standards, resources, and experience.
- Maintain vendor independence.
- Enable waveform portability between compliant platforms.
- Enable cognitive radio concepts.



# STRS Solution: Software-Defined Radio (SDR)

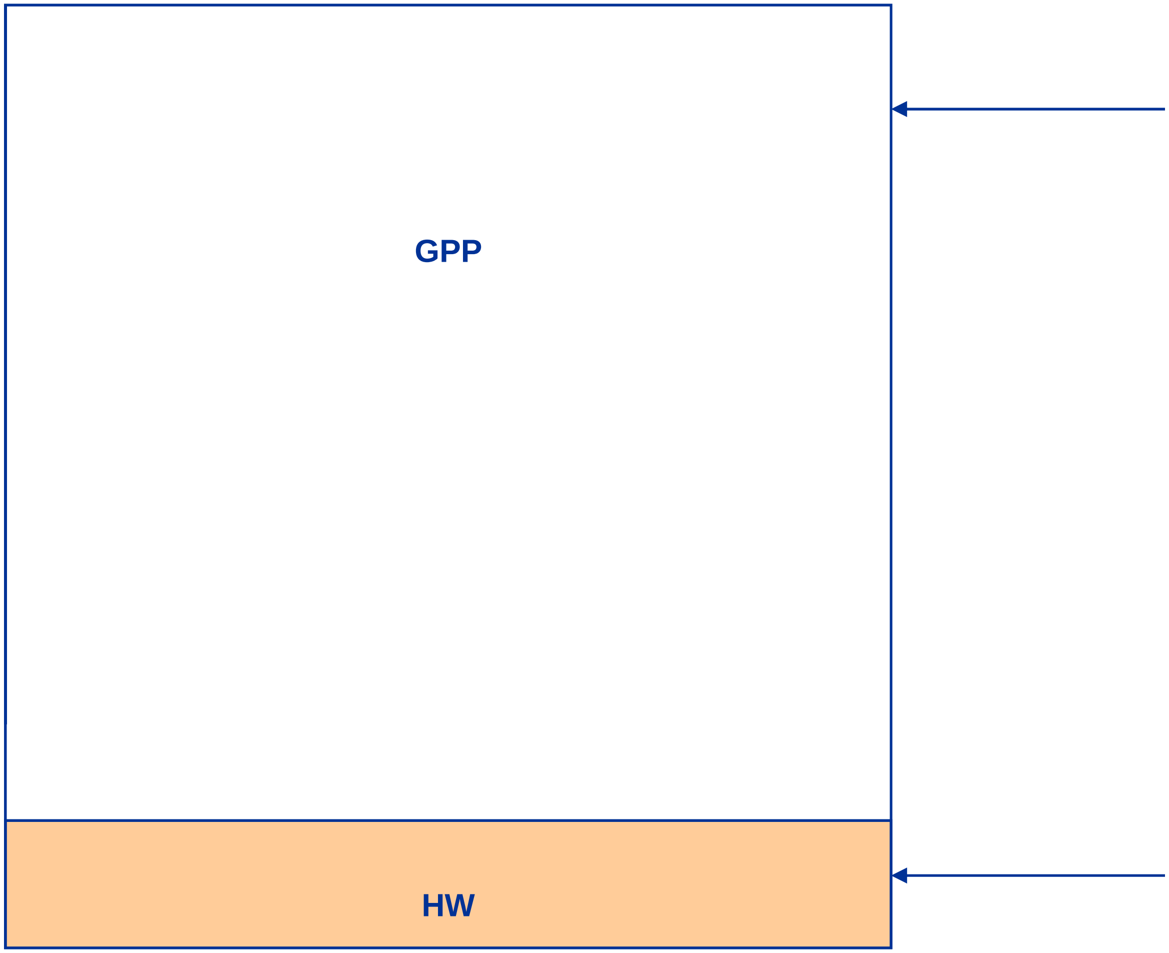
- SDRs are commonplace in commercial and military industries.
  - accommodates advances in technology
  - enables cognitive radio concepts
- SDRs allow encapsulation of functionality.
  - allows multiple vendors to work on different parts of the radio at once
  - allows updates to one part not to affect the other parts of the radio
  - allows portability
- Software design and implementation processes may be leveraged to lower risk and increase reliability

# STRS Background

- SDRs present unique challenges in space.
  - Radiation environment
  - Temperature extremes
  - Autonomous operation
  - Size, weight, and power (SWaP) limitations
  - Timescale of deployments
  - Lengthy development cycles
- JTRS/SCA and OMG/SWRADIO were investigated
  - including CORBA was too cumbersome due to SWaP
  - including an XML parser was too cumbersome due to SWaP
  - SCA’s XML configuration files were too complex for our needs
  - Used Platform Independent Model (PIM) as a starting point for STRS API design
- Decided to allow a C language interface to minimize SWaP

# STRS Hardware and Software Structure

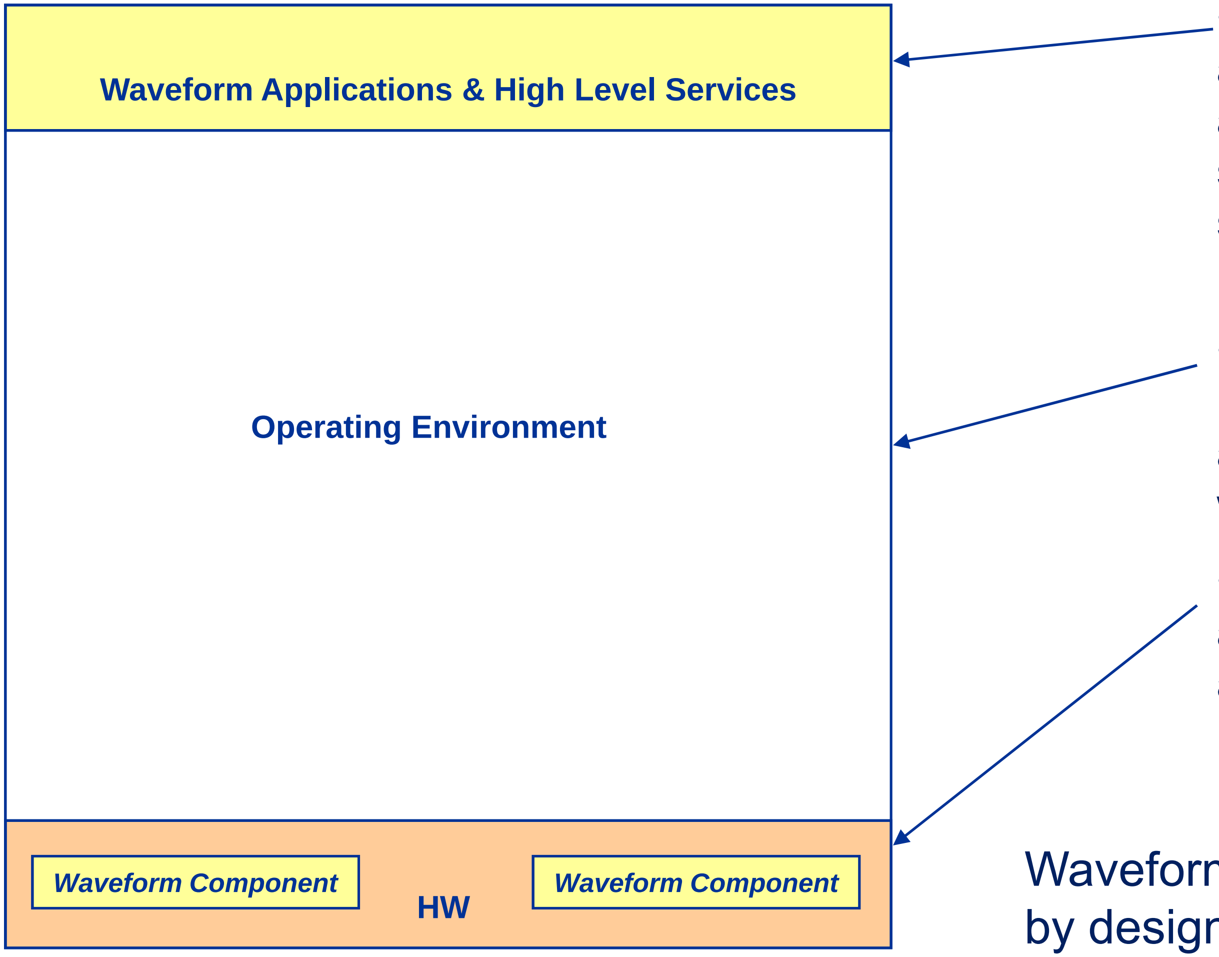
# SDR Signal Processing Hardware



- General Purpose Processor (GPP) typically contains and executes the Managing Software enabling Software Defined Radio functionality

- Specialized Hardware contains and executes Application Software (i.e. firmware) enabling higher rate processing within the Software Defined Radio (e.g. FPGA)

# Waveform Application Decomposition



- Low rate waveform application components allocated to processor based signal processor. High level services used to control and monitor application software

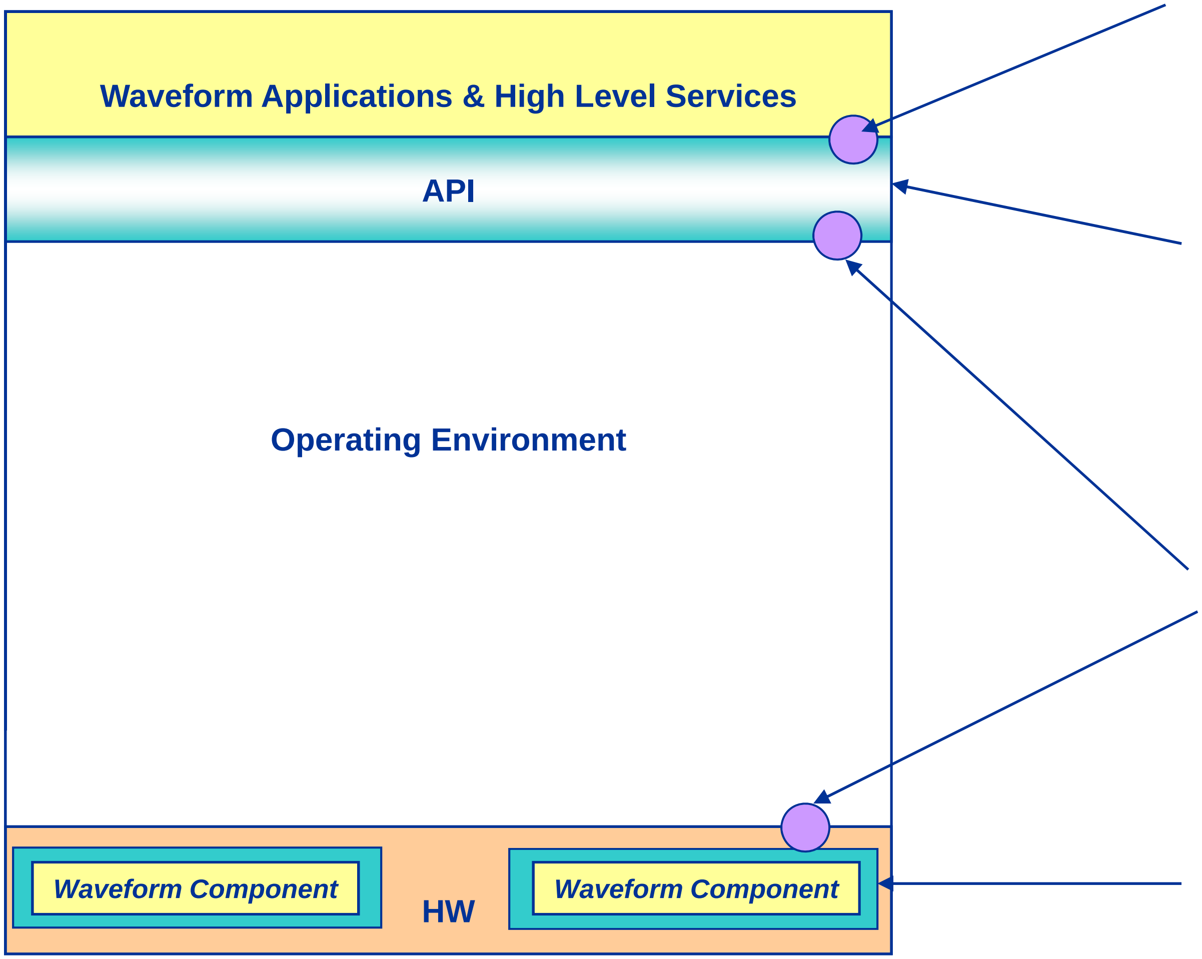
- Operating Environment (OE) provides access to processor and specialized hardware for waveform processing

