CASE STUDY: USING THE OMG SWRADIO PROFILE AND SDR FORUM INPUT FOR NASA'S SPACE TELECOMMUNICATIONS RADIO SYSTEM

Janette C. Briones (NASA Glenn Research Center, Cleveland, Ohio, USA; Janette.C.Briones@nasa.gov); Louis M. Handler (NASA Glenn Research Center, Cleveland, Ohio, USA); Charles S. Hall (Analex Corporation, Cleveland, Ohio, USA); Richard C. Reinhart (NASA Glenn Research Center, Cleveland, Ohio, USA); and Thomas J. Kacpura (NASA Glenn Research Center, Cleveland, Ohio, USA).

ABSTRACT

The Space Telecommunication Radio System (STRS) standard is a Software Defined Radio (SDR) architecture standard developed by NASA. The goal of STRS is to reduce NASA's dependence on custom, proprietary architectures with unique and varying interfaces and hardware and support reuse of waveforms across platforms. The STRS project worked with members of the Object Management Group (OMG), Software Defined Radio Forum, and industry partners to leverage existing standards and knowledge. This collaboration included investigating the use of the OMG's Platform-Independent Model (PIM) SWRadio as the basis for an STRS PIM. This paper details the influence of the OMG technologies on the STRS update effort, findings in the STRS/SWRadio mapping, and provides a summary of the SDR Forum recommendations.

1. INTRODUCTION

With the advancements in digital signal processing systems over the last few years, software defined radios can now support the signaling waveforms in the 10-100's of Mbps required by NASA, both now and envisioned in the future Lunar networks. However, NASA's unique requirement of operation in space places severe constraints on the digital processing and analog hardware that NASA can use in the space domain. These constraints include limited power, mass, size, heat, and risk such that processing capacity for space often lags the capability available terrestrially by several technology generations resulting in less processing and memory resources in space than terrestrially available.

As NASA begins to infuse SDR technology into space missions, an effort was initiated to consider what SDR architecture is appropriate for NASA to reduce its risk of developing and using SDRs, while maintaining a minimal architecture profile to meet the resource constrained platforms required by missions. In 2007, NASA, with industry contributions, published the STRS Architecture Standard Release 1.01 [2] which defined the initial architecture for space SDRs. The goal was to minimize the architectural requirements on an SDR, yet provide limited standardization across different SDRs within the Agency. The standardization was intended to reduce NASA's dependence on custom, proprietary architectures with unique and varying interfaces and hardware.

In the last year, NASA has published updates to the STRS architecture for space software defined radios [1]. Many of these updates have focused on key elements of the software architecture, including leveraging industry comments and SDR Forum Space Working Group (SWG) support. The STRS Architecture Standard document [2] describes a software architectural model that shows the relationship between the software layers in an STRS compliant radio. The model illustrates the different software elements used in the software execution and defines the application programming interface (API) layers between a waveform application and the operating environment, and between the operation environment and the hardware platform.

The OMG's SWRadio specification [3] is focused on the portability of waveforms across software defined radios. It does this by adding communications and Open Systems Interface components and facilities and a model/technology separation to the Software Communication Architecture (SCA) [4]. Additionally, it creates a new standardized Unified Modeling Language (UML) profile for the software radio. NASA leveraged the OMG SWRadio specification to better define both the infrastructure and waveform APIs. The models are defined using UML, which supports the description of the software systems using an object-oriented style. The UML models are used to visualize and provide a formal description of the components and the interfaces between them. The UML models are used to show the

mandates elements of the STRS architecture as well as additional optional functionality.

2. OMG INFLUENCE

A key recommendation from the SDR Forum Space Working Group was to align with the OMG SWRadio specification where possible. This recommendation was implemented by NASA. NASA's first step in the alignment process was to insure the proper meta-model and mapping of the OMG, SCA, and space Platform-Specific Models (PSMs) and PIMs. Figure 1 illustrates that the OMG and space PIMs are nearly the same, but that the PSMs and implementations are very different. NASA extended the SWRadio specification to incorporate space-based components tempered by the space constraints and mission requirements for optimizing the use of available resources. It also added components to provide means to conduct more extensive testing, which is required for space applications.

