Government Electronics & Information Technology Association

FAA SWIM Program

SOA Best Practices – Industry Input

March 2008

Acknowledgements

The Federal Aviation Administration has requested industry input on best practices of using Service Oriented Architecture (SOA) for the FAA SWIM program. GEIA is a trade association that includes many industry partners who support the FAA, and GEIA formed a working group to prepare this whitepaper in response to the FAA request.

Thanks to the following people for contributing to this whitepaper:

Doc. Version: **7** (03/24/2008)
Editor: Steve Prescott
Contributors:

Organization	Participants
Oracle	Steve Prescott Mike Yeganeh Thani Sokka Jim Simmons
Lockheed Martin	Al Secen
Computer Sciences Corp.	Vic Church
Boeing	Sherry Yang
Harris	John Dockendorf Keith Bourke
IBM	Chris Hulett Mike Moomaw
Raytheon	Eric Rolfe Ed Stevens
SITA	Kathy Kearns Mansour Rezaei-Mazinani
Nortel	Steve McAllister

About GEIA

GEIA develops and distributes forecasts of the Federal marketplace, creates best-practice industry standards, and maintains a committee structure through which its 100-plus members work with representatives of FAA and other Federal agencies on matters of mutual concern. In 2008, GEIA will be merging with the Information Technology Association of America (ITAA) and assuming the ITAA name. GEIA contact:

Dan C. Heinemeier, CAE
President
Government Electronics and Information Technology Association
2500 Wilson Boulevard, Arlington, VA 22201
www.geia.org
703-907-7565

Table of Contents

1 Preface: Scope of Recommendations 1
2 FAA SWIM Objectives 2
2.1 Purpose and Scope 2
2.2 Program Objectives 3
2.3 Design Objectives 5
3 Service Oriented Architecture 6
3.1 SOA Framework 6
3.2 SOA Benefits 8
3.2.1 Benefits To FAA Business Operations 8
3.2.2 Benefits To FAA Technology Operations 10
4 Solution Architecture 11
4.1 FAA Solution Vision 11
4.1.1 Service Container Concept 13
4.2 Industry Solution Architecture 14
4.2.1 Capability #1: Interface Management 15
4.2.2 Capability #2: Messaging 16
4.2.3 Capability #3: Security 17
4.2.4 Capability #4: Enterprise Service Management 18
4.3 SOA Design Considerations 19
4.3.1 Registry 19
4.3.2 Enterprise Service Bus 19
4.3.3 Legacy Integration 21
4.3.4 Security 22
4.3.5 Orchestration 24
4.3.6 Infrastructure Management 25
5 Best Practices 26
5.1 SOA Maturity Model 26
5.2 Business Process Management 27
5.3 Building a Service Portfolio 28
5.3.1 Service Profiling 28
5.3.2 Service Categories 29
5.3.3 Service Granularity 30
5.4 Design Patterns 31
5.5 Standards-based Integrated Solutions 32
5.6 Federal SOA Guidelines 33
6 Appendices 34
6.1 Glossary 34
6.2 FAA SWIM Acronyms 36

Figures

Figure 1: SOA Reference Architecture 6
Figure 2: SOA Framework – Service Categories 7
Figure 3: SOA Benefits to an IT Organization 10
Figure 4: SWIM Federated Enterprise Architecture 11
Figure 5: SWIM Architecture with Core Services 12
Figure 6: SWIM Segment 1 Core Capabilities 12
Figure 7: GEIA SOA Architecture for SWIM 14
Figure 8: Security Standards 23
Figure 9: SOA Maturity Model 26
Figure 10: Business Process Management – Lifecycle 27
Figure 11: Service Categories 29
Figure 12: Federal SOA Guidelines 33

1 Preface: Scope of Recommendations

This whitepaper aims to address two large topics:

- **SWIM:** FAA program for "System Wide Information Management"
- **SOA:** A software industry framework called "Service Oriented Architecture"

Each of these topics is large – and still evolving – and much discussion ensued among this paper's authors around the appropriate scope to cover. Typical question included:

- Will this piece of the solution fall in Segment-1 or some future Segment?
- Who will implement this piece: the SWIM program or the Implementing Programs?
- Will SWIM be a single enterprise-wide SOA solution or a federated model?
- Does this contradict established practice among organizations supporting FAA?
- Which SOA benefits will the FAA use: just system-interfaces or others such as Business Process Modeling, Workflow Orchestration, and Business Activity Monitoring?

These are good questions, and the answers are often still evolving along with SWIM and SOA.

Consequently, this whitepaper takes the following two steps to address this ambiguity:

- **Broad "Best Practices"**
  Because the FAA seeks industry input on SOA Best Practices, artificially limiting the recommendations in time (Segment-1 vs. Segment-2) or by implementer (SWIM vs. SIP) would under-serve that request. SWIM, the SIPs, and the eventual NextGen program have long planning horizons (20+ years), and SOA will benefit many elements of this overall NAS community over this planning horizon. Consequently, this paper initially takes a very broad scope of describing SOA best practices *for the full NAS community*, not necessarily just for the SWIM program or just for Segment-1.
- **"Closer Look" Marginal Indicators**
  At the same time, we recognize that the questions above are sensitive topics. SWIM does not want to threaten existing program with a major re-engineering of the NAS, and many of the SIPs have strong feelings about what capabilities are in-scope for *their individual programs* vs. in-scope for a SWIM "overlay" capability. To address this sensitivity throughout the paper, certain sections will include a marginal note indicating that this particular section requires further analysis regarding the proper place and way for the NAS community to implement the recommendations. Here is an example:

> **A Closer Look:**
> This topic merits additional discussion among FAA stakeholders regarding the proper time and place to incorporate these ideas into the overall FAA modernization vision – as indicated in the whitepaper's Preface.

2 FAA SWIM Objectives

2.1 Purpose and Scope

Over the next ten to fifteen years, air traffic in the United States (and the world) will change radically. There will be more traffic, perhaps three times as much. The resources of the National Airspace System (NAS) will not expand to keep pace, so greater efficiency in using the NAS will be required. There will be changes in flight patterns and strategies, such as reliance on regional airports and both larger and smaller aircraft. There will be changes in air navigation, communications, and integrated planning, involving both government and industry. In the face of these changes, the FAA needs to modernize and expand its capabilities to maintain the safety and efficiency of US aviation.

Neither the volume of airspace nor the number of runways can grow as fast as air traffic. The operational and technological changes needed to increase NAS capacity constitute the Next Generation Air Transport System, or NextGen. The changes will include more detailed and rigorous flight planning, more autonomous flight operations, and new roles for air traffic controllers (such as management by exception). NextGen will require improved common situational awareness, integration of air traffic management and control, consistent use of weather data and forecasts for flight planning, and better coordination of responses to adverse conditions. All of this requires that FAA systems become more integrated with each other and with other air traffic stakeholders.

The US Government is planning for change through the Joint Program Development Office (JPDO), which oversees the evolution of NextGen concepts. The FAA is a key participant in the JPDO, which also includes Defense, Commerce, and Homeland Security. NextGen operational improvements depend on enhanced information exchanges and integration of FAA systems.

Historically, FAA systems have been built to solve specific problems. Information sharing has occurred through negotiation of point-to-point interfaces between pairs of systems. Once defined, each interface is expensive and time-consuming to change. System evolution is constrained by the number of tightly coupled interfaces and varied modernization schedules. To streamline the evolution and modernization process, the FAA has developed the System Wide Information Management (SWIM) concept to support loosely coupled, many-to-many data exchange interfaces. When implemented, SWIM will allow information producers and consumers to exchange data in a secure, robust, standards-based, loosely coupled environment.

When implemented, SWIM will allow information producers and consumers to exchange data in a secure, robust, standards-based, loosely-coupled environment.

The FAA has established a Program Office to perform the engineering and acquisition of the SWIM environment. One of the critical early decisions was to use a service-oriented architecture (SOA) model for the environment. SWIM will be deployed in Segments (stages), with the first segment planned for the 2008-2012 timeframe. A second early decision was that

Segment 1 would be implemented by existing NAS programs (starting with ERAM) following standards and guidance defined by the SWIM PO.

This SOA Best Practices report will assist the Program Office to construct that guidance. It will identify industry best practices based on SOA programs in other contexts (government and commercial, and describe them in terms of application to SWIM goals. The remainder of this section discusses SWIM Program goals, design objectives, and high-level use cases for SWIM.

Section 2 will address the benefits expected from SWIM use of the SOA model.

The general goals of SWIM, shared by many FAA initiatives, are to improve the efficiency and usability of the NAS and to deliver enhanced value to stakeholders (NAS users, the public at large, FAA organizations and employees). The specific goals include improved sharing of information (leading to better decision-making and operational effectiveness), improved systems integration (reducing functional redundancy and improving information quality), and greater flexibility to accommodate the system and operational changes required for NextGen. To achieve these goals, specific objectives are defined.

2.2 Program Objectives

The SWIM Program Office (SPO) has established a two-layer framework for defining and describing program objectives. One layer is the end-user data exchange set; the other is the implementing technology. The former set is defined by NAS system communities of interest (COIs); the latter by the SPO system engineering and architecture team (with input from SWIM Implementing Programs or SIPs).

COI-defined Services. The primary program objective is to implement COI-defined data exchange services using SWIM as the exchange framework. Nine services (or service families) were identified for segment 1 by the COIs as feasible and desirable using current modernization programs. New sets of services will be defined (using the COI process) for subsequent segments. All of the segment 1 services represent existing or planned interfaces among NAS systems and airspace users. The objective is to use SWIM in the implementation or modernization of the planned interfaces.