- High rate waveform application components allocated to specialized processing hardware

Waveform decomposition driven by designers, not architecture

# STRS Open Architecture

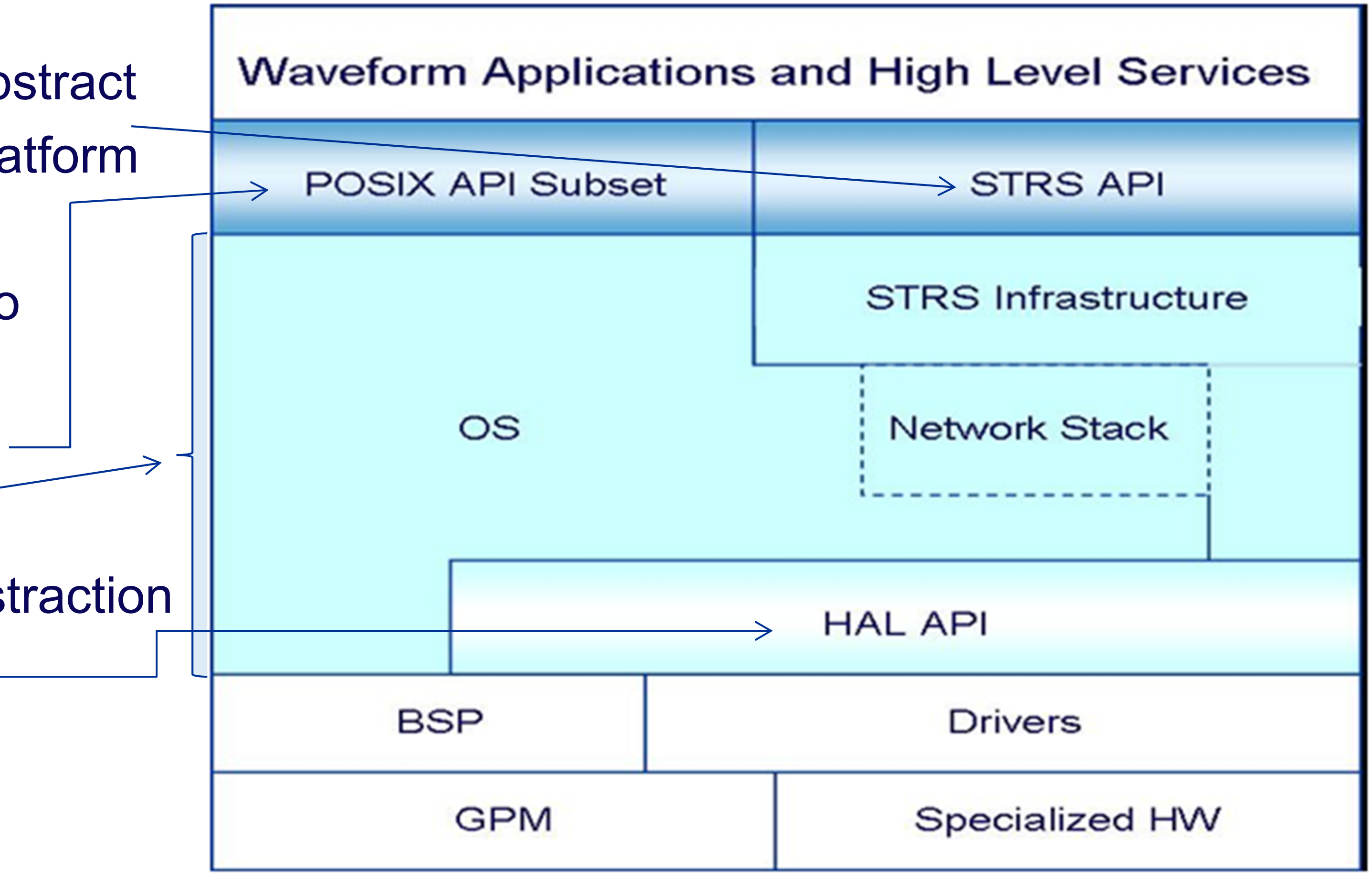
## Waveform Application API and Hardware Abstraction



- Waveform designers access specified API to access processor and implements radio controls in the GPP
- APIs specified by architecture separate the waveform from the Operating Environment for waveform portability
- Operating Environment provides published interfaces (API) services and hardware abstraction control to waveform
- Hardware Abstraction Layer (HAL) also provides wrapper to help port HDL software among platforms

# STRS Architecture

- Layer cake model
  - Waveform applications and high level services are insulated from OE by APIs
  - STRS APIs abstract away many platform differences
  - POSIX used to reduce API development
  - OE
  - Hardware Abstraction Layer (HAL)



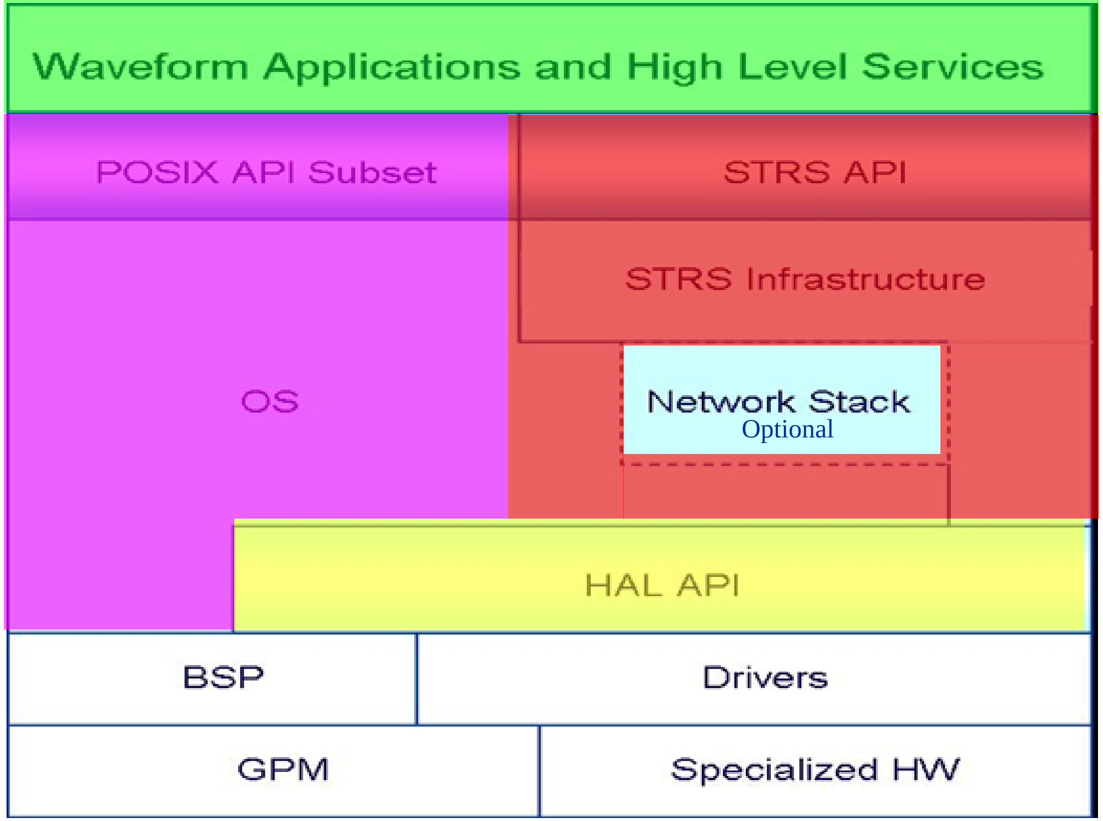
# STRS Architecture Conformance

🟩 Conformant to STRS Architecture Standard requirements for applications

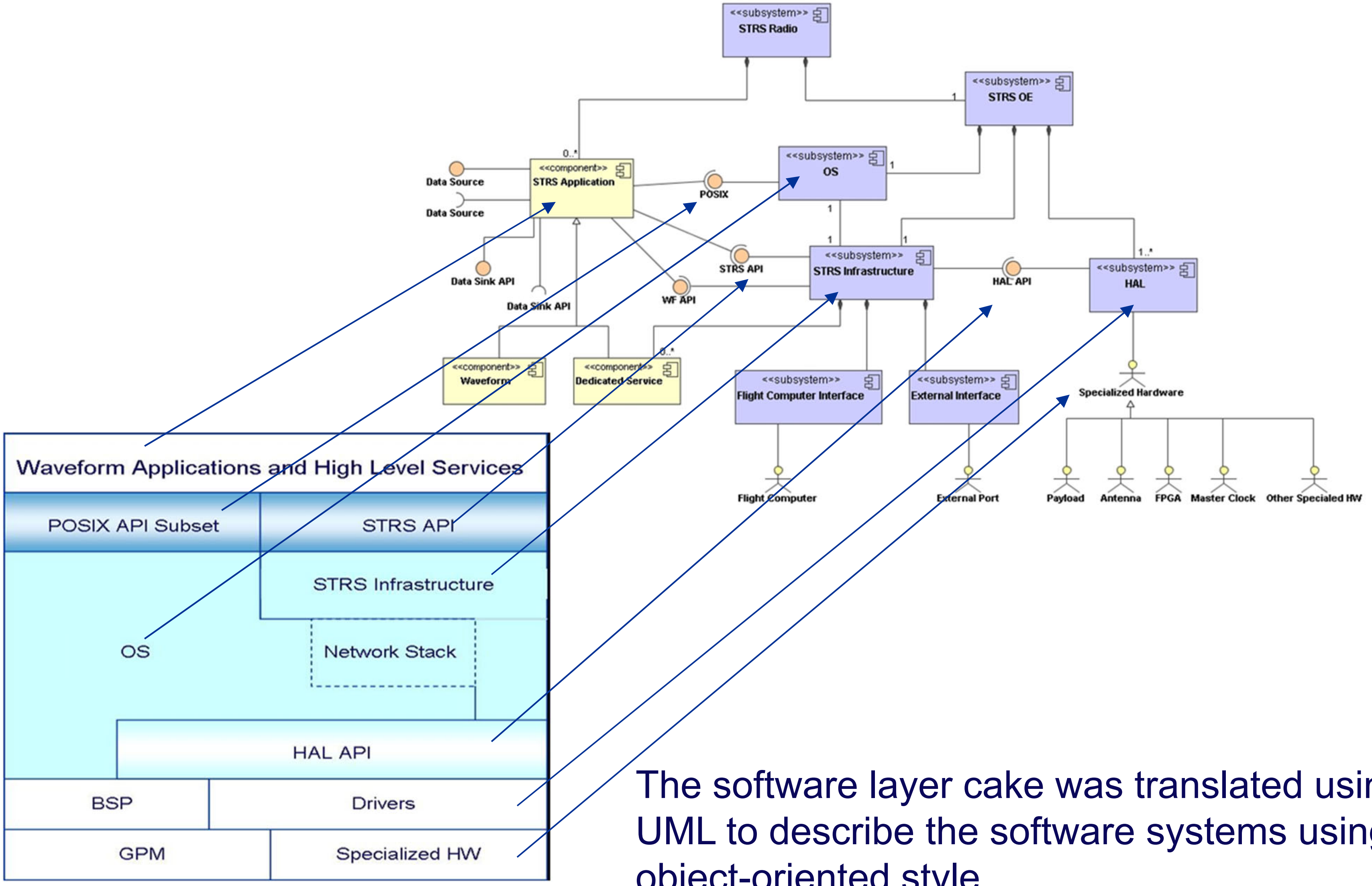
🟥 Conformant to STRS Architecture Standard (STRS and WF APIs)