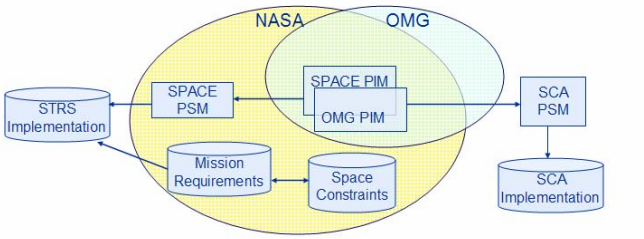

Figure 1 – PIM/PSM

The OMG influenced NASA's decision to change the method names to align the naming convention more closely to the OMG's PIM but to retain the prefixes used by STRS (WF and STRS) to support C language implementation. The group also evaluated the use of ports at the PIM level for STRS and the potential use of transformations to entry points at the PSM level.

3. PIM/PSM MAPPING

Figure 2 is an STRS class diagram in UML that illustrates the inheritance between the classes and the corresponding implementation objects in C++ and the API groupings. The STRS Application Control API is comprised of the STRS ComponentIdentifier, STRS ControllableComponent, STRS LifeCycle, STRS PropertySet, and STRS TestableObject API groups. These groups exhibit similar functionality to the OMG SWRADIO or SCA specifications. The major difference between the SWRadio specification and the STRS PSM involve the replacement of ports in the SWRadio specification with the optional source or sink in STRS. The port replacement for STRS was necessary because STRS does not require CORBA. Instead, STRS is designed to have a method in a waveform call a method in another waveform (or Device) by calling a corresponding method in the infrastructure. Having such methods in the infrastructure is a heavy burden unless the signatures for those methods are known in advance. The STRS API was implemented using the same OMG SWRadio PIM.

The naming conventions being followed for STRS are similar to the SWRadio PIM but also include STRS specific requirements. The STRS method names were updated to conform more closely to the OMG SWRadio naming convention, with the addition of the prefixes WF and STRS to support C language binding. For example, the STRS_ControllableComponent adds "STRS_" to the name of the component and "WF_" is appended to the OMG/SWRadio "start" and "stop" operations to obtain "WF_Start" and WF_Stop", respectively. The figure becomes largely a PSM diagram and captures the naming conventions as they transform the SWRadio PIM to the STRS architecture and implementation technologies.

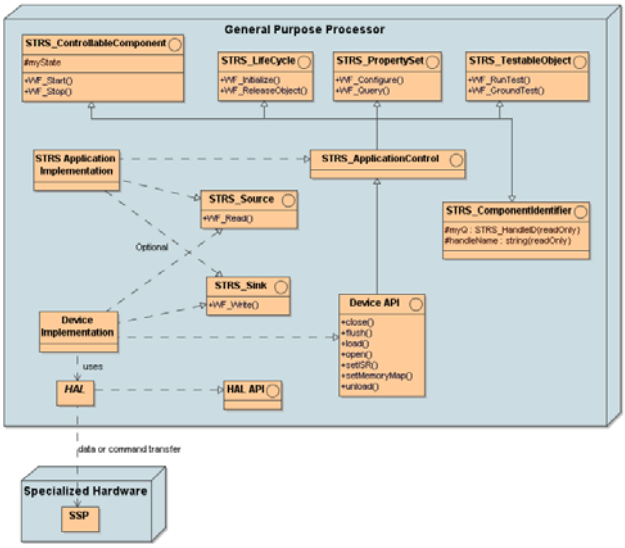

Figure 2 – STRS PSM

NASA explored the transformation rules to align with the OMG's PIM to allow a C language implementation, where STRS would need to transform the exceptions into return values from the methods. To make the transformation rule(s) easier to write for a STRS PSM it was recommended that the STRS PSM leverage the SWRadio names generally, and the prefix additions reflect the STRS specific needs.

NASA extended the SWRadio Specification to incorporate space-based components tempered by the space requirements for interoperability, reconfigurability, reprogrammability, and portability. In

STRS_TestableObject there are two operations, WF_RunTest and WF_GroundTest(). WF_GroundTest was added because NASA requires extensive testing before sending an SDR into space. In the future, the WF_GroundTest() method could be optional in the SWRadio PIM, but is required for STRS implementations..

Other differences in the SWRadio mapping to the space PIM/PSM exist. OMG identifiers have an ID and STRS has an ID and name pair. For example, a waveform has a unique name and ID associated with it where the ID is used in communicating with other parts of the radio and the name is used within the waveform for identification of the waveform in a more meaningful way. The OMG has only a string ID that is set initially and it is assumed that one can get the name using CORBA methods; however, STRS has an integer handle ID and a sting name that are set initially. The handle ID is used in STRS commands and the name is used in messages. The STRS_ComponentIdentifier is required for STRS implementation but for SWRadio is optional.