Segment 1 envisions providing nine core services for three Communities of Interest:

- Flight & Flow
- Aeronautical Information
- Weather

To clarify: there is a Flight and Flow COI (focused on flight operations and traffic flow management); this COI defined a Flight Data Publication Service (FDPS) as one of the nine Segment 1 capabilities. The data exchanges identified in the FDPS reflect publication of the Flight Object concept developed by the Enroute Automation Modernization (ERAM) program. ERAM will publish the flight objects to a variety of integrating systems; the SPO goal is to use SWIM as the implementation framework. SWIM does not require new interfaces; it provides the mechanism for development that is already required.

The Segment 1 business services are defined in the SWIM Final Program Requirements (FPR): Segment 1 document dated May 23, 2007 (Revision 7.3). It identifies and describes the services in the following categories:

- Flight and Flow Management
    - Flight Data Publication
    - Terminal Data Publication
    - Flow Data Publication
    - Runway Visual Range (RVR) Data Publication
    - Reroute Data Publication
- Aeronautical Information Management
    - Special Use Area (SUA) Data Publication
- Weather
    - Corridor Integrated Weather System (CIWS) Data Publication
    - Integrated Terminal Weather Service (ITWS) Data Publication
    - PIREP Data Publication

SWIM Core Capabilities. The SWIM framework, which will be implemented in Segment 1 by existing NAS programs, will be comprised of four "core capabilities". The SPO will provide guidance on the standards to be used and off-the-shelf software to be employed.

The SWIM FPR (noted above) specifies the following core capabilities (extracted from the FPR document and slightly augmented here):

> **Interface Management** includes capabilities (Service Design-Time Environment) that enable Service Providers to expose services and Service Consumers to find services. It includes supporting capabilities such as descriptions of the services performed (typically, in a service registry) and data exchange requirements to assist in interface development. It also provides support for managing metadata such as the schemas that define the format and semantics of interface data elements.
>
> **Messaging** includes mechanisms (Service Run-Time Environment) supporting a variety of service invocation styles (e.g., 1- way or 2-way message exchange patterns with request/reply or publish/subscribe) and data exchange protocols. It enables reliable message delivery and message routing including the structures and metadata supporting routing and policy. Messaging capabilities can include reliable delivery allowing service consumers to receive queued messages while connected or after reconnecting to the network. It provides Quality of Service (QoS) including priority and response time.
>
> **Security** includes mechanisms (Service Design-Time and Run-Time Environments) to enforce security policies at the service and message level including providing authorization-based access to data and services. It ensures both Service Consumers and Service Providers can verify identities, authenticate themselves and assert access privileges via authorization; and ensures confidentiality of information exchanged while invoking and consuming services. It also protects information integrity, that is, guards against unauthorized modification of data and services. SWIM security is focused on application-level interfaces and messages consistent with enterprise SOA principles.
>
> **Enterprise Service Management** (Service Design-Time and Run-Time Environments) includes Governance and Monitoring. Governance manages services across all service lifecycle phases based on conformance to SWIM Policies and Guidelines in Service Design-Time. Monitoring is how NAS system ensures the key requirements are met including the ability to capture, view, and report on service performance and usage. QoS and other performance metrics are defined and measured consistent with system and service requirements and address items such as throughput, reliability, availability, latency, response time, and fault data (e.g., for isolation and repair).

A secondary program objective is to have implementing programs use consistent, interoperable, off-the-shelf components in the deployment of COI services. (*Off-the-shelf* specifically includes both COTS and open-source software.) Different NAS program use different system platforms (hardware and software), so it is not possible to specify any single

"one size fits all" core services implementation. Accordingly, this objective focuses on interoperability, to be achieved through standards and integration tools. For example, each SWIM implementing data producer will provide its own information assurance and security, and its own web service and messaging capabilities. The program objective is for those core services to be implemented consistently.

To that end, the SPO will define specific elements (e.g., required added elements in SOAP packaging, and required XML schema standards for common options) that will apply to various SOA mechanisms. Specific versions of standards will be defined, as well, to avoid interoperation problems. The SPO will not, however, constrain the design choices such as when to use web services and when to use message queues.

2.3 Design Objectives

As noted, one of the key design decisions for SWIM was to implement a SOA framework for information sharing. SWIM is being designed as an enterprise framework for NAS systems. As such, SWIM will require consistent approaches to service management, information assurance, service definition, service discovery, data and meta-data schema management, messaging patterns, and other aspects of information exchange. The SWIM design will be documented using FAA Enterprise Architecture-specified formalisms (e.g., selected DODAF artifacts).

For core services, the SWIM design will provide a detailed plan to assure enterprise-level consistency. The design goals include:

- Maximizing the use of COTS and open source software (minimizing the supported code base, minimizing maintenance costs)
- Using well-established standards (e.g., not all of the WS-* standards are sufficiently mature to support robust operations)
- Separating design-time and run-time capabilities (where appropriate) to permit incremental implementation of SWIM-based applications
- Meeting NAS requirements for performance, reliability, maintainability, security, and so on
- Reducing barriers to information sharing

For COI-defined services (that is, specific applications to provide services using SWIM), the primary goal is long-term interoperability. That is, the framework design for application development is intended to permit flexibility in Segment 1 while encouraging consistency of architecture in the future.

3 Service Oriented Architecture

3.1 SOA Framework

A Closer Look:
This topic merits additional discussion among FAA stakeholders regarding the proper time and place to incorporate these ideas into the overall FAA modernization vision – as indicated in the whitepaper's Preface.

SOA provides a holistic mechanism to align the business and IT organizations:

- ✓ SOA encompasses the tools and methodologies for capturing business design, and uses that design information to help improve the business.
- ✓ SOA covers the programming model, tools, and techniques for implementing the business design in information systems.
- ✓ SOA contains the middleware infrastructure for hosting that implementation.
- ✓ SOA encompasses the management of that implementation to ensure availability to the business and efficient use of resources in the execution of that implementation.
- ✓ SOA encompasses the establishment of who has authority and the processes that are used to control changes in the business design and its implementation.
- ✓ And ultimately, SOA accelerates the time-to-value for these benefits.

While point-solutions exist in the commercial and open-source communities to fulfill subset of these capabilities, basing SWIM on such a collection of disconnected elements would impose undue cost and risk on the FAA. Instead, the FAA will benefit from basing SWIM on a SOA framework that provides a comprehensive architecture and set of offerings, technologies, and practices that address all of the above points – as illustrated in Figure 1.

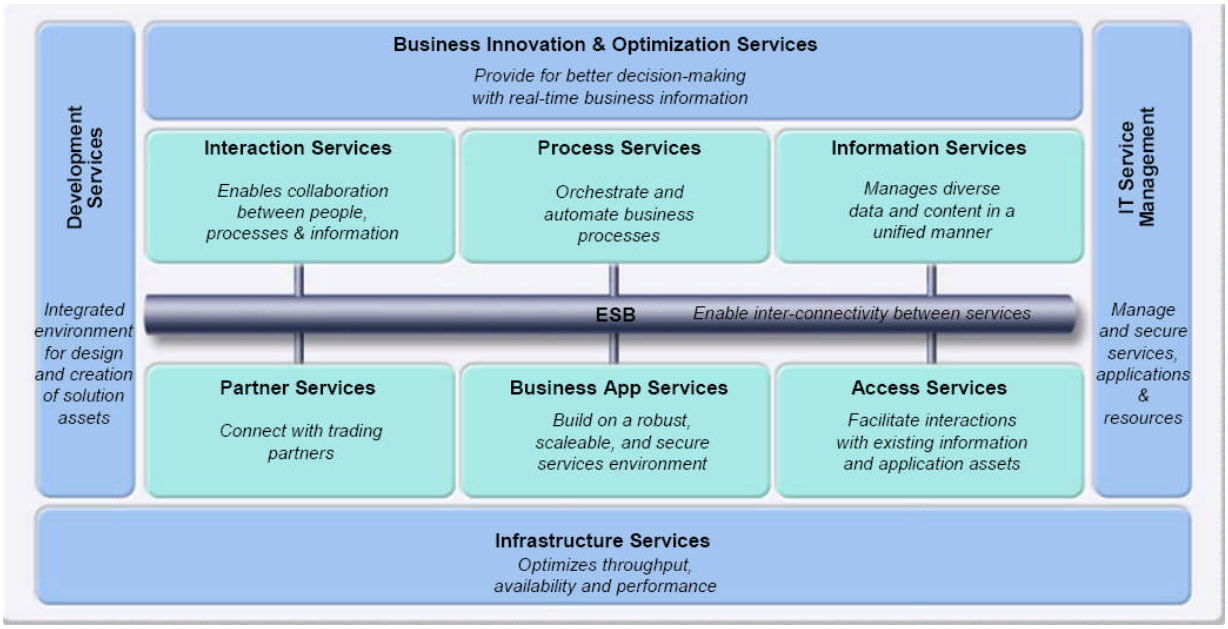


Figure 1: SOA Reference Architecture

Two bundles comprise this framework: *inner* services and *outer* services:

- **Inner Services** are used by applications within the runtime environment
- **Outer Services** are supporting components used in support of the core services

Figure 2 provides brief descriptions of service types in these two categories.