🟪 Compliant to POSIX PSE51 or Subset with Waiver

🟨 Documented HAL and HID as required by STRS Architecture Specification

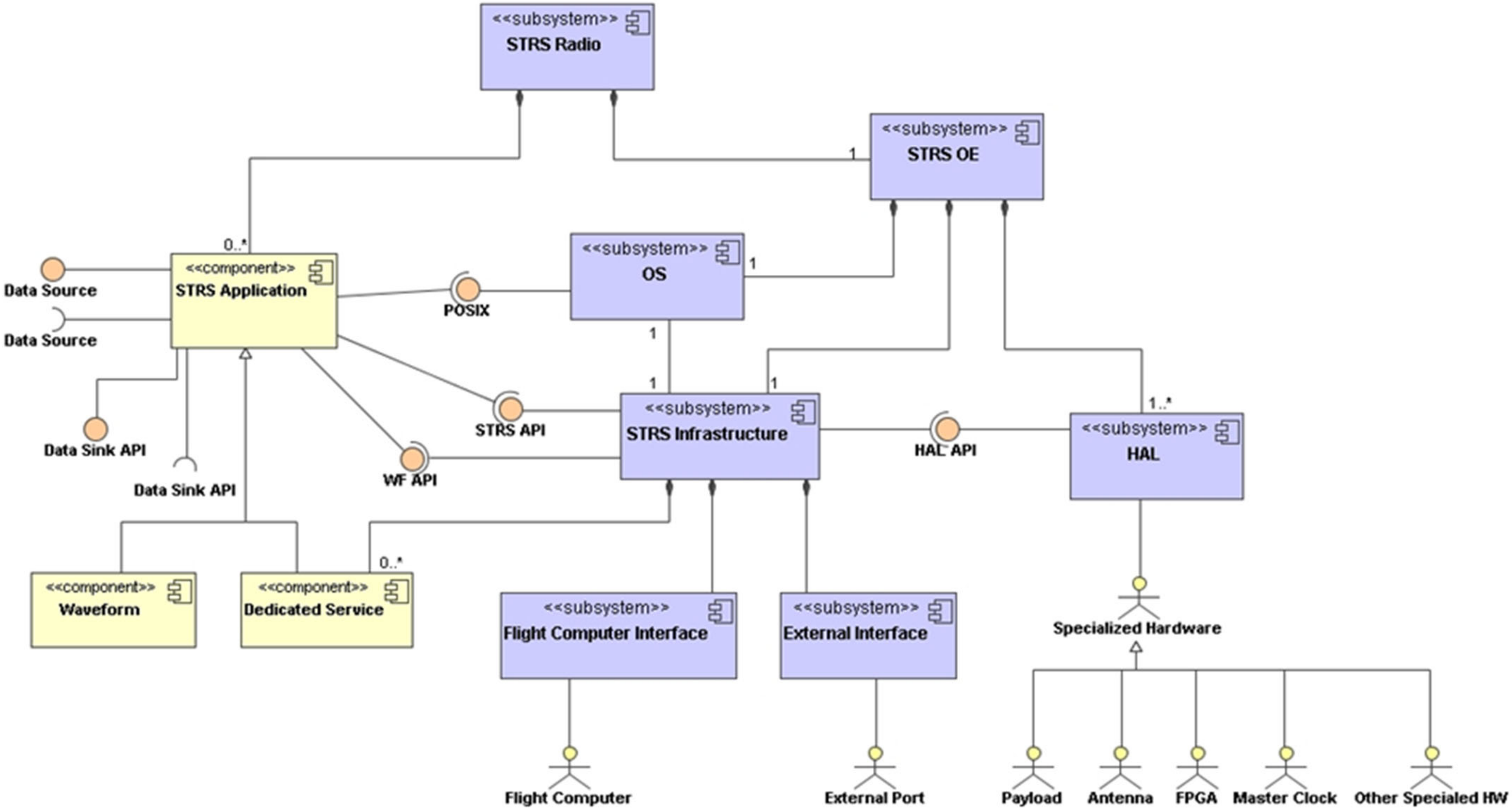


# Layer Cake Transition to UML



The software layer cake was translated using UML to describe the software systems using object-oriented style.

# STRS Layered Structure



# STRS Infrastructure APIs

# STRS Infrastructure APIs

- STRS Infrastructure APIs are used:
  - Waveform calls methods in Infrastructure.
  - Infrastructure calls appropriate method in another Waveform, Device, or Infrastructure.
- Purpose:
  - Methods separate a request from the accomplishment of that request.
  - Methods are ‘extern “C”’ so that they can be called from either C or C++.
  - Methods insulate waveforms from having to know how another waveform, device or the infrastructure is implemented.

# STRS Infrastructure APIs

Queue Control

- STRS_QueueCreate
- STRS_QueueDelete
- STRS_Read
- STRS_Register
- STRS_Log
- STRS_Write
- STRS_Unregister

Device Control

- STRS_DeviceClose
- STRS_DeviceFlush
- STRS_DeviceLoad
- STRS_DeviceOpen
- STRS_DeviceReset
- STRS_DeviceStart
- STRS_DeviceStop
- STRS_DeviceUnload
- STRS_SetISR

Testing

- STRS_RunTest
- STRS_GroundTest

Attribute

- STRS_Configure
- STRS_Query

Process Errors

- STRS_GetErrorQueue
- STRS_IsOK

Control

- STRS_Initialize
- STRS_ReleaseObject
- STRS_Start
- STRS_Stop

Application

- STRS_HandleRequest
- STRS_InstantiateApp
- STRS_AbortApp

Time

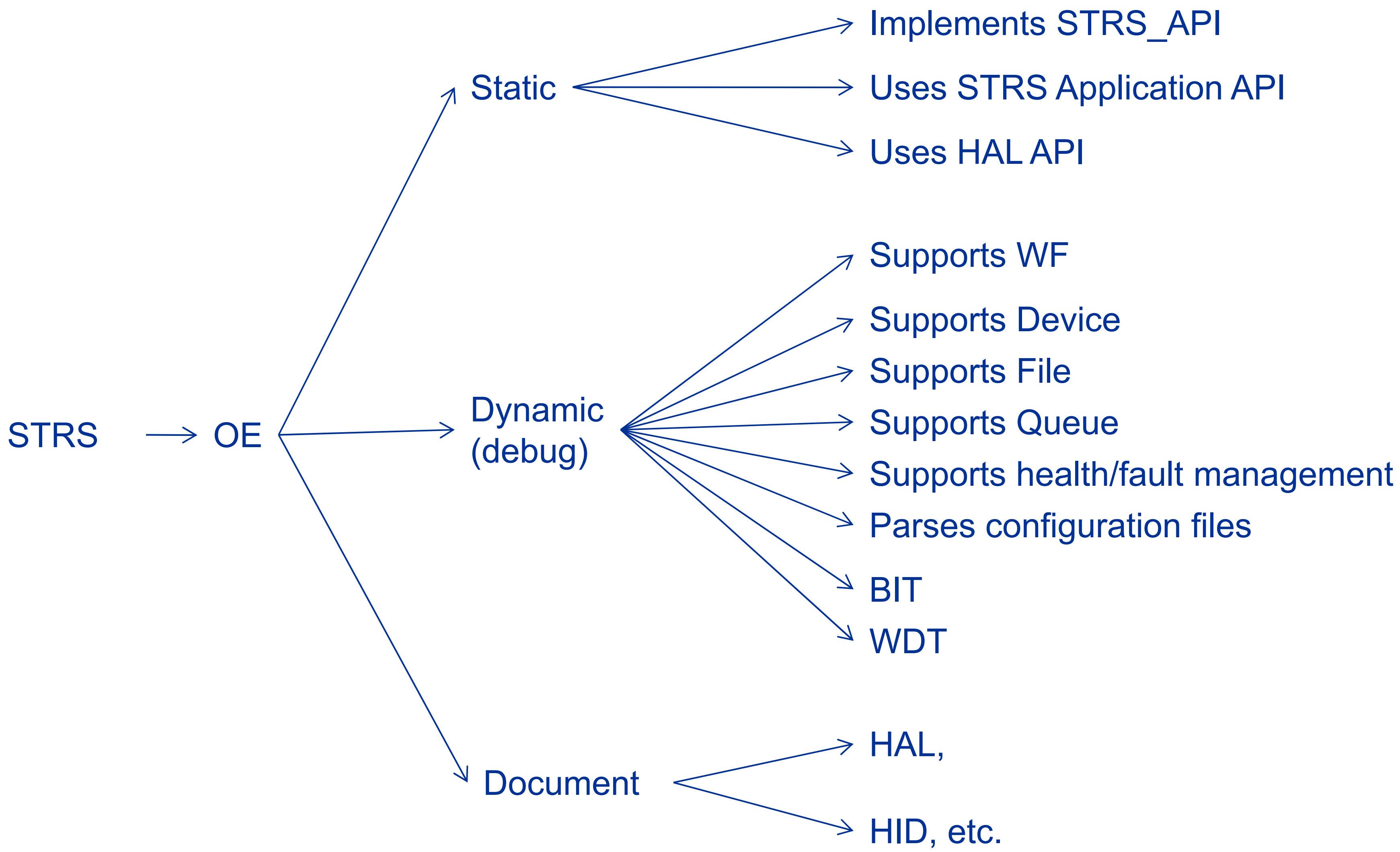
- STRS_GetNanoseconds
- STRS_GetSeconds
- STRS_GetTimeWarp
- STRS_GetTime
- STRS_SetTime
- STRS_Synch

File (Named Area)

- STRS_FileClose
- STRS_FileGetFreeSpace
- STRS_FileGetSize
- STRS_FileOpen
- STRS_FileRemove
- STRS_FileRename

- The STRS Software Architecture presents a consistent set of APIs to allow waveform applications, services, and communication equipment to interoperate in meeting a waveform specification
- These APIs are used in general or to control one waveform from another
- The list to the left is the minimum list of APIs that the STRS platform infrastructure must implement

# STRS OE Compliance



STRS → OE

OE → Static
- Implements STRS_API
- Uses STRS Application API
- Uses HAL API

OE → Dynamic (debug)
- Supports WF
- Supports Device
- Supports File
- Supports Queue
- Supports health/fault management
- Parses configuration files
- BIT
- WDT

OE → Document
- HAL,
- HID, etc.