Table 1 shows the interface and method name comparison between STRS, OMG and SCA. Although the PSM names are different, at the PIM level the names could match one for one. The PSM names are different to distinguish the implementation differences, such as allowing C but not CORBA. The OMG uses ports instead to pass data between the different components.

STRS	OMG	SCA
STRS_ControllableComponent:: • WF_Start() • WF_Stop	ControllableComponent:: • start() • stop()	Resource:: • start() • stop()
STRS_LifeCycle:: • WF_Initialize() • WF_ReleaseObject()	Lifecycle:: • initialize() • releaseObject()	Lifecycle:: • initialize() • releaseObject()
STRS_PropertySet:: • WF_Configure() • WF_Query()	PropertySet:: • configure() • query()	PropertySet:: • configure() • query()
STRS_TestableObject:: • WF_RunTest() • WF_GroundTest()	TestableObject:: • runTest()	TestableObject:: • runTest()
STRS_Sink:: • WF_Write()		
STRS_Source:: • WF_Read()		

Table 1 - Interface/Method Name Comparison

4. POSIX

One of the primary goals of the STRS architecture is to facilitate the porting of waveform applications to new radio platforms. The architecture supports portability by requiring waveform applications running on the general purpose processor (GPP) to use a set of standard APIs to access the radio resources. Many of the resources and services provided by the radio will be via the real-time operating system running on the GPP. Instead of creating STRS specific interfaces, the architecture uses POSIX, an open standard defined by the IEEE and the Open Group, to provide applications a portable interface to operating system resources.

POSIX is an acronym for Portable Operating System Interface and refers to a family of IEEE standards which describe the fundamental services and functions necessary to provide a Unix-like kernel interface to applications. The limited resources on a space-based platform make it impractical to support the base IEEE 1003.1 POSIX standard which supports the large set of capabilities found on a UNIX server. STRS therefore uses IEEE 1003.13 “Standard for Information Technology Standardized Application Environment Profile (AEP) POSIX®Realtime and Embedded Application Support” that was created to address the issues of limited resource platforms by defining subsets of the IEEE 1003.1 specification. These subsets are defined using a feature of POSIX 1003.1 which provides a means to implement a subset of the interfaces by using sub profiling option groups. These option groups specify units of functionality that can be removed from the base POSIX specification.

The IEEE 1003.13 standard specifies four AEPs that are more suitable to embedded platforms. These profiles are:

PSE51 – Minimal Realtime Systems Profile 51
PSE52 – Realtime Controller System Profile 52
PSE53 – Dedicated Realtime System Profile 53
PSE54 – Multi-Purpose Realtime System Profile 54

The PSE54 profile is the largest with each lower numbered profile being a smaller subset as shown in Figure 3.

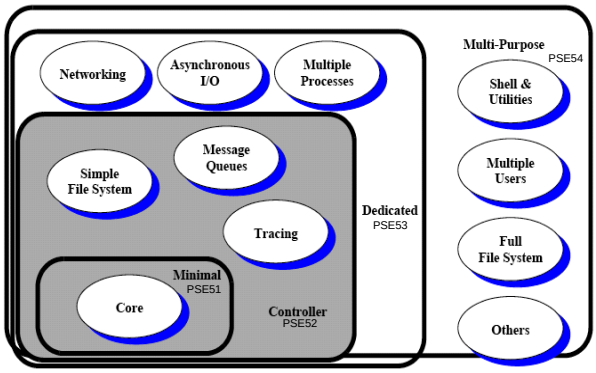

Figure 3 - PSE54 Profile

The STRS standard requires that platforms provide the operating system interfaces defined in PSE51, which is the smallest of the four profiles.

An STRS operating environment can either use an RTOS that conforms with 1003.13 PSE51 or provide a POSIX abstraction layer that provides all PSE51 interfaces as shown in Figure 4.

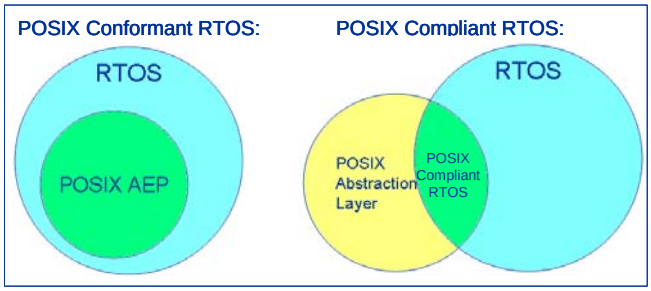

Figure 4 POSIX Conformant vs Compliant RTOS

Some NASA missions have resource constraints that will make it onerous to support even the minimal POSIX PSE51 profile. Based on Space Working Group discussions a waiver process is being developed to allow these constrained missions to deploy operating environments with a POSIX abstraction layer providing the minimal subset of POSIX (PSE51) necessary to support the mission waveforms. The waveforms could be ported to radios supporting the full PSE51 profile however the waiver radio would be limited to running waveforms that don't require the missing interfaces.