	Service Type	Description
Inner Services	Interaction Services	Provide the capabilities required to deliver IT functions and data to users, meeting their specific preferences.
	Process Services	Provide the control capabilities required to manage the flow and interactions of multiple services in ways that implement business processes.
	Information Services	Provide the capabilities necessary to federate, replicate and transform disparate data sources.
	Partner Services	Provide the document, protocol, and partner management capabilities for business processes that involve interactions with outside partners and suppliers.
	Business Application Services	Are called by service consumers. Service consumers include other components in the logical architecture such as portal or a business processes.
	Access Services	Provide bridging capabilities between core applications, prepackaged applications, enterprise data stores and the ESB to incorporate services that are delivered through existing applications into an SOA.
Outer Services	Enterprise Service Bus	Provides an infrastructure that removes the direct connection dependency between service consumers and providers.
	Business Innovation and Optimization Services	Are primarily used to represent the tools and the metadata structures for encoding and simulating the business design, including the business policies and objectives. Business innovation and optimization services exist in the architecture to help capture, encode, analyze and iteratively refine the business design.
	Development Services	Encompass the entire suite of architecture tools, development tools, visual composition tools, assembly tools, methodologies, debugging aids, instrumentation tools, asset repositories, discovery agents, and publishing mechanisms needed to construct an SOA based application.
	IT Service Management	Represents the set of management tools used to monitor an organization's service flows, the health of the underlying system, the utilization of resources, the identification of outages and bottlenecks, the attainment of service goals, the enforcement of administrative policies, and recovery from failures.
	Infrastructure Services	Form the core of the information technology runtime environment used for hosting SOA applications. These services provide the ability to optimize throughput, availability, performance and management.

Figure 2: SOA Framework – Service Categories

3.2 SOA Benefits

SOA benefits two stakeholder groups: FAA business operations and IT. Below are descriptions of the benefits to each of these groups.

3.2.1 Benefits To FAA Business Operations

The National Airspace System is tied together by data exchanges among mission systems. The exchange mechanisms are highly resistant to change, as they interlock components and limit flexibility. NAS systems are effective at maintaining the safety of US aviation, but not efficient in use of resources. Too often, data exists in one system and is needed in another—and there is no simple way to share it. As air traffic gets busier and more complex, the lack of efficiency becomes an acute problem. There are improvements planned, prototyped, demonstrated, but not implemented because the cost of change is so high and the pace of change is so slow. SWIM will enable flexibility and agility, and support the essential data sharing for the NAS of the future.

The FPR defines the mission shortfalls and the benefits that SWIM is expected to provide, addressing those shortfalls using a SOA enterprise framework.

1. **Costs to develop, test, deploy and support new interfaces and applications are too high.** Costs of developing and maintaining custom point-to-point interfaces limits connectivity. SWIM enables:

    - Reusable, loosely coupled interfaces versus many point-to-point interfaces
    - Reduced time and complexity for building new applications and interfacing existing applications
    - Common shared services for information management replacing costly redundancies

The process for defining and implementing new data interfaces is cumbersome and error prone. Each interface is designed to solve a specific problem, and this leads to mismatches for any other problem. The interface has too much, or too little, or the timing (data frequency) is wrong, or the interface uses proprietary coding. Interfaces may be optimized and tuned for a particular exchange (sometimes down to the bit level for performance reasons). Any other use of a point-to-point interface must either deal with the unique features or negotiate a generalization. The threshold for a new application to be implemented is high: either accept suboptimal data interchanges, or create new ones.

The SOA approach offers tools and patterns for more generalized interfaces. A service-oriented structure may cost more to create, but is much easier to reuse than custom interfaces. The interchange framework (e.g., web services, messaging, flexible formats built with XML) simplifies the interchange design and allows broader sharing of data.

2. **The NAS is not an agile air traffic system**. The NAS is difficult to dynamically adapt to special events, disruptions and changing NAS user business models. SWIM facilitates:

    - Greater independence of geographical facilities and operations
    - Easier and quicker system failure recovery
    - Special events planning and implementation
    - Automation and platform convergence consistent with the NAS Enterprise Architecture

SOA systems are inherently loosely coupled, and services provide separation of capability definition from its implementation. By generalizing the data sharing, SWIM will support rapid development and deployment of tools using the available data.

3. **Data sharing in the NAS is labor-intensive.** Agility requires rapid, widespread and cost-effective dissemination of information. The current NAS infrastructure makes this cost prohibitive.

    - SWIM provides the conduit so that shared data can be published once and distributed electronically.

Because of the finely tuned, optimized nature of NAS interfaces (for instance, why send metadata when the sender and receiver both now the precise nature of the data?), data products require extensive preparation and management. A case in point is the "adaptation data" describing the physical and logical elements of the NAS (fixes, airports, etc.) The data is provided by many sources and updated periodically. Every change requires elaborate reconciliation and formatting, reapplication of corrections, verification, and synchronized deployment. SWIM will use SOA mechanisms and patterns to streamline and automate the data management and inter-program coordination.

4. **Timely access to common data is lacking in the NAS.** A lack of shared situational awareness limits visibility into the current state of the NAS for NAS users and their customers.

    - SWIM makes published data available to all authorized users

A key goal of air traffic management is ensuring that all parties use the same information in making decisions. The shared situational awareness will be enhanced by making data available on demand and in common formats to all NAS users. The cost of point-to-point interfaces reduces the sharing of information; with SOA-supported common access, SWIM facilitates adding more subscribers to data feeds.

5. T**he underlying tools to support becoming a performance–based organization are currently lacking**. The information required to measure and monitor NAS performance is often not available; this limits the ability of the FAA to meet its goal to become a performance-based organization.

    - SWIM provides the mechanism so that published data can be mined for appropriate metrics.

With enterprise service management and information assurance capabilities, the SWIM infrastructure will provide the data necessary for performance optimization.

Enterprise SOA systems are flexible. The data sharing made possible through SOA mechanisms will accelerate the modernization of NAS systems and improve the performance of air traffic operations.

3.2.2 Benefits To FAA Technology Operations

An IT organization comprises many distinct roles, and SOA provides different benefits to each role. Figure 3 highlights a breakdown of these SOA benefit by IT role.

IT Role	SOA Benefits
Information Systems Architect	An information systems architect will see SOA as being about two things: ▪ SOA describes a style of Enterprise Architecture that structures artifacts in the information system as a set of services that can be composed to form other services. ▪ SOA establishes a set of principles for loose coupling, modularity, encapsulation, re-use and composability that yields the flexibility needed to ensure the information system is able to both keep up with the rate of change demanded in the business design and become a leading driver of change to achieve better productivity, profitability and competitiveness.
Systems Architect	The systems architect can gain value from SOA by exploiting the tools and methodologies offered by SOA for automating the business design that remains valuable to the business over time.
Application Programmers	From the perspective of application programmers, SOA is a set of programming models and tools for building, accessing and assembling services that implement the business design together with a runtime that will execute those services efficiently. Programmers gain value from SOA by being more productive in creating and re-using software that is more reliable and robust in the face of the evolving business design.
Operations Staff	From the perspective of the operations staff, a benefit of SOA is that it enables them to implement IT changes incrementally, replacing complex chains of machine and software dependencies with modularized services that can be substituted, tailored, modified, and deployed in a granular fashion over a virtualized infrastructure. It makes the IT staff's work easier by dividing software capabilities into units of function. It provides tools that fit the skills, conceptual model, and task that an individual IT worker needs to perform, rather than requiring every IT worker to understand everything about the distributed system and its implementation. Moreover, SOA enables the operations staff to correlate capacity requirements and problem determination with the business processes being hosted on the system. From this, the operations staff can prioritize their activities to address the issues with more relevance and impact to the business.

Figure 3: SOA Benefits to an IT Organization

4 Solution Architecture

This section discusses overall SOA-based solution architecture for SWIM. To show alignment between the FAA's existing vision for SWIM and this new industry input, the discussion first provides a foundational recap of the FAA vision. From there the discussion outlines GEIA's recommendation for an overall SOA-based SWIM Architecture. Finally the discussion provides specific design recommendations on a variety of SOA elements for SWIM.

4.1 FAA Solution Vision

The FAA is currently in the early stages of Segment 1 for SWIM. In August 2007 the FAA produced a slide deck titled SWIM Segment 1 Program Overview.[1] To establish an FAA foundation for the upcoming GEIA recommendations, the following brief discussion recaps three levels of detail from the FAA slides:

- Level 1: Segment 1 Overview ("enterprise")
- Level 2: SWIM Architecture with Core Services
- Level 3: Core Services of Segment 1

Figure 4 provides a high-level illustration of the Segment 1 overview – i.e., the enterprise wide integration of FAA systems using SWIM.

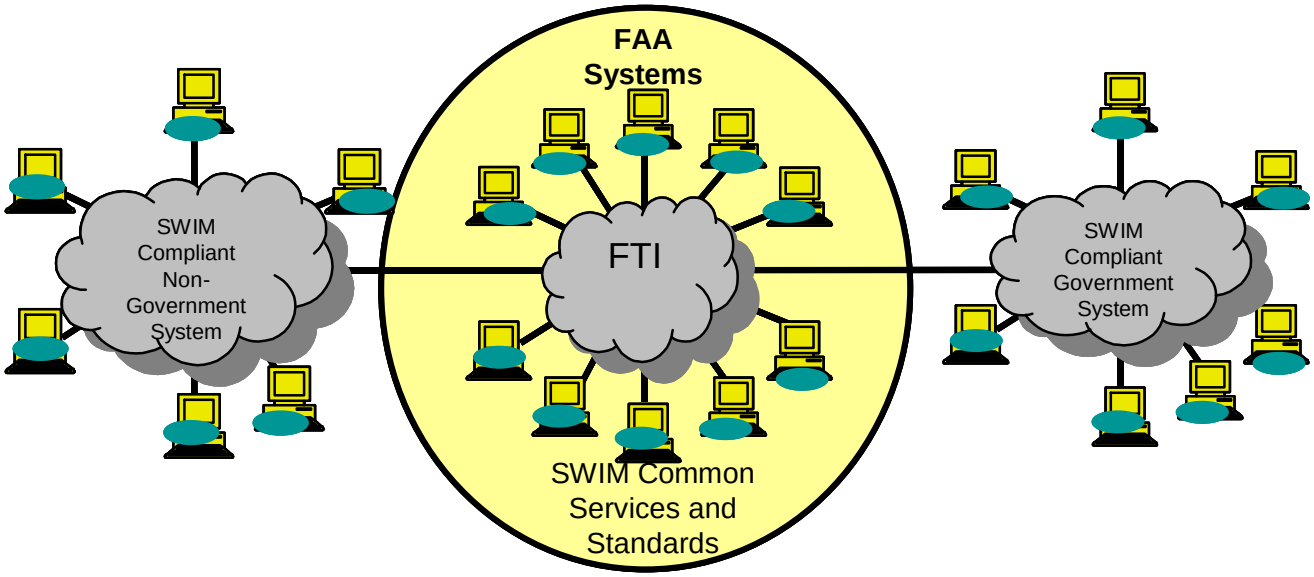


Figure 4: SWIM Federated Enterprise Architecture

The next level of detail is examining the interfaces among nodes on this overview. The FAA has established a notion called "Core Services" as a consistent capability existing at each node to provide a uniform mechanism for communicating among nodes. Figure 5 illustrates the FAA view of how Core Services fit into the overall SWIM architecture.