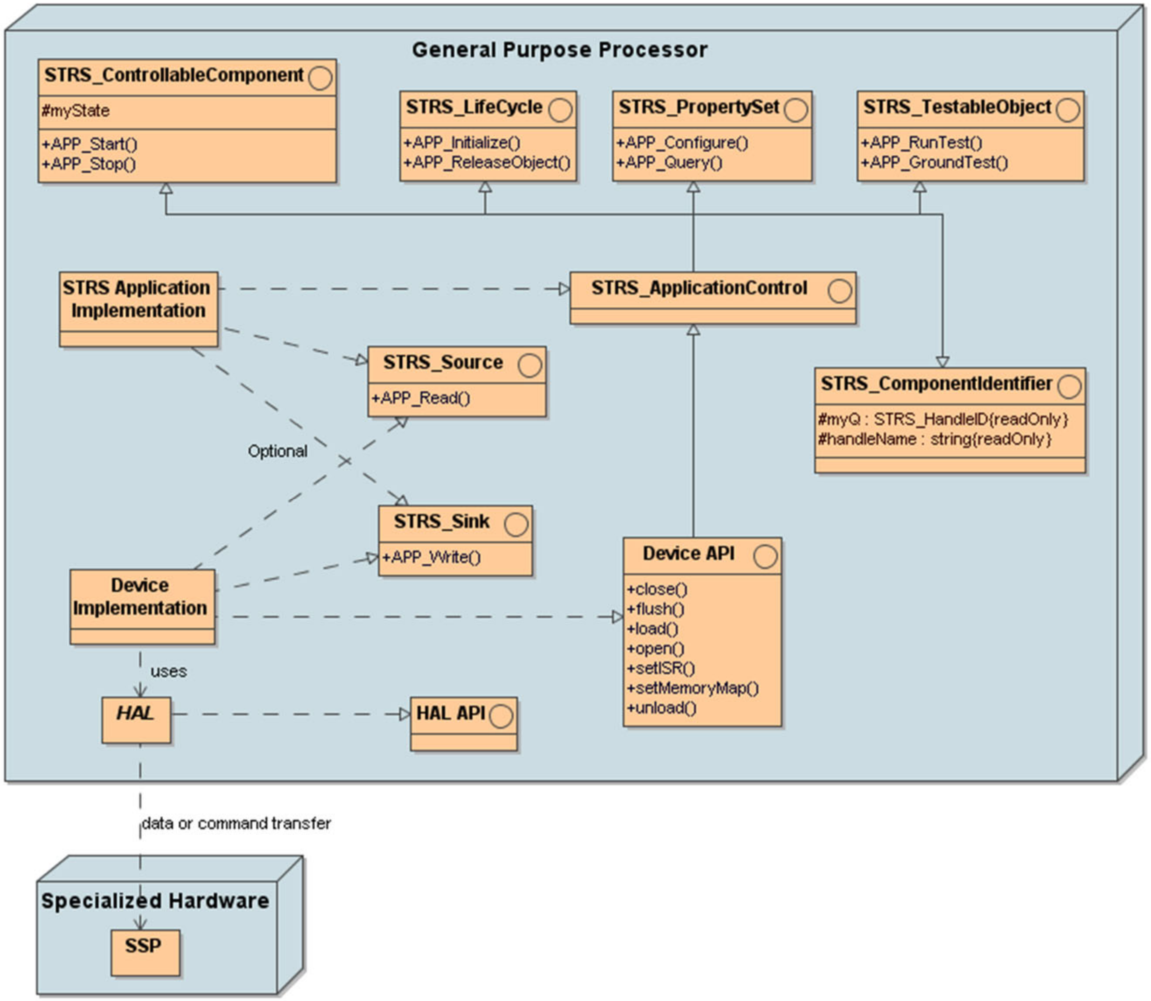
# STRS Waveform APIs

# STRS Waveform Application Compliance

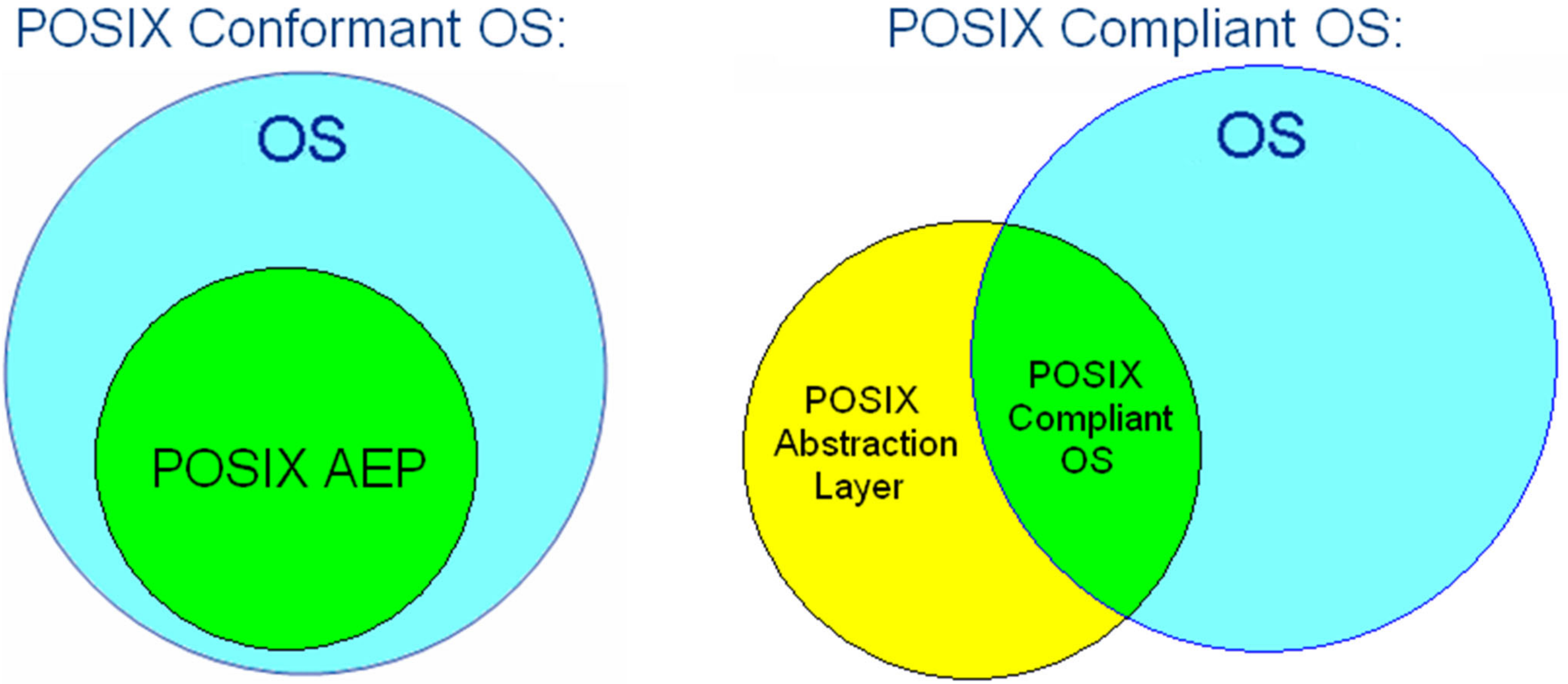
- A waveform is an STRS Application and must implement the APIs shown in the diagram

- An STRS Application has OMG similarity; but STRS requires everything, except source and sink (STRS replaces OMG ports with source/sinks)

- The diagram shows how a Device fits in the infrastructure
    - o Device is internal, must have the shown functionality
    - o Device is an abstraction (proxy) that uses the HAL to get to the hardware
    - o No standard for the HAL API. Standard is at Device level (provider)



# POSIX Compliance/Conformance

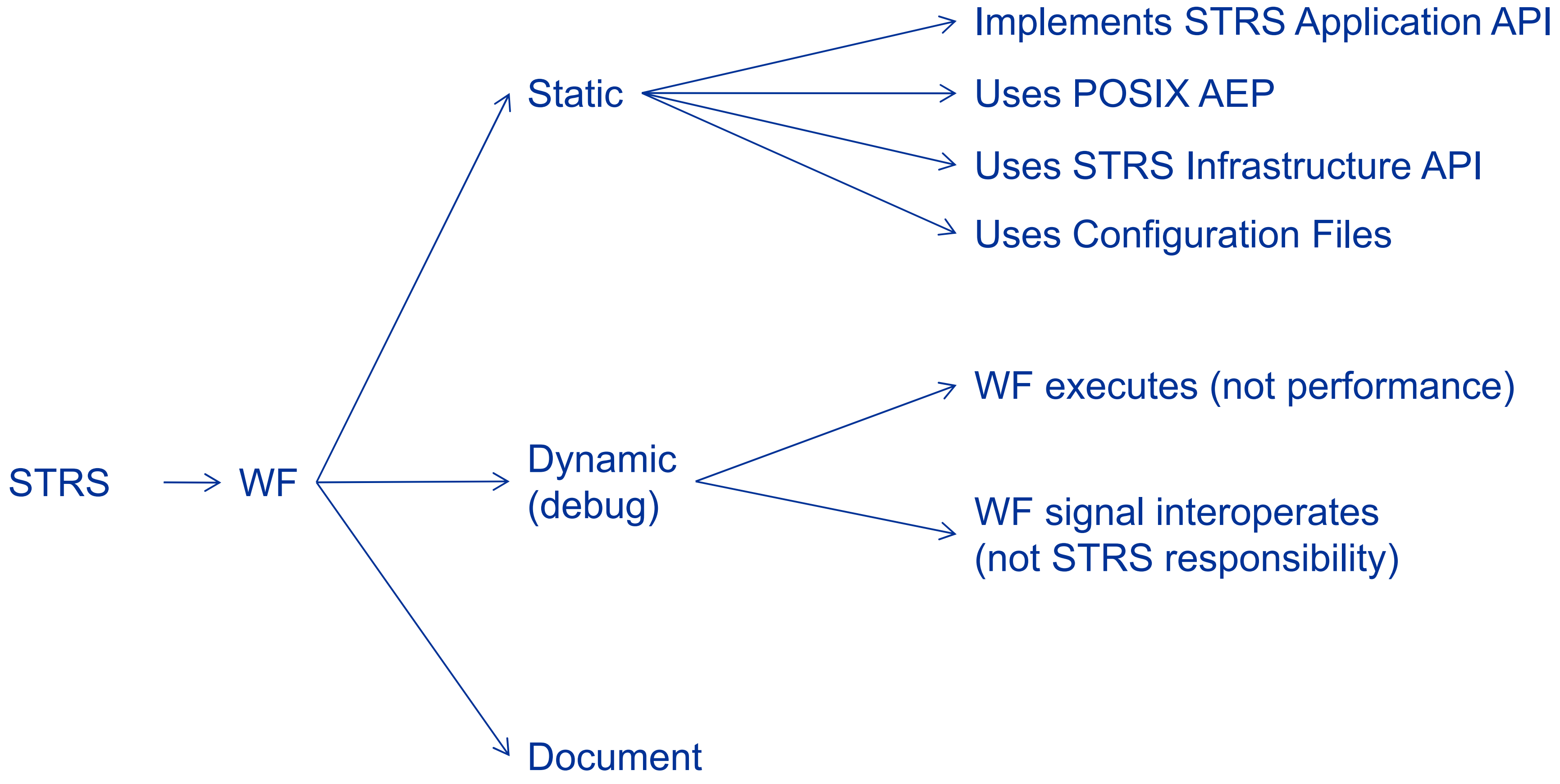


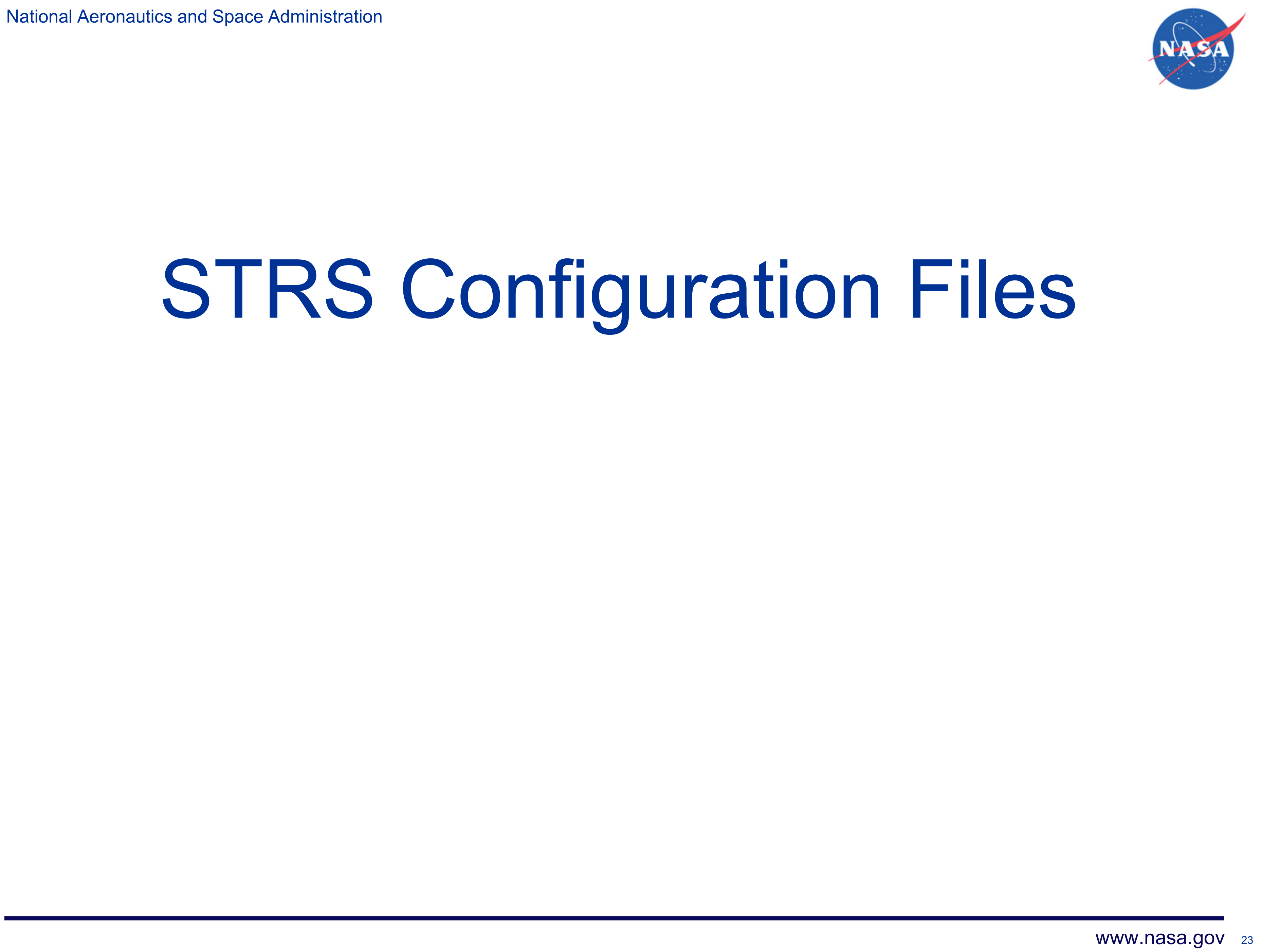
POSIX Conformant OS:

POSIX Compliant OS:

An STRS operating environment can either use an OS that conforms with 1003.13 PSE51 or provide a POSIX abstraction layer that provides missing PSE51 interfaces. For constrained resource platforms, the POSIX requirement is based on waveform requirements so that the **waveforms are upward compatible** (require POSIX methods).

# STRS Waveform Compliance

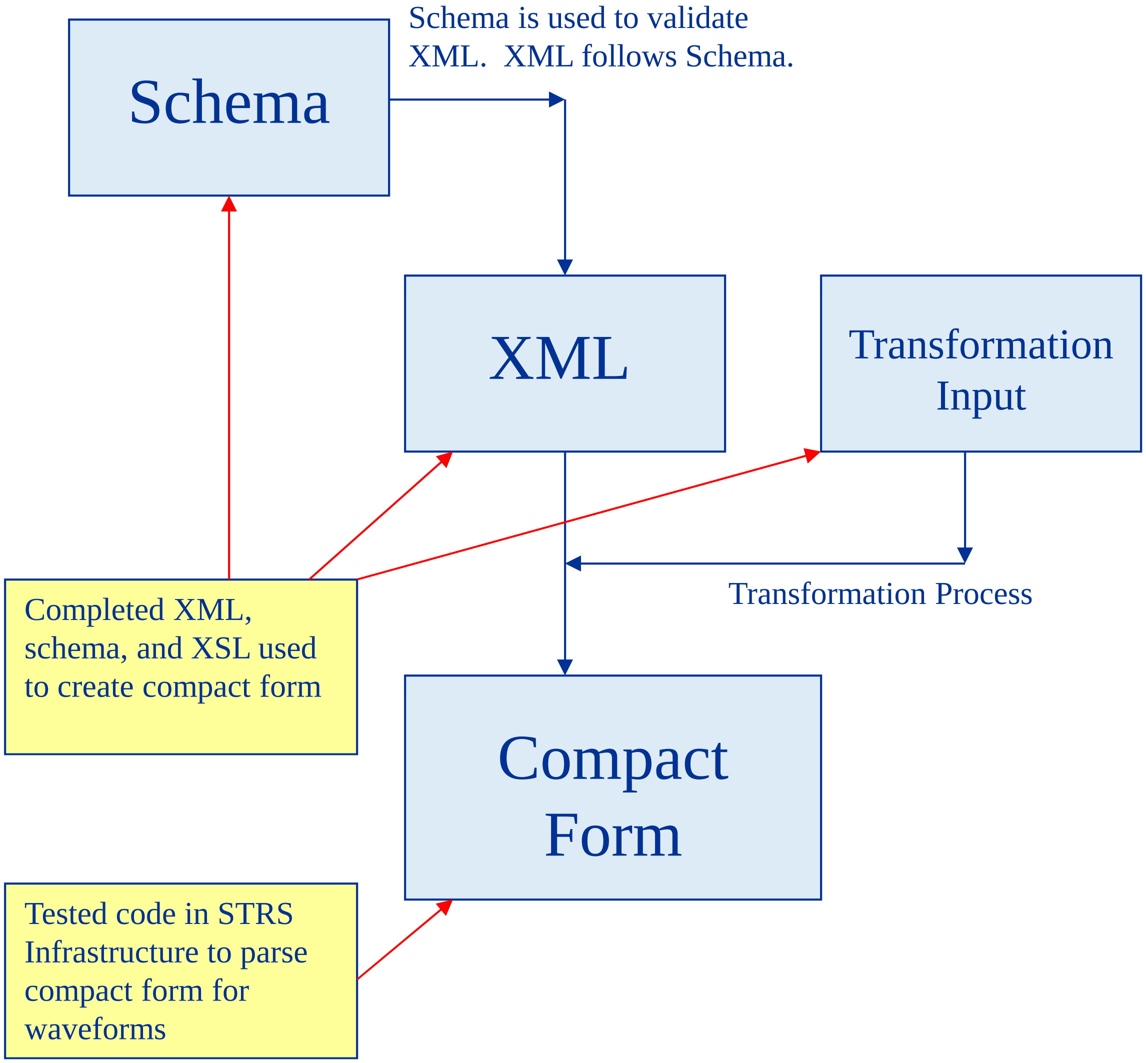

STRS
WF
Static
Dynamic (debug)
Document
Implements STRS Application API
Uses POSIX AEP
Uses STRS Infrastructure API
Uses Configuration Files
WF executes (not performance)
WF signal interoperates (not STRS responsibility)



# STRS Configuration Files

# Configuration Files

- Require schema and XML as part of the architecture
- The required XML should be transformed to a compact format
- The approach for the transformation is not mandated as part of the architecture
- STRS Reference Implementation uses XSL/XSLT to transform XML to an S-expression as compact form


Schema
Schema is used to validate XML.  XML follows Schema.
XML
Transformation Input
Transformation Process
Compact Form
Completed XML, schema, and XSL used to create compact form
Tested code in STRS Infrastructure to parse compact form for waveforms

# STRS Reference Documents

# STRS Reference Documents

- Space Telecommunications Radio System (STRS) Architecture Standard Release 1.02.1, December 2010, NASA TM 2010-216809

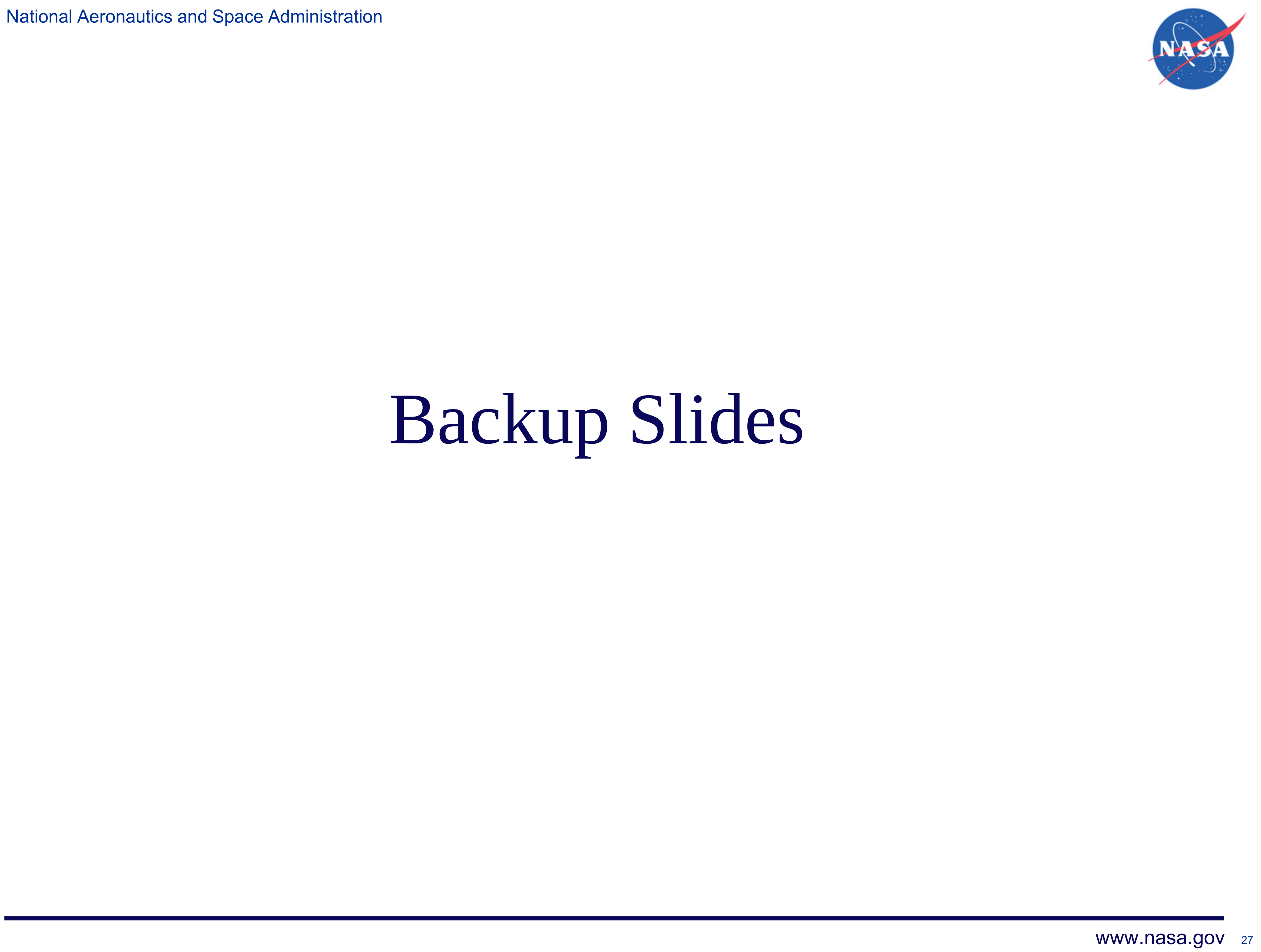
http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20110002806_2011001718.pdf

- Space Telecommunications Radio System (STRS) Architecture Goals/Objectives and Level 1 Requirements Document, June 2007, NASA TM 2007-215042.

http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20080008862_2008008550.pdf