5. SDR FORUM COMMENTS SUMMARY

The SDR Forum provided comments to NASA on the STRS Standard Release 1.01 through their document entitled; *Comments on NASA Space Telecommunications Radio System (STRS) Open Architecture Specification* [5]. The document provides both general comments and specific, recommended changes to the architecture that NASA is considering or has already implemented in the current version (Release 1.02), which is not yet formally released. Below are excerpts, shown in italics, from the document along with comments or status on the different recommendations.

Open Architecture - *The consensus of the Space WG is that the STRS should continue to evolve towards an open standard rather than a NASA unique standard. An open standard would promote wider acceptance and relieve NASA of the entire burden of maintaining a standard, while still allowing NASA to influence the content and direction. Furthermore, the development of an industry standard would provide a forum for wider contributions and comments.*

The SDR Forum recommended that the STRS align with the SDR Forum, the OMG, and the IEEE SCC41 for purposes of distributing the burden and cost of NRE across NASA and all consortia members contributing to the STRS, and to further broaden and enhance the quality of the implementation and deployment of STRS-based standards. Such quality will be realized through the development of tools implementing STRS modeling which is viable because of the market that is created based on the collaborative efforts of multiple consortia working together to establish Space SDR standards.

NASA is continuing its participation with the OMG, SDR Forum, and IEEE in the development of the open, STRS Architecture. The leveraging of the OMG UML Profile, the modifications of the architecture based on the SDR Forum comments are just a sample of the industry involvement in STRS.

Leverage Existing Standards - *With the combination of the standards development organization (SDO)/process and the mindshare of industry in the SDR Forum, OMG, and IEEE SDOs, NASA has identified a powerful collaborative core competency that it does not have to duplicate. The NASA/SDO affiliation relieves NASA of the time consuming, costly, and very complex responsibilities of standards development and maintenance, (i.e., the SDO Business Model) allowing NASA to be comprehensive in its business of Space Communications applications – and still strongly influence the standards process.*

NASA agrees that existing standards should be leveraged. While NASA's domain is unique compared to many others, the goal to reduce complexity, power consumption, and risk is generally shared by all. A common, space SDR architecture for both NASA and industry, benefits all the participants. NASA will continue to look for ways to leverage the SDO processes.

Develop an Integrated Set of Specifications - *Segregating the architecture into a cohesive set of specifications covering the system, infrastructure, and waveform would provide both complete coverage of the Space SDR system and promote more concise and clear definition of each of the areas by limiting assumptions and implementation approaches within the specification document and forcing a specific set of interfaces and protocols to be defined.*

The Architecture Description Document that was released and reviewed at the NASA STRS industry day currently addresses the Software Infrastructure, Hardware Architecture, and Waveforms within a single specification. Consequently, the specification has tightly intertwined dependencies and implementation assumptions within each of these three areas. While there are certainly dependencies and relationships between each of these components that must be addressed in a comprehensive SDR design, each area should be developed as an independent specification to the greatest extent possible.

1. ***System Context/External Interface Specification –*** *A concise description of the system context and external interface specifications for the Space SDR should be provided as a top-level system document.*

2. ***SDR Systems Specification****: The systems specification provides a systems view of the radio. This captures radio system requirements, use cases, and quantitative information regarding the space SDR.*

3. ***Software Infrastructure Specification:*** *The software infrastructure specification should define the management infrastructure and services provided by the space SDR.*

4. ***Waveform Implementation Guideline:*** *The waveform specification should define the parameters, guidelines, and constraints that should be followed when developing a waveform for the space SDR. The waveform specification should have a format of an Interface Definition Document (IDD), which has a dependency on the system specification to identify baseline processing capabilities, interconnections, and protocols and the infrastructure specification to identify the common waveform control interfaces. It will also be dependent on the actual radio system for a particular mission. Thus, this will be a top level specification from which specific implementation specifications must be derived based on the actual radio system.*

Future versions of the architecture will continue to separate the different aspects of the architecture. A waveform specification is being written to address the waveform specific aspects and development processes for release with a future version. The current version (Release 1.02) is intended to correct deficiencies in the released version (Release 1.01) realized during the reference implementation development and thus the software, hardware, and waveform aspects remain in a single document.