[1] SWIM Segment 1 Program Overview, FAA slide presentation, August 2007

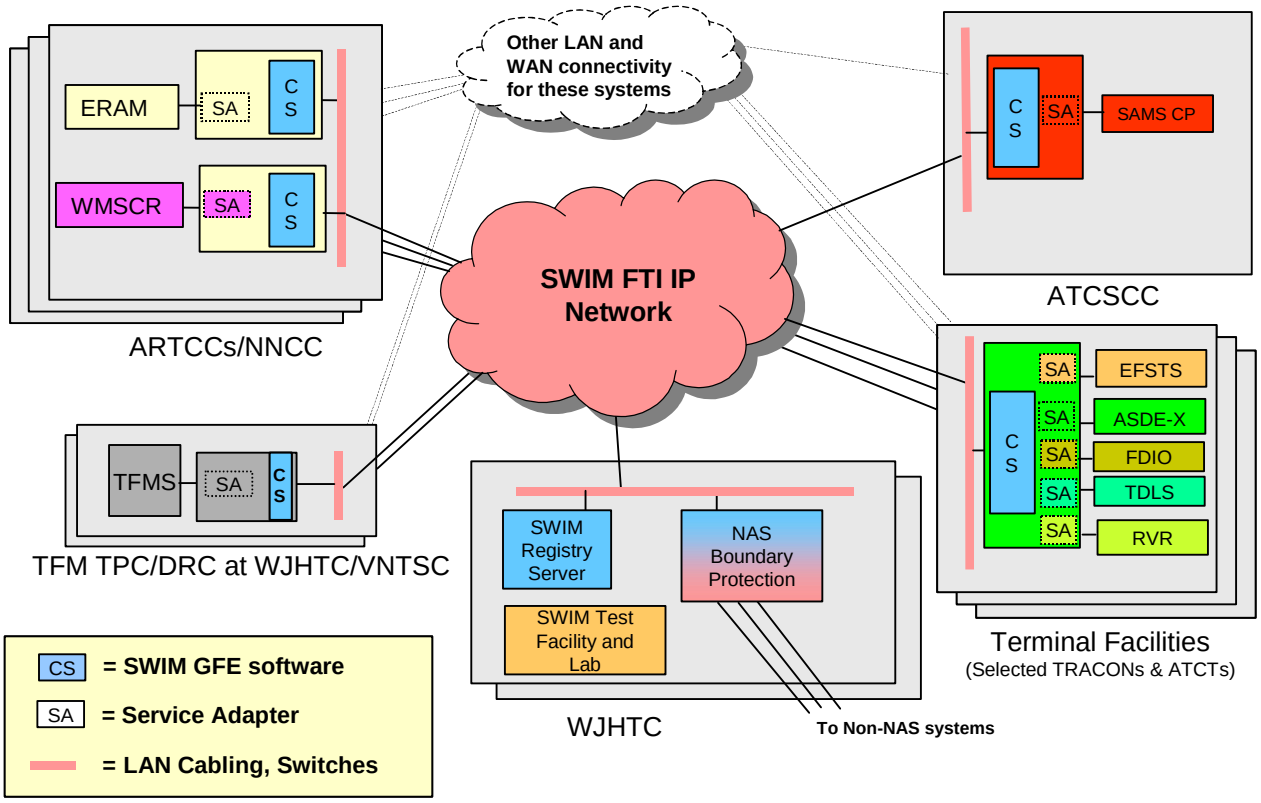


Figure 5: SWIM Architecture with Core Services

Finally, the above diagram raises the question of what specific capabilities comprise these Core Services. The FAA has provided a vision for that too – reproduced in Figure 6.

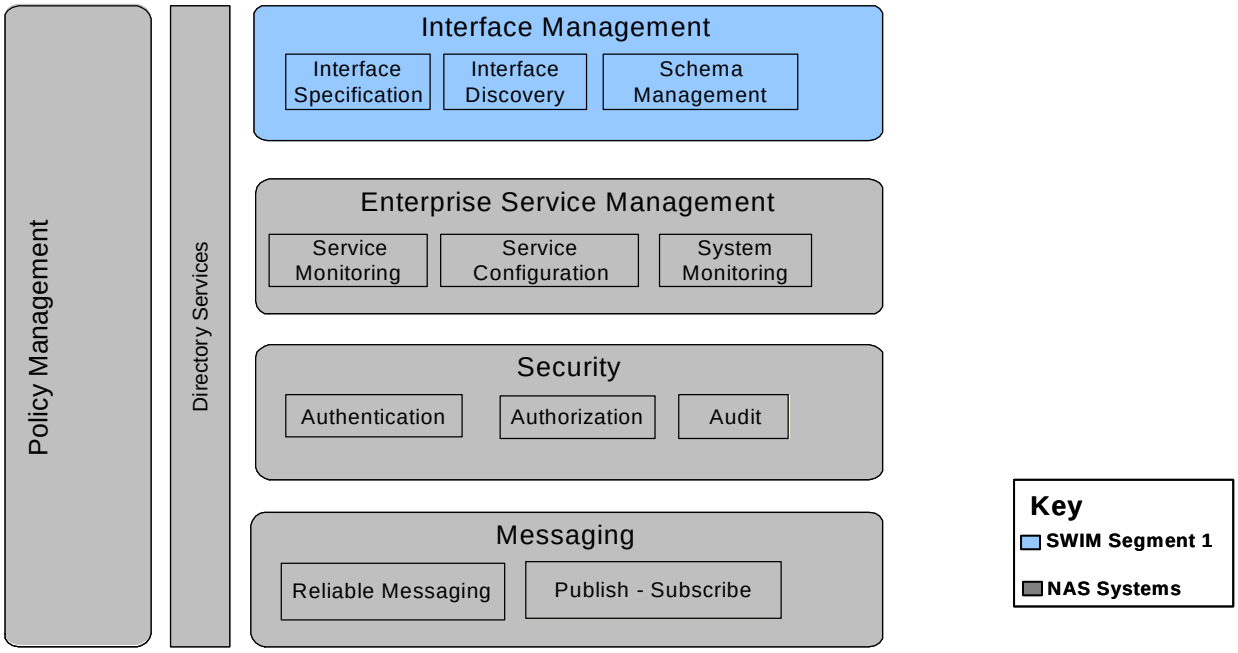


Figure 6: SWIM Segment 1 Core Capabilities

4.1.1 Service Container Concept

> **A Closer Look:**
> This topic merits additional discussion among FAA stakeholders regarding the proper time and place to incorporate these ideas into the overall FAA modernization vision – as indicated in the whitepaper's Preface.

The FAA's vision for Segment 1 of SWIM includes the concept of a "Service Container" (SC) that provides certain SOA capabilities and will be distributed in nature – located at each of the SWIM Implementing Programs (SIPs).

This raises several of the "boundary" questions mentioned in this whitepaper's Preface:

- Which SOA elements will reside *inside* the Service Container?
- Which SOA elements will reside in the SIPs but outside their Service Containers?
- Which SOA elements will the FAA provide *centrally*?
- Which SOA elements will the FAA provide in a *federated* fashion – by the SIPs?

Answers to these questions may change over time as the FAA gains experience from "early adopters" of SWIM services and adjusts the solution to deliver maximum value to FAA stakeholders. In Segment 1, SWIM will not create many central resources for implementing programs. There will not be a central ESB providing messaging, security, and similar functions; instead – for the most part – the responsibility of implementing these core (infrastructure) capabilities will belong to NAS programs such as ERAM and TFM-M. There are two areas where the FAA does envision creating centralized services during Segment 1: (a) a design-time registry to assist in common service access, and (b) test bed capabilities to support interoperability testing.

Keeping an eye toward the future – beyond Segment 1 – is important, however. While Segment 1 may not create a large number of centralized services, it is possible that future Segments will expand this pool of services. The Service Container plays a key role in delivering this flexibility to FAA programs. The Service Container will act as a service wrapper providing attachment points for security, messaging, service management, and interface management capabilities (and possibly other SWIM services in future Segments). The lightweight SC will not *provide* these capabilities, but will provide a standard mechanism for *connecting* them to services.

As the SWIM architecture evolves, the SC will help pave a way to interoperability in future FAA Segments. It is likely that the distribution of services will change over time – possibly gravitating toward the *centralized* pool. The SC can help provide continuity for FAA programs as this re-distribution of service-fulfillment occurs. Even in Segment 1, services will need infrastructure capabilities, and during Segment 1 those services will likely be fulfilled via existing FAA programs for services such as for authentication and authorization, service monitoring and management, message queue management, and so on. The SC will provide a wrapper that supports a seamless transition from *program*-provided infrastructure to *SWIM*-provided infrastructure in the future. The key is flexibility: while the SC construct does not

obligate the FAA to changing the location of services, it provides the FAA with the *ability* to re-locate certain services in the future if such relocation would be beneficial.