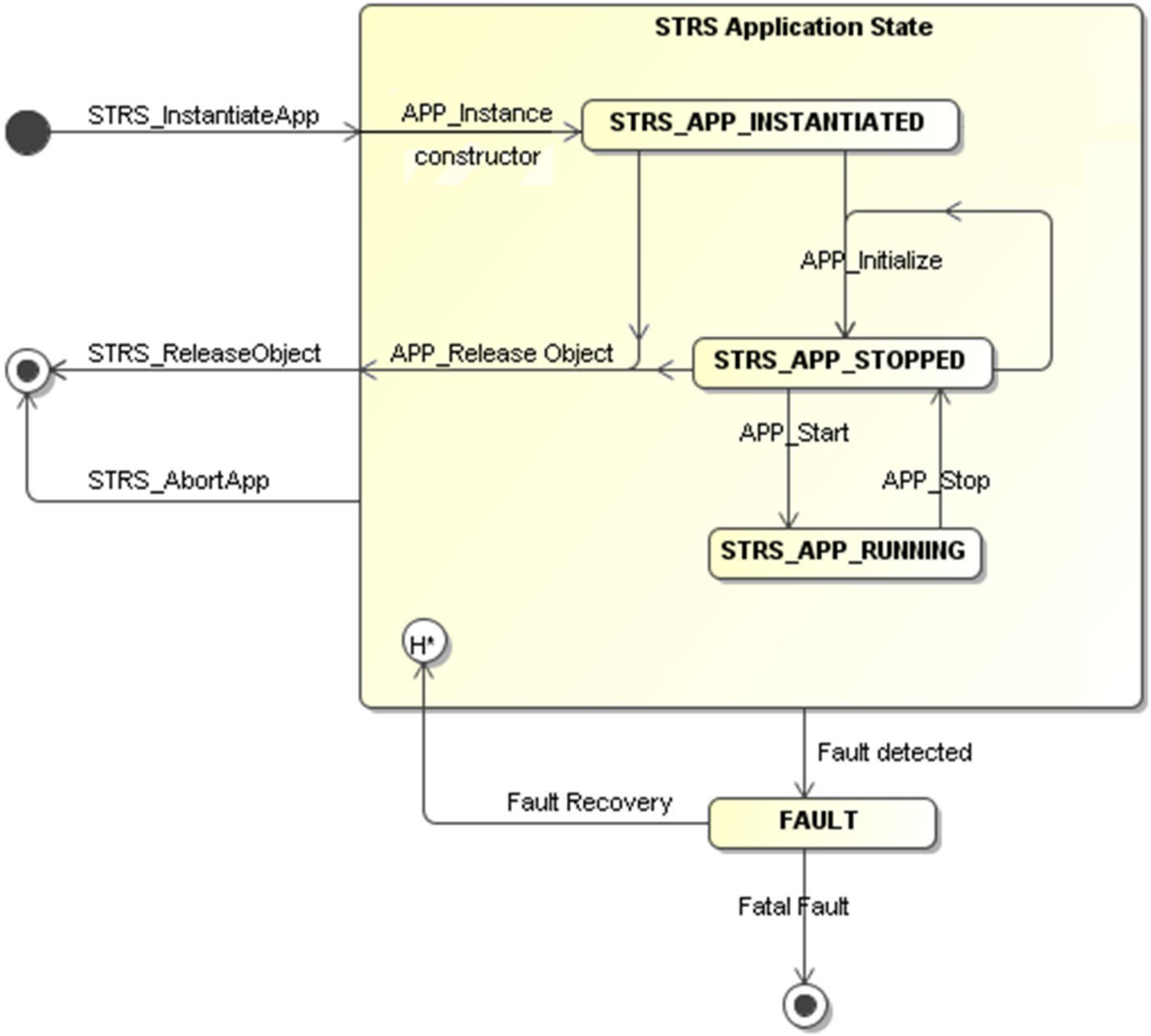
- Space Telecommunications Radio System (STRS) Definitions and Acronyms, May 2008, NASA TM 2008-215445.

http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20090005977_2009004914.pdf

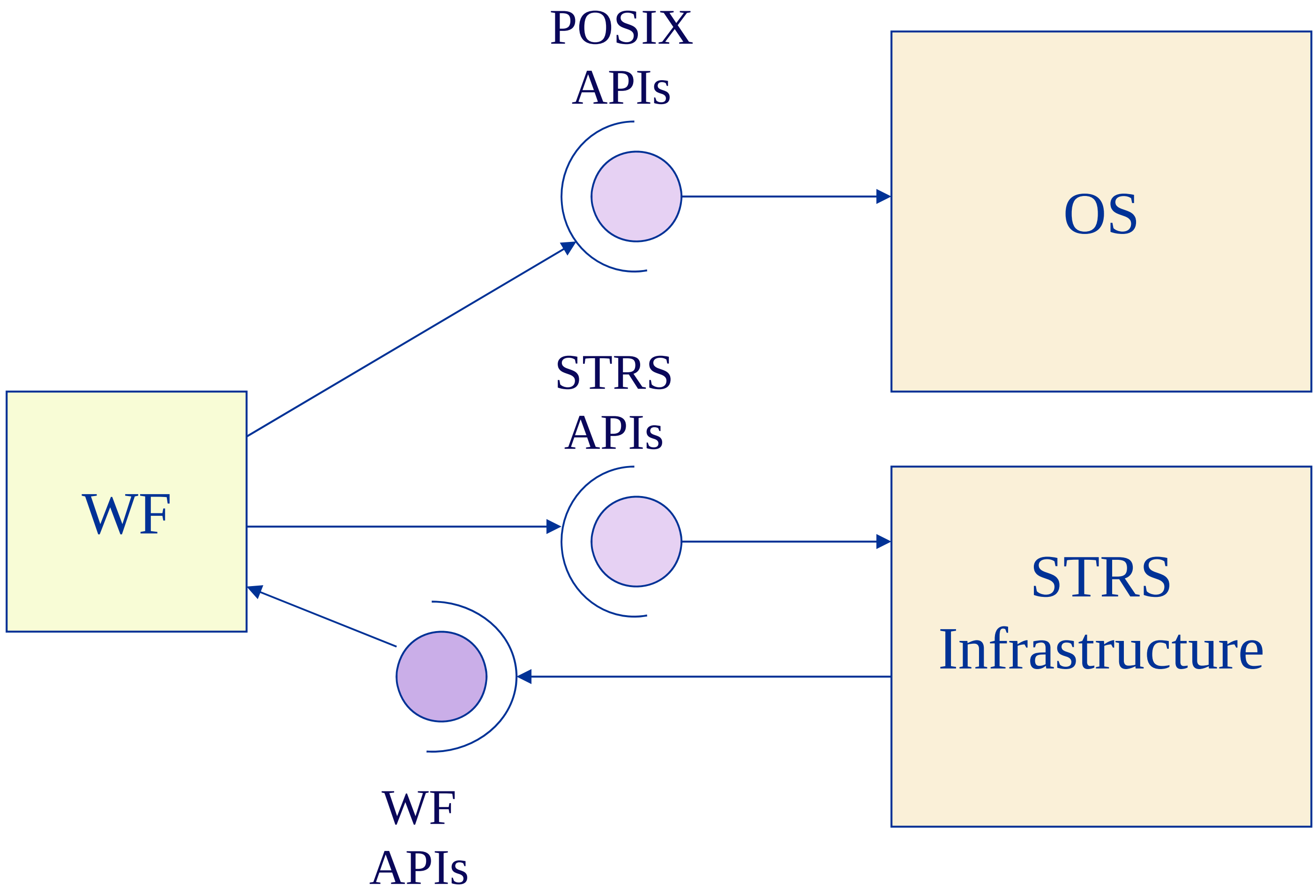


# Backup Slides

# Waveform State Diagram



# Simplified Diagram



# STRS Reference Implementation Development Process

**Use Case Identifier: Example Use Case**

**Description:** A Description of the use case goes here.

**External Actors:** External actor(s) involved with this specific use-case.

**Related Use Cases:** Use cases that related to this use case by interaction or similarity.

**Precondition:** Any precondition that must exist before this use case can occur. Usually the initial RADIO state and WAVEFORM state is listed
**Triggering event:** Event that triggers use case. Not all use cases have triggering events

**Main Flow:**
1.This will be an ordered list of steps necessary to perform interaction. This is the nominal flow. Alternate flows will be listed below.
2.Step 2
3.Step 3
4.Step 4
5.etc

**Resulting event:** If completion of use case results in an event it is listed here. Usually the resulting RADIO state and WAVEFORM state is listed.
**Post condition:** Describe the result of the use case interaction. (This is the post condition from the nominal flow)

**Alternatives:**
**1a)** This is where alternate flows are identified. The alternative will be identified by the number of the main flow where the branch occurs followed by a letter a-z.
**3a)** This is an example of an alternative flow for step 3.
**3b)** This is the second step in the alternative flow for step 3.
**7a-8a)** This is an alternative flow for a range of steps from the main flow.