STRS Architecture Conformance - *In considering the factors that constitute conformance with the STRS Architecture, the relevance of a variety of market and application concerns is recognized. In general, the requirements for conformance to or compliance with the STRS Standard should not imply or mandate an explicit or de facto implementation, or force or promote reliance on a design tool set or design process. The primary criteria for assessing conformance/compliance should be satisfaction of the behavior specification of the STRS APIs. One implication that emerges from this conformance philosophy is relaxation of the commitment to mandate POSIX as a requirement for conformance to STRS. Although many of the capabilities of POSIX can benefit STRS architecture implementations, this is not the case for every application. A mandate to conform to POSIX in every instance precludes highly-efficient implementations where the application does not require POSIX; forcing the implementation simply diminishes its efficiency. Furthermore, a mandate has the undesirable impact of forcing platform vendors to select from a very limited set of RTOS solutions that provide POSIX for their platform, and may in turn inhibit innovation and slow the adoption of STRS. Without a clearly identifiable return on investment, space radio providers are not expected to expend resources to add a POSIX interface to an existing platform. It is much less costly to simply add the abstracted STRS APIs that are used for creating and deleting task resources, especially considering that adding the POSIX interface requires expertise outside their core competency to capture share in a comparatively small market segment.*

In the current STRS version, NASA has allowed the POSIX API abstraction in lieu of a fully POSIX compliant operating system to help meet the needs on the smallest resource constrained platforms, as discussed earlier. Coupled with the earlier recommendation of noting the dependencies between operating environment and waveforms, the standard requires waveform upward compatibility with larger platform operating environments (OE) deploying a fully POSIX compliant OE.

Reduce Review Cycle - *It is recommended that NASA reduce the time to respond to industry input and release of documents related to the space software radio specifications. This will promote more timely input and feedback from industry and standards organizations and*

help achieve the deployment of the technology in the time-frame required for future missions.

NASA continues to value the input from industry and strives to provide timely feedback and document releases for comments.

6. CONCLUSION

NASA's STRS architecture has been upgraded, incorporating industry comments through the SDR Forum Space Working Group and leveraging the OMG SWRadio specification to better define both the infrastructure and waveform APIs. NASA's SDRs are resource constrained, with specific space radio architecture constrains and requirements that must be considered when leveraging commercial standards such as SWRadio specification for SDRs.

The SDR Forum's comments included: maintain STRS as an open architecture, continue to leverage OMG, SDR Forum, and IEEE input and standards processes, develop an integrated set of specifications yet separate the architecture into aspects of the SDR system, platform software infrastructure, and waveform. The SDR Forum also addressed architecture conformance, in particular suggesting an alternative to strict POSIX conformance and finally, encouraged NASA to incorporate industry comments and subsequent releases in a timely manner to strengthen the NASA industry collaboration.

This paper discussed the SDR Forum comment to leverage the OMG SWRadio specification in detail. NASA aligned with the OMG's PIM and explored the transformation rules to allow a C language implementation.

Remaining differences include: WF_Read and WF_Write are required in STRS but the SCA has pull packet and push packet defined as separate ports. WF_GroundTest is required in STRS and the SCA has only runTest to accommodate any type of testing, including the testing that is removed before final deployment. Aside from these minor differences, NASA has aligned with the OMG PIM. NASA recommends that the OMG consider GroundTest() as an optional method for testing.

7. ACKNOWLEDGEMENT

The SDR Forum comments document and the Space PIM alignment with OMG PIM were developed by NASA and a number of participating companies with highly valued input, including; ZIN Technologies, Harris Corporation, General Dynamics AIS, L-3 Corporation, PrismTech, Boeing Corporation, and Lockheed Martin Corporation.

8. REFERENCES

[1] L.M. Handler, J.C. Briones, T.J. Kacpura and C.S. Hall; "Updates to the NASA Space Telecommunications Radio System (STRS) Architecture", presented at the SDRF '07, November 5, 2007.

[2] R.C. Reinhart and T.J. Kacpura; "STRS Architecture Standard", Revision 1.01, June 2007

[3] OMG; Enhancing the Platform Independent Model (PIM) and Platform Specific Model (PSM) for Software Radio Components Specification Version 1.0 – a.k.a. SWRadio specification (OMG document number dtc/06-04-17, et. al)

[4] JTRS SCA URL< http://sca.jpeojtrs.mil/>.

[5] SDR Forum; Comments on NASA Space Telecommunications Radio System (STRS) Open Architecture Specification, July 2007.