Two points are worth noting regarding the above FAA goals and their mapping to the "SOA Framework" outlined in section 3.1:

- Centralized vs. Federated:
  In many cases the SOA Framework components can be decentralized and federated to support the Service Container concept. At the same time, decentralized, federated components tend to add complexity and risk over centralized solutions and weight should be given to architectures that include centralized components where possible.
- Coverage of "SOA Framework" Elements:
  The SOA Framework services listed earlier in Figure 2 do represent an industry best practice – particularly for enterprise-wide SOA implementations. The decision of which SOA Framework components are contained in the Service Container is a matter of naming convention as long as all the SOA Framework components are included in the overall FAA NAS solution-space.

4.2 Industry Solution Architecture

GEIA supports the FAA vision for the SWIM architecture. The FAA architecture reflects a solid understanding of both the macro view – how to establish an enterprise-wide foundation for modernization – and the micro view – how to link this foundation to specific applications.

Figure 7 illustrates a GEIA-endorsed high-level architecture toward which SWIM should evolve. There are two key reasons for GEIA recommending this architecture: it adheres to the overall SOA Framework (section 3.1) and to the FAA SWIM vision (section 4.1).

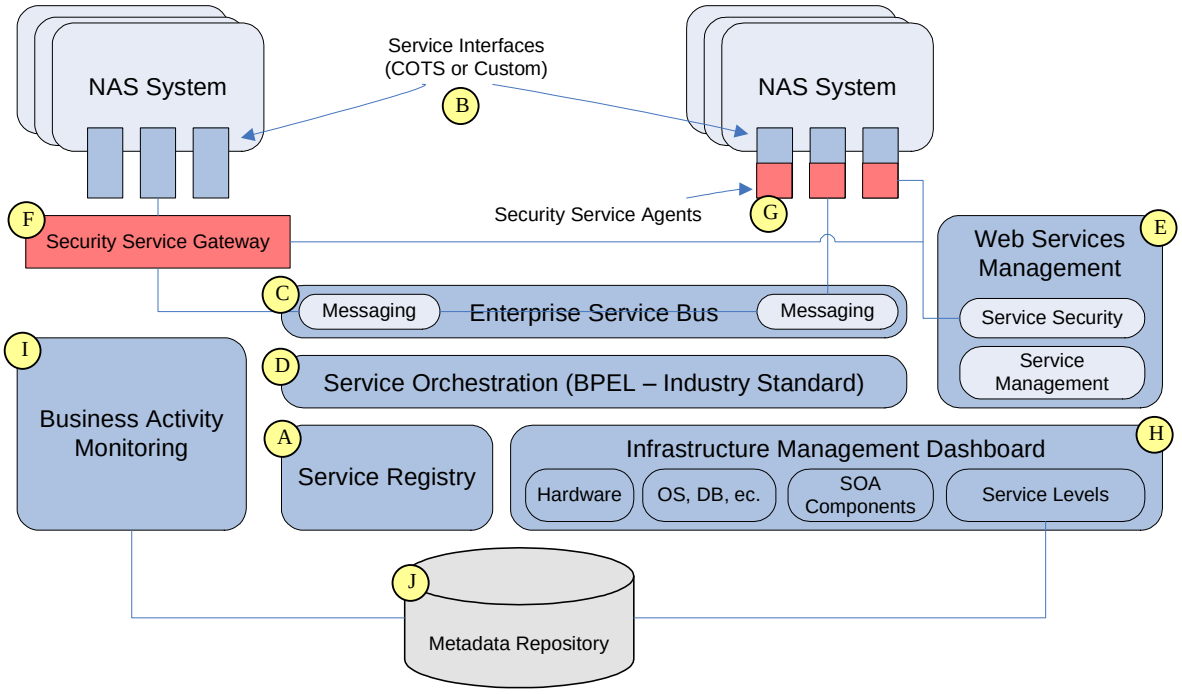


Figure 7: GEIA SOA Architecture for SWIM

The FAA organizes SWIM's initial Core Services into four groups. Below is a discussion of how the GEIA architecture illustrated above support those FAA groupings. The four high-level groups are below:

1. Interface Management
2. Messaging
3. Security
4. Enterprise Service Management

4.2.1 Capability #1: Interface Management

Label	Architectural Element	Description
A	Service Registry	A service registry provides a very important function in any SOA architecture – advertising the services that are available for reuse by applications. This is particularly useful at design time when software developers aim to reduce FAA cost and complexity by re-using existing services rather than creating redundant services. A registry can also be useful at runtime when an application dynamically "discovers" what services exist that meet a particular requirement. The industry standard for SOA service registries is called UDDI ("Universal Description, Discovery, and Integration"), and GEIA endorses the inclusion of a registry that includes UDDI-based interfaces in SWIM for the publishing and discovery of services.
B	Service Interfaces	FAA applications need to "plug in" to SWIM in order to achieve the FAA's vision for net-centric operations. GEIA encourages the FAA to consider two high-level strategies for these interfaces: essentially, the classic "build vs. buy" options. • On the "build" side, industry standards exist that provide a uniform mechanism for SOA applications to exchange information. One such example is JCA (Java Connector Architecture). JCA has the added benefit that other FAA SOA programs have adopted JCA as their standard for application interfaces, so SWIM's adoption of this same standard would improve interoperability between SWIM & these other programs. • On the "buy" side, various industry partners provide pre-built interfaces to a wide variety of data sources and/or application that may exist at the FAA. Through a balance of build and buy, the FAA will be able to establish connectors to its applications cost effectively and mitigating undue risk.

4.2.2 Capability #2: Messaging

Label	Architectural Element	Description
C	Enterprise Service Bus	An ESB is interoperable messaging platform that facilitates the exchange of data between target service endpoints while ensuring quality of service characteristics. While an ESB is a standard SOA component, there are several design considerations on how best to deploy an ESB at the FAA. Section 3.3.2 discusses those considerations.
D	Service Orchestration	The term "orchestration" refers to a high-level coordination of the fine-grained interactions – system interactions and human interactions – in order to achieve a higher-level FAA business service and ultimately and end-to-end business process. Industry has adopted an industry standard called BPEL (Business Process Execution Language) for SOA orchestration. Based on this BPEL standard, industry solutions exist that will allow the FAA to define orchestration patterns and monitor the actual flow of transactions through these patterns in day-to-day operations. By adopting BPEL-based orchestration, the FAA will reduce risk by establishing a standards-based service orchestration platform for declaratively defining the logic that controls service interactions.

4.2.3 Capability #3: Security

Label	Architectural Element	Description
E	Security Service: Manager	Three security capabilities are important: 1. Establishing policies for security 2. Enforcing those policies 3. Auditing compliance with those policies It is risky to leave security up to the whims of individual development teams. They may not be aware of current FAA security policies and therefore may (inadvertently) not enforce those policies properly in FAA application software. GEIA encourages the FAA to decouple security from the development of individual applications. Deploy an infrastructure that allows the FAA to establish security policies centrally, and then use "gateways" and "agents" (described below) to enforce those policies for the collection of FAA applications. As part of establishing security policies, FAA will need to establish a federated security model for SWIM. Many different FAA systems as well as external entities will need to access SWIM services. It is unrealistic to believe that maintaining a single security directory for all of these entities is feasible. As a result, the FAA will need to establish the appropriate trust relationships with these entities and leverage standards such as WS-Trust and SAML to allow these entities to interact with SWIM.
F	Security Service: Gateway	A security *gateway* is a collective solution – it provides a shared mechanism for enforcing FAA security policies across the group of applications at a particular SWIM endpoint (facility or cluster of servers). The security Gateway will play a key role in helping the FAA enforce federated security policies. The gateway will be responsible for receiving requests from external entities as well as other FAA systems and passing the security tokens to the security service for authentication and authorization.
G	Security Service: Agents	A security *agent* is an application-specific solution – it provides a mechanism for enforcing FAA security policies associated specifically with a particular application. It provides finer grained enforcement than gateways but still provides the important de-coupling of policies from application development described above in the Security Service Manager.

4.2.4 Capability #4: Enterprise Service Management

Label	Architectural Element	Description
H	Dashboard #1 **Infrastructure Management**	Reliability of SWIM will be vital to the FAA. Because SWIM will enable Air Traffic Control modernization, having a foundation that FAA applications can rely on as available, secure, and providing high performance is crucial to the achievement of true net-centric operations. To provide the FAA with comprehensive governance of SWIM infrastructure, this management dashboard should support the following elements: • Hardware: Monitoring the servers, storage, and network routers that comprise SWIM will be crucial to FAA's ensuring high availability of SWIM. • Operating System, Database, etc.: Monitoring the availability and patch levels of key system software – operating systems, databases, etc., – will provide the next level of SWIM quality assurance. • SOA Components: Monitoring the SOA elements comprising SWIM Core Services will provide the FAA with these capabilities: - Dynamically discover BPEL processes and the underlying services that are being orchestrated - Perform risk- and business impact analysis - Realize dependencies within a complex distributed SOA environment - Provide drill down capability to trace transactions through the environment to diagnose bottlenecks and system problems • Service Levels: Establishing SLAs may become necessary for the FAA to gain broad adoption of SWIM, and having the proper tools to *enforce* these SLAs will become a vital element to help the FAA fulfill its commitments. Modern IT dashboards exist that monitor compliance with SLA targets. • Charge Back: Having the ability to account for the usage of Web Services and allocate costs back to the users of those Web Services would provide a means for the FAA to fairly apportion SWIM costs.
I	Dashboard #1 **Business Activity Monitoring**	In addition to the system-level monitoring listed above, SWIM will also benefit from a second type of monitoring called "business activity monitoring". This can provide the FAA with a dashboard simulating a true end-user experience: reports of air traffic congestion, alerts to in-flight safety incidents, etc. By essentially "listening" to the real-time stream of FAA transactions flowing through SWIM – weather, surveillance, etc. – this second dashboard capability helps the FAA rise above the

		infrastructure to monitor the true end-user experience.
J	Metadata Repository	Stores "information about information" for key FAA data sets. This repository will help provide a common understanding of data elements across applications and data stores and assist the FAA in routing data to the right people and systems.