**Comments:** Comments on use case.

Requirements

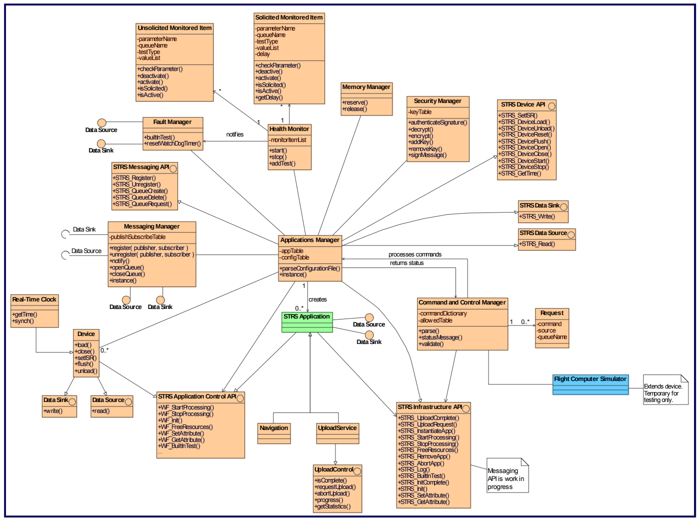
Use Case

Refines Requirements

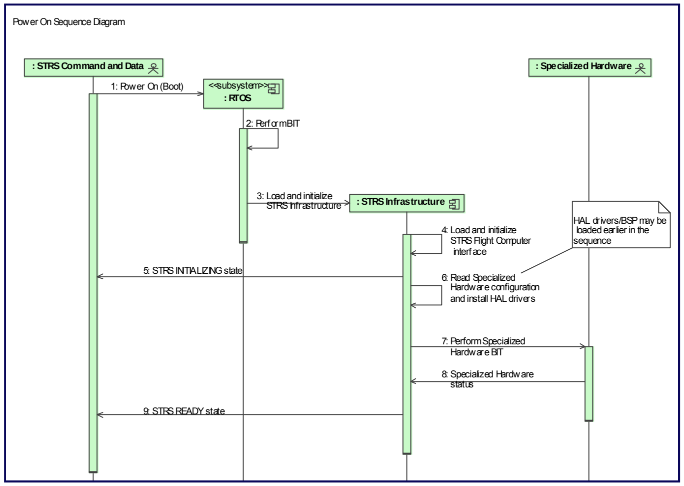
Drives Static View

Drives Dynamic View

Refines Requirements

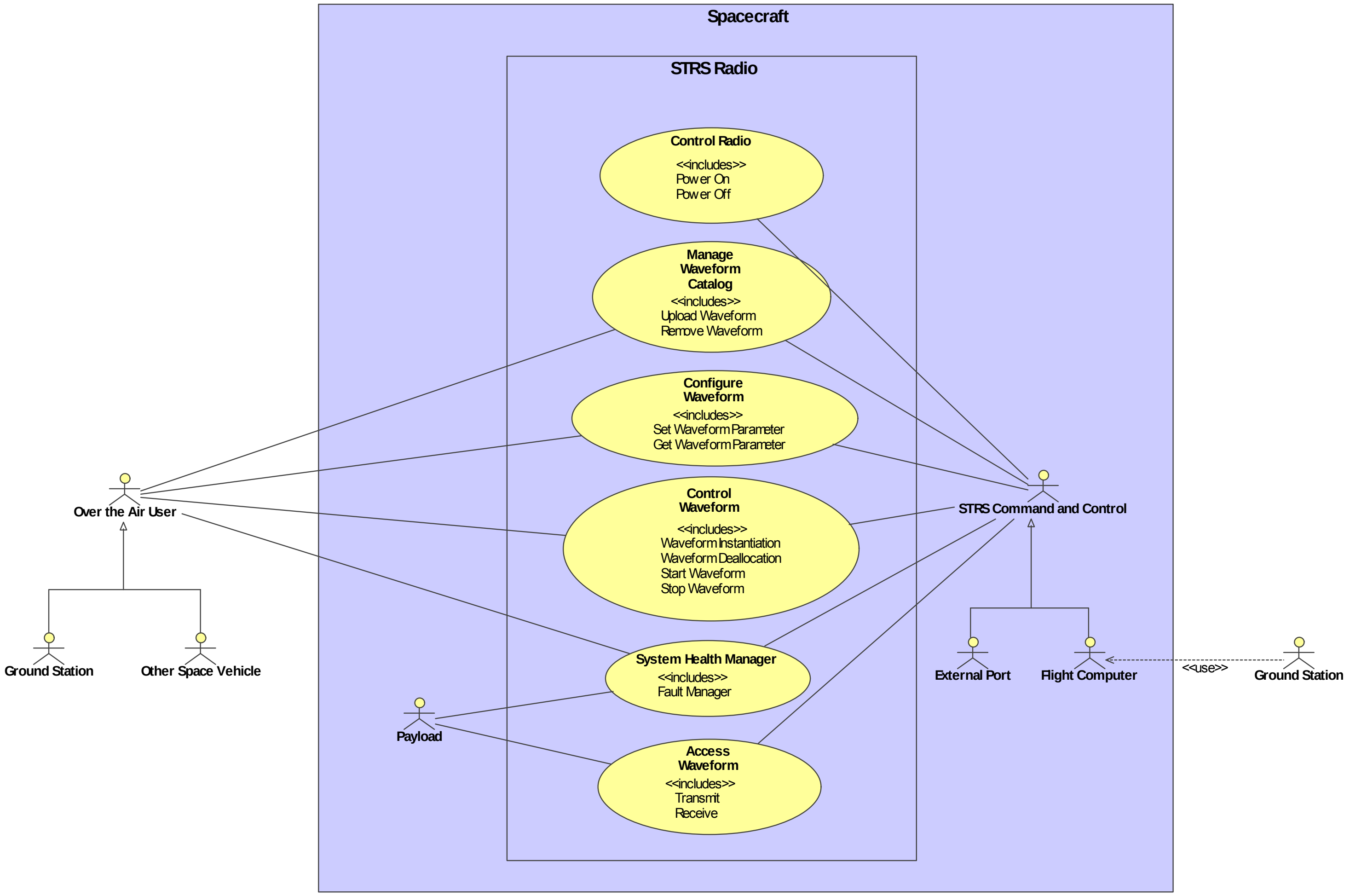


Class Diagram


Power On Sequence Diagram
: STRS Command and Data
<<subsystem>>
: RTOS
: Specialized Hardware
1: Power On (Boot)
2: PerformBIT
3: Load and initialize STRS Infrastructure
: STRS Infrastructure
4: Load and initialize STRS Flight Computer interface
HAL drivers/BSP may be loaded earlier in the sequence
5: STRS INITIALIZING state
6: Read Specialized Hardware configuration and install HAL drivers
7: Perform Specialized Hardware BIT
8: Specialized Hardware status
9: STRS READY state

Sequence Diagram

# Use Case Overview



Spacecraft

STRS Radio

Control Radio
<<includes>>
Power On
Power Off

Manage Waveform Catalog
<<includes>>
Upload Waveform
Remove Waveform

Configure Waveform
<<includes>>
Set Waveform Parameter
Get Waveform Parameter

Control Waveform
<<includes>>
Waveform Instantiation
Waveform Deallocation
Start Waveform
Stop Waveform

System Health Manager
<<includes>>
Fault Manager

Access Waveform
<<includes>>
Transmit
Receive

Over the Air User

Ground Station

Other Space Vehicle

Payload

STRS Command and Control

External Port

Flight Computer

<<use>>

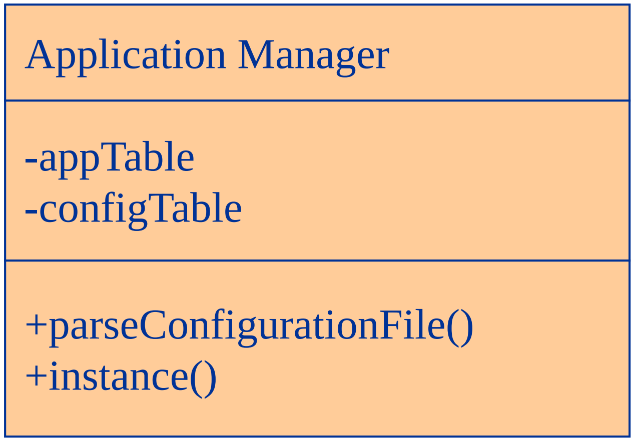
Ground Station

A set of use cases were developed which is a set of scenarios that capture the different ways that external users interact with the STRS radio.

# Class Example

## Application Manager

- The Application Manager is responsible for the passing of messages or invoking commands in other application objects such as devices, waveforms, or services actively running on the STRS radio.
- It is responsible for creating or aborting application objects, waveforms, or services.
- It is also responsible for parsing the Configuration Files and setting corresponding values in the appropriate classes.



Application Manager

-appTable
-configTable

+parseConfigurationFile()
+instance()

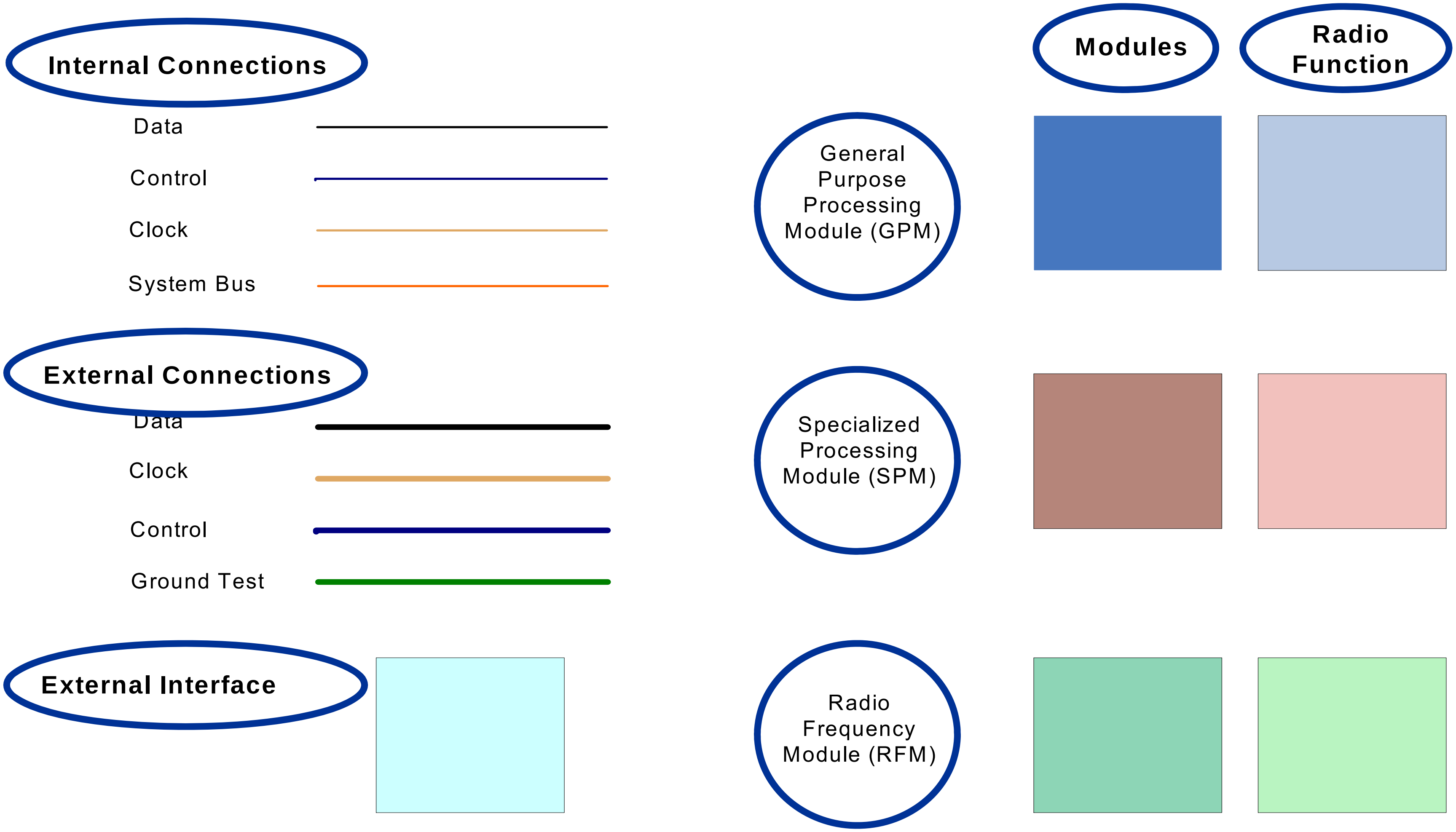
Above is an example of the UML representation of a Class

Name – *Name that identifies the class and describes the functionality*

Attributes – *Variables containing the applicable data*

Methods – *Functions that are called to implement some operation*

# STRS Open Architecture Hardware Representation



Internal Connections

Data

Control

Clock

System Bus

External Connections

Data

Clock

Control

Ground Test

External Interface

Modules

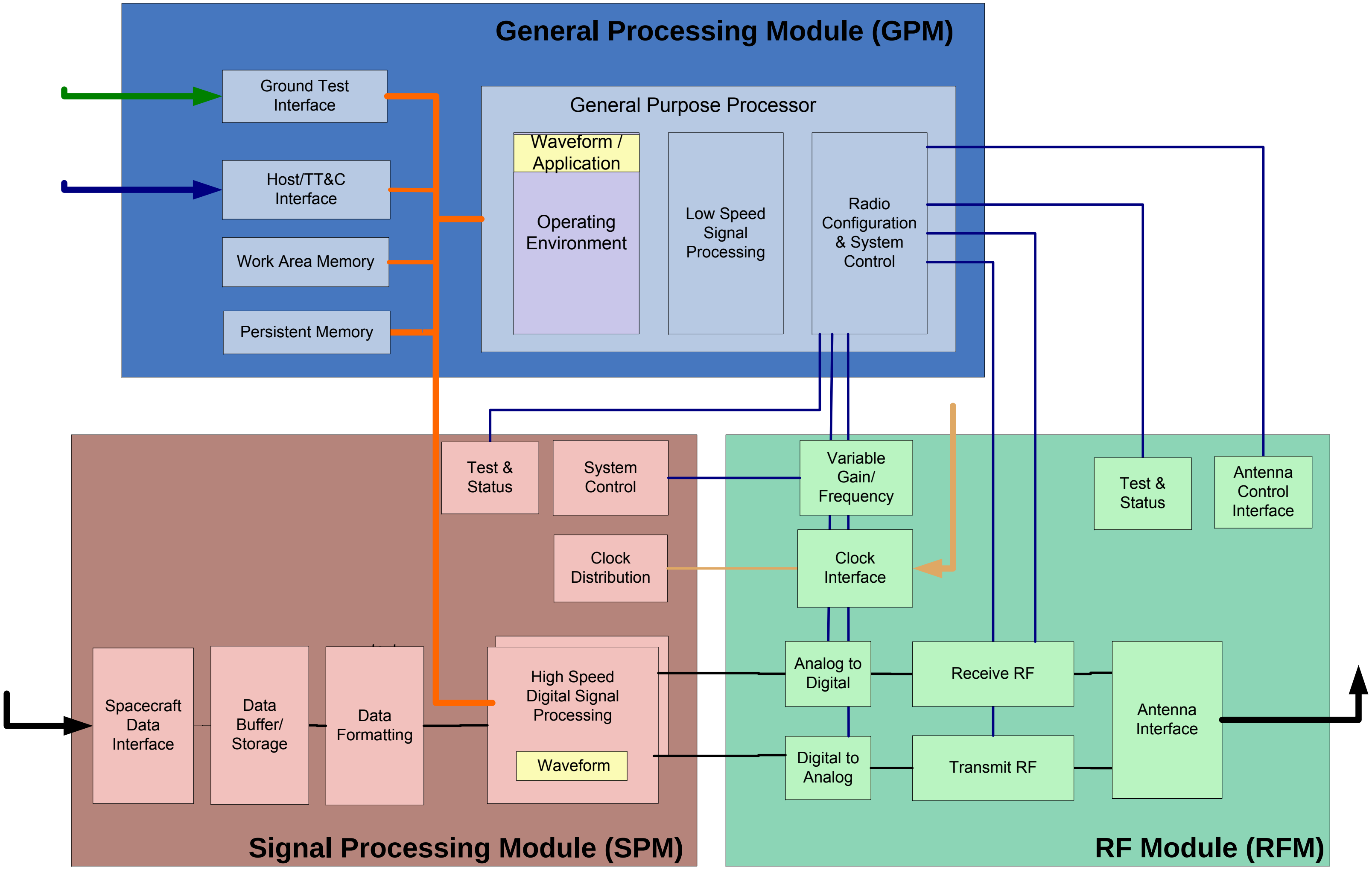
Radio Function

General Purpose Processing Module (GPM)