4.3 SOA Design Considerations

Several SOA components have robust enough capabilities that they merit an expanded discussion around options the FAA will have in implementing these components. This section of the document includes those expanded discussions.

4.3.1 Registry

A key component of a successful SOA implementation is a registry that includes interfaces based on the Universal Description, Discovery and Integration (UDDI) specification. A registry is essentially an online directory enabling service *providers* to advertise their offerings and allowing service *consumers* to find services that match their criteria. It provides a "white pages" listing of service providers, a "yellow pages" listing of the services offered, and technical information needed to access a service as defined in the Web Services Description Language (WSDL) document for that service.

Governance is another benefit of a registry, as it provides a central platform for services such as the following:

- Lifecycle management of services and resources
- Ensuring quality and external and internal standards compliance
- Notifying stakeholders of change
- Adherence to policy
- Access control to services
- Tracking additional metadata on services including such things as ownership, current users, status, plans, etc.

One important FAA consideration centers on *when* to use a SOA directory. Theoretically there are two times: (1) at build-time [by programmers] and (2) at run-time [by the application in production]. GEIA recommends that the FAA use a directory only at build-time. However, the registry technology should be capable of being used effectively at run-time in the future, especially for selection between multiple instances of a service that may be available in the infrastructure. Since usage will initially be most prevalent at build-time, an important feature would be the ability to store the actual artifacts for the service as well as the WSDL, to enable development and testing in additional applications.

4.3.2 Enterprise Service Bus

The use of an Enterprise Service Bus (ESB) provides a much-needed intermediary layer that facilitates data delivery, service access, service reuse, and service management of an enterprise SOA implementation. ESB also supports intelligently directed communication and mediates

relationships among loosely coupled and decoupled business components. SOA is a fundamental shift in the way applications are designed, developed, and integrated. It also facilitates the development of enterprise applications as modular business services that can be easily integrated and reused.

At the same time, it is important to acknowledge that SOA opens some unique challenges. The FAA can address these challenges using an ESB, as describes in the following points:

- **Reliable Messaging:** Reliable transport of data continues to be a basic need for any integration solution. While the principle of SOA calls for standards-based, platform-independent messaging protocols, this principle does not inherently allow for *reliable* delivery of data. Standards are emerging to support this capability, but they are not always mature or widely adopted. Some standards are industry-agnostic such as WS-RM. Other standards are unique to the aviation industry such as TypeX, a SOAP- and WS- based industry standard to be published as an IATA standard to enable reliable messaging based on IATA and ICAO addresses. Demonstrating the compatibility of these two types of standards (industry-agnostic & aviation), TypeX has been implemented by various integrators (e.g., SITA and ARINC) and some users. TypeX can be used in a SOA environment and plugged into ESB, and it makes use WS-* security specifications for security functions.
  ESB Benefits: A multi-protocol ESB can be used to support SOA messaging patterns and utilize technologies such as JMS (and commercial implementations such as Oracle AQ and IBM MQ Series) for guaranteed delivery and clustered topologies for availability at the middle tier and database layers.
- **Service Virtualization:** SOA implies a basic architectural paradigm in which any service consumer can access a service provider from any platform (within security constraints). This, in turn, implies that the appropriate protocol and syntactic mediation is in place to insulate consumers and providers. Service virtualization is the primary driver for implementing an ESB, and most other use cases are variations of it. Lack of clean layering, or "separation of concerns", at design time introduces unnecessary coupling between business logic and IT details. The impact of these cross-dependencies might not be noticeable at first, but as the integration scope grows, they start to erode the initial benefits of a SOA implementation.
  ESB Benefits: The ESB architecture removes all the point-to-point dependencies by providing an abstraction layer allowing the mediation of disparate data and protocols.
- **Policy Management:** Access by known and unknown service consumers results in the need for an abstracted policy management model that is capable of enforcing authentication, authorization, and encryption in addition to more complex business-level policies independent of the service provider implementation.
  ESB Benefits: Rather than hand coding these policies into each service, the ESB allows centralized configuration and auditing of security policies. This also provides a separation of duties between developers and the security implementation. Key standards include: LDAP, WS-Security, and SAML.
- **Management and Monitoring of Services:** An increasing number of services results in an increasingly complex environment. This environment must be monitored for availability, performance, and any technical or business-level errors.
  ESB Benefits: Service levels and quality-of-service monitoring should be managed at

the ESB layer as well, and integrated SOA Systems Management capabilities are crucial to administer the complexity of SOA.

- **System Heterogeneity:** Today's new applications are tomorrow's legacy, as one can observe in common applications as well as the software used to connect them. This proliferation of new technology is inevitable, and system landscapes must be architected to support such change. As modern application development technologies quickly evolve, the Model, View, Controller paradigm allows the UI layer to swap in and out without impacting business and data logic.
  ESB Benefits: The ESB provides all the services needed by today's application developers building rich, web-based applications on all kinds of platforms from Java frameworks to Adobe Flash and Flex.
- **Abstraction of Business Logic from Technical Implementation Details:** One goal of SOA is to provide a layered approach to developing systems that insulates changes in technology from changes in business process, and vice versa. In effect, this "separation of concerns" must be designed into the architecture from the start.
  ESB Benefits: A SOA environment with an ESB can provide this insulation between the service consumer and provider. As technologies and end-points change, this can be managed centrally in the ESB and changed in one place providing the agility to adapt to changing requirements or implementation technologies.

4.3.3 Legacy Integration

Aviation Context

Like any large organization, the FAA hosts a wide range of IT applications that – individually – effectively perform their functions but – collectively – represent complexity that slows modernization. And yet modernization is necessary. This situation represents the classic problem of "changing the tires on a moving car" – how can the FAA migrate to a modernized net-centric set of applications (NextGen) in a graceful fashion that preserves continued operations of the existing NAS systems?

Legacy integration includes communications with FAA legacy systems and those with their business partners, airlines, airports and other Air Navigation Service Providers (ANSPs). These communications related to the exchange of operational messages such as flight plans, weather messages etc. are done either through AFTN (Aeronautical Fixed Telecommunication Network) or from airlines TypeB to AFTN. ICAO – the International Civil Air Organization – has also recommended a standard called AMHS (Aeronautical Message Handling System) that is under deployment by ANSPs to replace AFTN. There are discussions form new messaging recommendations in the move to XML and rich formats. In all cases gateways need to be specified to bridge the legacy and new environment that converts both protocols and business data formats to allow seamless interoperability. In some cases where similar functionality or data is not available, the FAA may need to make certain trade-offs and implement workarounds.

Solution Alternatives

Given these legacy systems, a perennial challenge for the FAA is finding a way to reduce the effort, cost and risk involved in developing and maintaining integration among legacy applications and "new technology" applications.

Two solution alternatives exist, and the FAA should consider both:

- **COTS Adapters:** Just as industry has long provided COTS *applications*, it is now possible to obtain COTS *integration* solutions between specific end-points. FAA's evaluation of this alternative should be identical to any other build/buy decision – evaluate the quality, cost, and risk associate with the build- vs. buy- solutions. Some adapters are to generic data sources (database, file transfer, Java messaging, etc.) while others are to true end-user applications (often back-office applications like Human Resources, Customer Relationship Management, etc.). On step in the FAA's due diligence should be to evaluation whether the use of any COTS adapters could reduce the cost and risk associate with legacy integration.
- **Development Standards:** If a COTS adapter does not exist for a particular FAA application, the other alternative is developing a SOA-based adapter using standards like JCA to provide real-time, bi-directional, and comprehensive connectivity. Adapters are usually metadata-driven and integrate with one or more recommended backend application programming interfaces (APIs). They translate the data from backend specific data format to a standard data representation like XML. This enables reuse of existing assets by exposing them as Services that can be integrated with new applications providing the "last mile of integration". By exposing the underlying backend applications as Services and expressing them as WSDLs via an ESB, they are available to SOA clients across the network. Some of the key standards include: Web Service Definition Language (WSDL), Web Service Invocation Framework (WSIF), Java Connector Architecture (JCA) and XML. The importance of standards in these essential areas should not be underestimated and prevents Vendor lock-in.

4.3.4 Security

Because of its loosely coupled connections and its use of open access (via Hypertext Transfer Protocol [HTTP]), SOA adds a new set of requirements to the security landscape. Below are descriptions of security standards that the FAA should consider and best practices regarding the online enforcement of security policies.

4.3.4.1 Security Standards

Many organizations rely on the Secure Socket Layer (SSL) protocol to protect access to SOA deployments. SSL provides authentication, confidentiality and message integrity. However, when the data is not "in transit," the data is not protected, which makes the environment vulnerable to attacks in multi-step transactions. As a result, there is a need to address more specific SOA security challenges by relying on additional, application-level industry standards. Figure 8 includes a sampling of these security standards.

Security Area	Standards for FAA Consideration
Content Security	• XML Encryption • XML Signature
Message-Level Security	• WS-Security
Metadata	• WS-Policy • WS-PolicyAssertions • WS-PolicyAttachment • WS-SecurityPolicy
Trust Management	• SAML • WS-Trust • WS-SecureConversation • WSFederation
Public Key Infrastructure	• PKCS • PKIX • XKMS

Figure 8: Security Standards

Some of these standards have been around for several years, originally designed for web applications and later leveraged by SOA, for example SSL (mentioned above), and Kerberos, a cross-platform authentication and single sign-on system. Other standards have specifically been created to provide security to networks of web services, for example WS-Security and WS-Policy. Most SOA industry standards are defined in XML frameworks. The last few years have seen the emergence of a plethora of XML-based specifications addressing various aspects of SOA security. Most of these specifications are part of the so-called WS-* (Web Services specifications) stack.

Most WS-* specifications started as proprietary industry initiatives. Some of these specifications (e.g., WS-Security, WS-Trust, and WS-Policy) have been transferred over to standards bodies such as the Organization for the Advancement of Structured Information Standards (OASIS) or the World Wide Web Consortium (W3C). WS-* specifications often depend on each other, for example, WS-Policy uses WSPolicyAssertions. WS-* specifications also leverage non-WS specifications, for example, WS-Security uses XML Encryption and XML Signature.

4.3.4.2 Security Policy Enforcement

Until recently, the onus was on individual developers to embed security code directly into web services they developed. This approach is fine when the number of developers is small (e.g., 5 people) and the number of SOA services is small (e.g. 10 services). A number of factors emerge, however, that make this "embedded security" approach inappropriate:

- **Increasing Security Requirements:** Requirements such as cryptographic data protection, identity management, and governance are beyond the skill-sets of many developers. Assuming that each developer will diligently and accurately implement the wide array of FAA security policies is a risk proposition.

- **Increasing Development Staff:** As SOA takes hold in the FAA, the number of developers implementing SOA-based applications will increase dramatically. These developers will work on many different FAA programs and will be employees of many different organizations (FAA and integrators). Placing the responsibility of implementing FAA security policies in the hands of this very diverse and distributed team provides a difficult governance challenge to the FAA.
- **Increasing System Complexity:** Because SOA creates a net-centric environment its complexity is affected by the number of systems (nodes) in the network. Once the FAA establishes SWIM as the enterprise-wide standard for net-centric NAS operations, the number of FAA systems participating in this community will grow. This growth itself will bring new security challenges as the number of service *providers* and *consumers* will grow correspondingly. This growth will necessitate the development of a federated security model across all of the systems interacting with SWIM.

A solution to these three challenges lies in decoupling the security from the individual web services. Rather than asking programmers to embed security logic in each individual web service, the FAA should have the capability for its security organization to do the following:

- *Define* security policies
- *Establish trust relationships* and a federated security model
- *Enforce* security policies (at the perimeter or in specific web services)
- *Audit Compliance* with security policies

4.3.5 Orchestration

Various activities are executed to implement the runtime lifecycle of a business process. Process orchestration is when a central process coordinates the execution of different Web services operations. The central conductor (as in an orchestra) is aware of the overall goal of the orchestration, the operations involved, and the order of the operation invocation. This centralized management allows Web services to be added or removed without each being aware of its effect on others, as well as compensatory processes to be implemented in case of faults and exceptions.

Process orchestration and process choreography are similar only to the degree that they describe service interfaces and interactions. For the purposes of this discussion, we will focus on process orchestration. An industry standard for encoding business orchestration logic is called the "Business Process Execution Language" (BPEL). When you use BPEL for orchestration, the BPEL language provides a standard for controlling the overall sequence of operations, invoking services and executing on a BPEL server.

A business process is a series of steps required to implement a business function. The business process may include system interactions and/or human interactions, within an enterprise or extending across corporate boundaries. Some enterprises merely document their business processes, for consistency or regulatory purposes. However, the past several years have seen an increase in the number of organizations that use IT systems to execute and monitor their business processes in an automated fashion. One of the key drivers behind this accelerating adoption of business process management (BPM) is the emergence of broadly accepted standards such as WS-BPEL for implementing and executing business processes. BPEL defines an XML-based language for specifying the behavior of business processes that are

based on Web services. The first public BPEL specification was released in July 2002, as BPEL4WS version 1.0, combining IBM's Web Services Flow Language (WSFL) and Microsoft's XLANG specification. Version 1.1 was released in 2003, collaborated on by many industry vendors, and version 2.0 of the WS-BPEL standard was released from the OASIS standards body in April 2007.

The BPEL standard enables both interoperability and portability for an organization's business processes to a degree not previously possible. Interoperability enables business processes operating on different business process engines to interact with each other and with services published from many vendor and open source platforms. This is possible because BPEL is layered on top of the lower levels of the Web services stack — namely, SOAP, UDDI, WS-Addressing, WSDL, XML, XML Schema, and others. Interoperability in the arena of business processes has been possible for several years; however, the BPEL and Web services standards take it to the next level. But beyond interoperability, the concept of portability for business processes is truly something new. The process portability offered by BPEL ensures that business processes can be defined using tools that support the standard and then run on any compliant systems. Today, BPEL engines are available from large platform vendors such as Oracle and IBM as well as from smaller niche vendors and open source companies.

4.3.6 Infrastructure Management

Ideally, the FAA's SOA infrastructure should provide a platform for management features such as the following:

- Monitoring and diagnostic
- Externalized security
- Centralized auditing and logging
- Service level agreements
- Policy management (definition & enforcement)
- Scalability
- High Availability
- Disaster Recovery / Continuity of Operations (COOP)

How to achieve these capabilities will vary greatly from one IT vendor to another, and from one integrator organization to another. Because this particular topic is very closely coupled with the particular solution deployed, it is impossible for GEIA to make very specific recommendations without favoring one vendor's solution over another. Consequently GEIA simply recommends that future FAA solicitations invite the vendors to describe how their respective solutions provide these important capabilities.

5 Best Practices

In addition to the component-level recommendations described above in Section 4, there are also higher-level best practices the FAA should consider. Those appear here in Section 5.

5.1 SOA Maturity Model

Large organizations like the FAA are in different stages of migrating to SOA. Typically organizations embrace a new framework like SOA in stages – "testing the waters" with a pilot project, then incrementally expanding the solution toward a goal of enterprise-wide adoption.

Mitigating risk is crucial in this evolution, and one means of mitigating risk is through the use of a SOA Maturity Model. Benefits to the FAA of using a SOA Maturity Model include:

- Situational Awareness: Understanding the typical stages of SOA adoption
- FAA Maturity Level: Baselining the FAA's current level of SOA maturity
- Best Practices: Based on the FAA's specific maturity level (#2 above), indentifying the specific best practices *at that level of maturity* to move the FAA toward success
- Long-term Planning: Aiding FAA strategic success by looking beyond the FAA's immediate (tactical) SOA goals toward what the FAA will likely encounter as it hits higher levels of SOA maturity in the future.

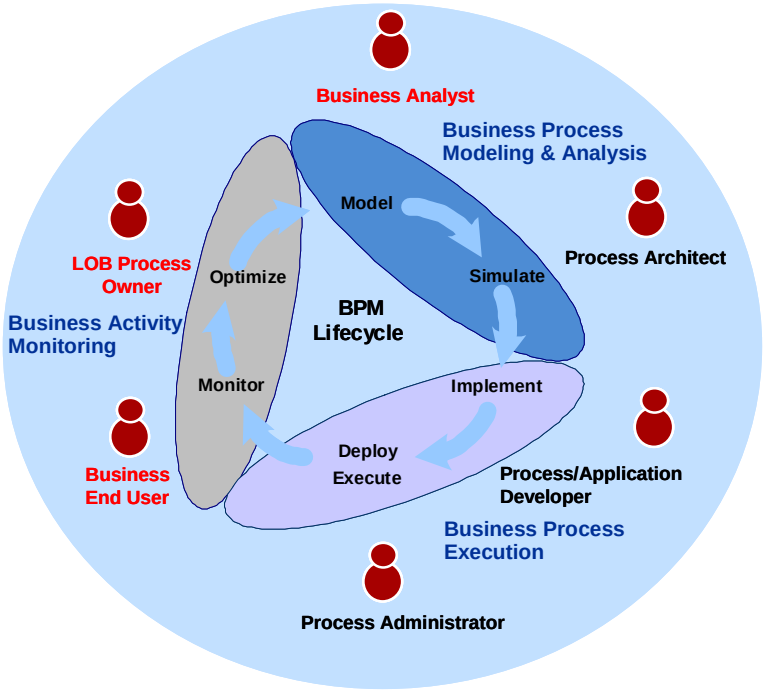
Several such models exist, and GEIA is not taking a position on which maturity model is best. To help the FAA visualize a maturity model, one representative example appears in Figure 9.

Figure 9: SOA Maturity Model

5.2 Business Process Management

> **A Closer Look:**
> This topic merits additional discussion among FAA stakeholders regarding the proper time and place to incorporate these ideas into the overall FAA modernization vision – as indicated in the whitepaper's Preface.

Business Process Management (BPM) governs an organization's cross-functional, end-user focused, end-to-end core business processes. BPM typically occurs in a cycle, as illustrated in Figure 10.



Collaboration of Business and IT

- ✓ **Business Process Owners**
  Set objectives and policies
- ✓ **Business Analysts**
  Modeling and analyze processes
- ✓ **IT Architects and Developers**
  Build, deploy and administer processes
- ✓ **Business Users**
  Operate and monitor processes – focusing on improving efficiency and identifying bottlenecks

Figure 10: Business Process Management – Lifecycle

Three phases comprise this cycle. Below is a brief description of each:

- Business Process Modeling and Analysis
  Business Process Modeling is a subset of Business Process Management. It creates the "as is", "what if" and "to be" scenarios for a planning and implementation. The FAA has established Communities of Interest (COIs) as an outreach mechanism into the end-user organizations to engage them in this modeling and analysis phase.
- Business Process Execution
  A business process is a series of steps required to implement a business function. The business process may include system interactions and/or human interactions, within an

enterprise or extending across corporate boundaries. The past several years have seen an increase in the number of organizations that use IT systems to *execute* their business processes in an automated fashion. A key driver behind this accelerating move toward execution of business processes is the emergence of broadly accepted standards such as WS-BPEL for implementing and executing business processes.