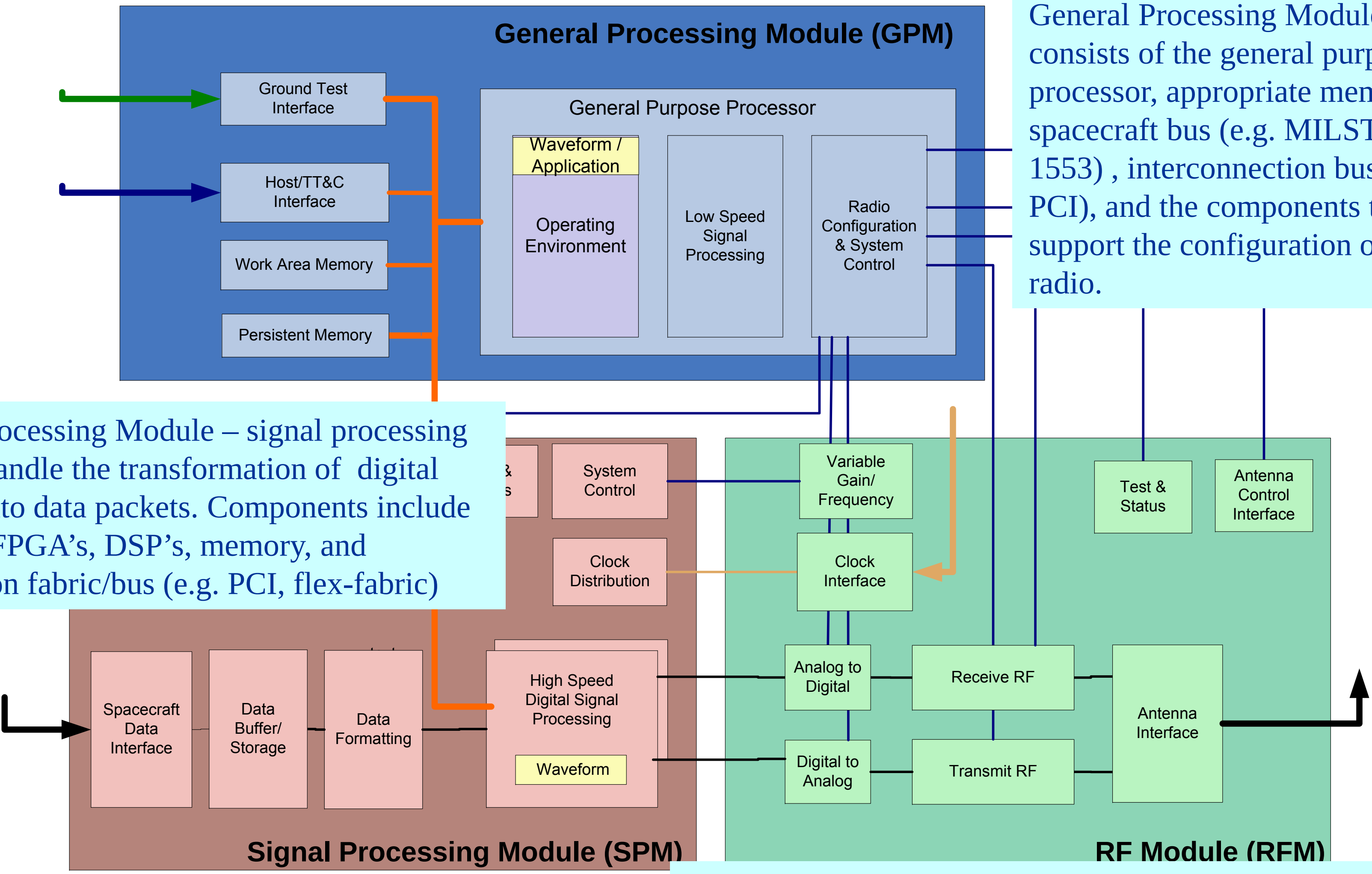
Specialized Processing Module (SPM)

Radio Frequency Module (RFM)

# SDR/STRS Hardware Functional Diagram



# SDR/STRS Hardware Functional Diagram

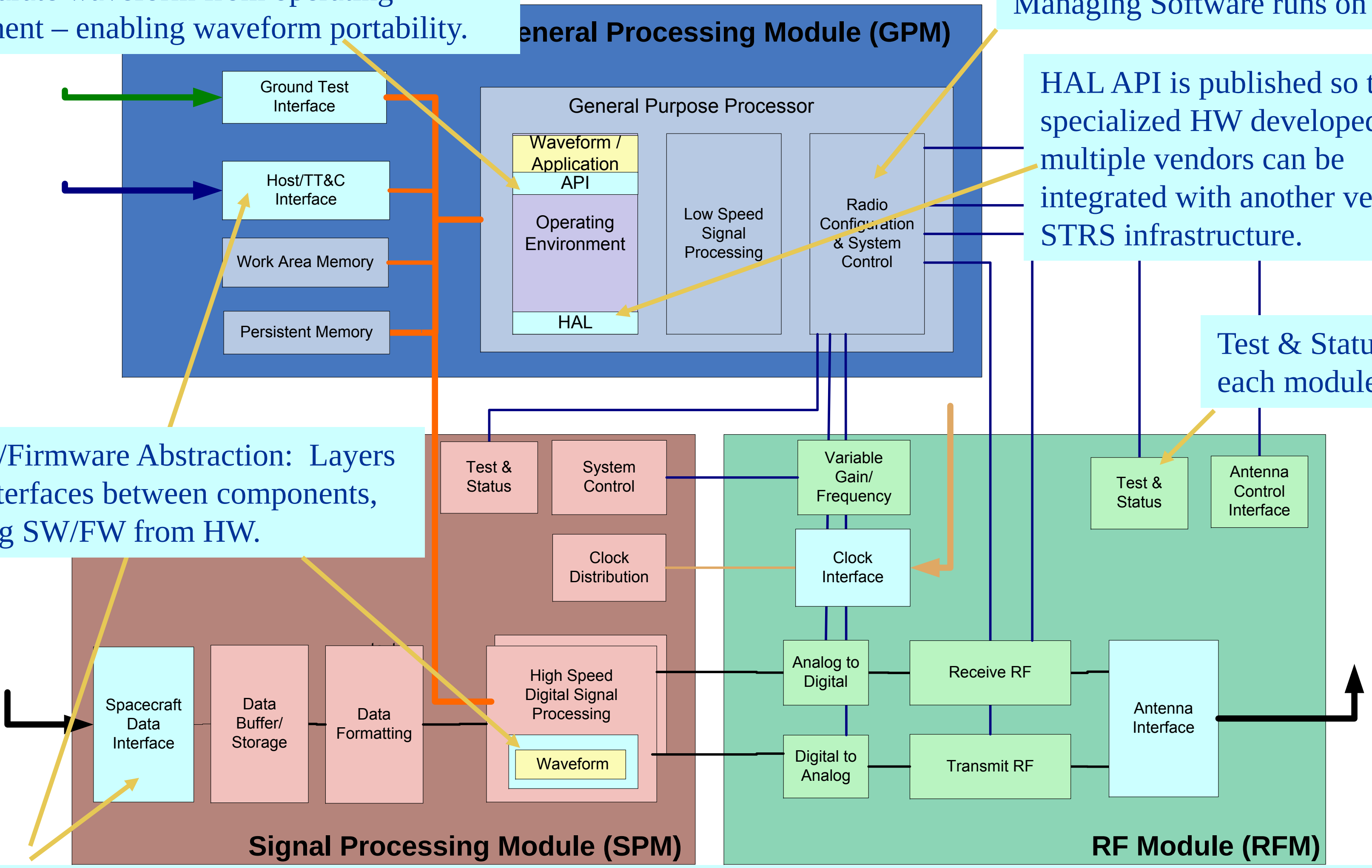


General Processing Module – consists of the general purpose processor, appropriate memory, spacecraft bus (e.g. MILSTD-1553) , interconnection bus (e.g. PCI), and the components to support the configuration of the radio.

Signal Processing Module – signal processing used to handle the transformation of digital signals into data packets. Components include ASIC's, FPGA's, DSP's, memory, and connection fabric/bus (e.g. PCI, flex-fabric)

RF Module –handles the RF functionality to transmits/receive the digital signal. Its associated components include RF switches, diplexer, filters, LNAs and power amplifiers.

# STRS Hardware Functional Diagram



APIs separate waveform from operating environment – enabling waveform portability.

Managing Software runs on GPM

HAL API is published so that specialized HW developed by multiple vendors can be integrated with another vendor's STRS infrastructure.

Test & Status on each module

Software/Firmware Abstraction: Layers define interfaces between components, separating SW/FW from HW.

Module Interfaces abstract and define the module functionality for data flow to waveform components. Enables multiple vendors to provide different modules or add modules to existing radios. Electrical interfaces, connector requirements, and physical requirements are specified/published by the platform provider.

# The End

| | |
|---|---|
| **REPORT DOCUMENTATION PAGE** | *Form Approved*<br>*OMB No. 0704-0188* |

| | | |
|---|---|---|
| **1. REPORT DATE** *(DD-MM-YYYY)*<br>01-12-2012 | **2. REPORT TYPE**<br>Technical Memorandum | **3. DATES COVERED** *(From - To)* |

**4. TITLE AND SUBTITLE**
Space Telecommunications Radio System (STRS) Architecture
Tutorial Part 1--Overview

**5a. CONTRACT NUMBER**

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**

**6. AUTHOR(S)**
Handler, Louis, M.; Briones, Janette, C.; Mortensen, Dale, J.; Reinhart, Richard, C.

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**
WBS 439432.04.07.01

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
National Aeronautics and Space Administration
John H. Glenn Research Center at Lewis Field
Cleveland, Ohio 44135-3191

**8. PERFORMING ORGANIZATION REPORT NUMBER**
E-18561

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
National Aeronautics and Space Administration
Washington, DC 20546-0001

**10. SPONSORING/MONITOR'S ACRONYM(S)**
NASA

**11. SPONSORING/MONITORING REPORT NUMBER**
NASA/TM-2012-217819

**12. DISTRIBUTION/AVAILABILITY STATEMENT**
Unclassified-Unlimited
Subject Categories: 17 and 61
Available electronically at http://www.sti.nasa.gov
This publication is available from the NASA Center for AeroSpace Information, 443-757-5802

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**

Space Telecommunications Radio System (STRS) Architecture Standard provides a NASA standard for software-defined radio. STRS is being demonstrated in the Space Communications and Navigation (SCaN) Testbed formerly known as Communications, Navigation and Networking Configurable Testbed (CoNNeCT). Ground station radios communicating the SCaN testbed are also being written to comply with the STRS architecture. The STRS Architecture Tutorial Overview presents a general introduction to the STRS architecture standard developed at the NASA Glenn Research Center (GRC), addresses frequently asked questions, and clarifies methods of implementing the standard. The STRS architecture should be used as a base for many of NASA's future telecommunications technologies. The presentation will provide a basic understanding of STRS.

**15. SUBJECT TERMS**
User manuals (computer programming); Architecture (computers); Telecommunications

| | | | | | |
|---|---|---|---|---|---|
| **16. SECURITY CLASSIFICATION OF:** | | | **17. LIMITATION OF ABSTRACT** | **18. NUMBER OF PAGES** | **19a. NAME OF RESPONSIBLE PERSON**<br>STI Help Desk (email:help@sti.nasa.gov) |
| **a. REPORT**<br>U | **b. ABSTRACT**<br>U | **c. THIS PAGE**<br>U | UU | 46 | **19b. TELEPHONE NUMBER** ***(include area code)***<br>443-757-5802 |