- Business Activity Monitoring
  FAA management in groups such Enroute, Terminal, etc. may gain a broader ability to monitor global aeronautic activity in real-time. The industry term "business activity monitoring" (BAM) refers to this capability. Because the SWIM infrastructure will include a shared framework for access to a wide variety of FAA information (weather, surveillance, etc.), FAA management may be able to use BAM tools to see a dashboard of key aeronautical information to which they have subscribed. While existing FAA programs such as ETMS and STARS already provide such monitoring capability today, SWIM opens the possibility of *broadening the access* to this information to a larger audience (within appropriate FAA security policies).

An integrated, standards-based BPM tool will help the FAA capture end-user processes and then automate the delivery of those processes into a production environment rapidly and securely.

5.3 Building a Service Portfolio

A Closer Look:
This topic merits additional discussion among FAA stakeholders regarding the proper time and place to incorporate these ideas into the overall FAA modernization vision – as indicated in the whitepaper's Preface.

Two key activities support building a service portfolio. Their descriptions appear below.

5.3.1 Service Profiling

The driving factor for building a portfolio of services is typically recognition of the need to align business needs with IT projects. The process typically evolves from an initial determination of required services, to discovery and classification of services and resources they depend on, such as policies defining specific business rules. The outcome, ideally, is a set of service-oriented business applications that can be modified and re-used to meet the changing business demands of the enterprise. The FAA is using its outreach to "Communities of Interest" (COIs) as the mechanism for this IT/Business alignment.

The first logical step in building a service portfolio is to determine what services are needed. Three proven techniques used to identify and discover candidate services include **top-down analysis**, **bottom-up analysis**, and **business process tracing**. Note that these techniques should be considered complementary, rather than exclusive, and should all play a role in the FAA's service discovery process. Descriptions of each technique appear below:

- **Top-down Analysis:**
  A top-down analysis will typically net a substantial set of candidate services aligned with business needs. However, this process alone will not uncover all the candidate services within an organization. For help in this area, the next step is to go bottom-up.
- **Bottom-up Analysis:**
  A thorough examination of existing services as well as IT infrastructure, application functionality, and data that have been previously used by business applications should be performed next. This bottom-up approach will typically yield a rich set of high-level as well as low-level candidate services.
- **Business Process Tracing:**
  As a complementary effort, you should trace the lifecycle of each business event to discover what services are needed to process the event through its lifecycle. This process, known as business process tracing, will not only uncover the services needed to process the event, but may also expose service candidates overlooked by a pure top-down or bottom-up approach.

The net sum of these service discovery efforts will be a conceptual service portfolio containing candidate services that are most likely needed as projects.

5.3.2 Service Categories

Several categories of services will likely emerge from the profiling exercise described above. Based on joint GEIA & FAA discussions, SWIM will likely have (at least) these three different categories of services. These FAA-specific categories are a variation on the customer-neutral categories listed earlier in section 3.1 of this document. See Figure 11 for details.

Service Category	Description	Example
Infrastructure Services	These will provide low-level services for all SIPs will use in general operation and interaction with other SIPs.	• Messaging (e.g., publish/subscribe) • Security (e.g., authentication) • Interface Mgmt (e.g., discovery)
Business Services – Local	These will provide business functionality for a single SIP.	• Determine Enroute ETA
Business Services – Global	These will provide business functionality for multiple SIPs.	• Severe Weather Notification • National Emergency Broadcast • Flight Status

Figure 11: Service Categories

Policy Impact: The FAA should think carefully about what policies to establish regarding SIPs' development and use of SWIM services. The three categories above are logical but not necessarily intuitive to every SIP without a concerted effort by the FAA to educate stakeholders about the differences among these categories. SIPs will have questions such as:

- When should I declare a new service an *infrastructure* service vs. a *business* service?

- When should I decline a new business service *local* vs. *global*?
- When I search for an existing service, where should I look first: *local* or *global*?
- When I change a service, how do I minimize disruption to SIPs using this service?

GEIA is writing a separate whitepaper on governance, and that whitepaper will likely provide guidance on these sorts of topics. For the purposes of the current whitepaper it is important to acknowledge that these topics exist for the FAA to consider.

5.3.3 Service Granularity

Striving for an appropriate level of granularity will maximize ease of use, reuse, and manageability. Building on the "Service Profiling" section earlier in this document, the methodology an organization follows to identify service candidates often leads to two different levels of granularity:

- **"Top Down" → Course-grained Services:**
  A top-down, business-driven analysis will typically uncover high-level (coarse-grained) services that map to existing functional requirements.
- **"Bottom Up" → Fine-grained Services:**
  A bottom-up analysis will typically yield a significant number of fine-grained services that reflect existing systems, processes, and operating procedures.

Both levels of granularity have advantages and disadvantages, so it is impossible to make a simplistic blanket recommendation of which level the FAA should use. Below are some pros and cons of each:

Service Granularity	Pro	Con
Course Grained	High-level services are more agile in a dynamic business environment as they are not tightly coupled to underlying infrastructure.	Coarse-grained services tend to lock in existing business process automation.
Fine Grained	Finer grained services can more readily be recombined to create new patterns.	Low-level services are tightly coupled to underlying infrastructure or application programming interfaces (APIs) and cannot easily be modified to suit changing business requirements.

The overall goal is combining a set of fine-grained services into higher-level coarse-grained services. Where the appropriate "boundaries" are for the FAA, SWIM, and SIPs is still up for discussion, and this is a fine and appropriate discussion to have for any organization in the early stages of adopting a SOA framework for modernization.

5.4 Design Patterns

With an increasingly large portfolio of services, effectively leveraging and reusing these services can prove challenging unless you follow the correct architectural patterns.
The first, and most crucial decision when designing a SOA is to identify the appropriate conditions under which a service should reside either on the bus or (for example) in a Business Process Management (BPM) tool. While on the surface these two capabilities may seem quite distinct, it is quite possible (and in some cases even preferable) to architect bus-like capabilities from a BPM tool. In general, the following guidelines apply here:

- Services on the ESB should be short-lived, discrete transactions
- Services requiring sophisticated rules to determine routing, access, and other data-based decisions should reside on the ESB. This may still be a "short-lived" transaction (per the previous bullet point) but one that has complex logic associated with it that is best fulfilled by a shared resource (ESB) rather than any single end-point application participating in the interaction.
- ESB services should contain no human workflow activities. Instead, an "orchestration" layer (based on the BPEL standard) would handle any human workflow.
- BPEL processes should parallel business processes
- Lower level logic should be built out in services (not in BPEL)

Beyond these high-level guidelines, there are lower-level design and development guidelines that you should implement globally. Some of these guidelines are not unique to ESB, but are made more relevant as a result of ESB.

- For each application, the appropriate adapter must be identified, along with the corresponding protocols and enveloping technique that must be applied. This includes identification of any standards that are required in the interaction (WS-Reliable Messaging and TypeX, for example). Be sure to consider transaction model (asynchronous or synchronous) as well as reliability needs when selecting the adapters and protocols.
- The source and target data objects involved must be defined, in detail, and mapped to a corresponding SOA object that the various SIPs can share (when appropriate).
- Design metrics into the service, then use "business activity monitoring" (BAM) dashboards to obtain easy access to real-time metrics.
- Identify security restrictions at all levels. These levels include the application, transaction, data object content, and potentially data field content. Security policies must also be defined to support these requirements.
- Leverage existing services where possible. All enterprise-scale SOAs should have a searchable registry of services.
- Identify auditing requirements, as well as technical and business-level error handling requirements and design services accordingly.
- When deploying the service, define and publish the WSDL. Register the service with the enterprise service registry.

5.5 Standards-based Integrated Solutions

Services Oriented Architecture is now becoming the mainstream for enterprise applications. The key to any successful SOA environment is the framework on which it is built. The most resilient framework is based on market-leading open standards. Support for key standards such as JAX-WS, BPEL, WSReliableMessaging, WS-Addressing, TypeX, SOAP with Attachments, MTOM, WS-Policy, UDDI, WS-Security and SCA as essential building blocks is a necessary foundation for the next generation of successful applications. In fact, without a robust, standards based platform that is directly focused on interoperability, it is impossible to build new composite applications using services. Other standards have specifically been created to provide security to networks of web services, for example WS-Security and WS-Policy. Most SOA industry standards are defined in XML frameworks. The last few years have seen the emergence of a plethora of XML-based specifications addressing various aspects of SOA security. Most of these specifications are part of the so-called WS-* (Web Services specifications) stack.

5.6 Federal SOA Guidelines

Because the FAA is a federal agency, it is worthwhile to review any best practices developed by other federal organizations. Brief references to two appear in Figure 12.

Federal Guideline	Description
SOA Predictive Metrics	On October 15, 2007, NIST published a framework called SOA Predictive Metrics. NIST intends this framework to provide a beneficial structure to Chief Architects and Program Managers across federal agencies that are building or evolving SOA and Enterprise Architecture (EA) systems. Reviews of this framework at a recent e-Government Conference indicated that SOA Predictive Metrics was particularly helpful in analyzing a program's Technology Baseline and Development Plan. http://www.antd.nist.gov/%7Esalasin/SOAMetrics
Practical Guide to Federal Service Oriented Architecture	On November 13, 2007, the Federal CIO Council published a pre-release (version 0.7) of a new document called Practical Guide to Federal Service Oriented Architecture. As noted in that document's Introduction, "This Practical Guide to Federal Service Oriented Architecture has been written to help federal chief architects and chief information officers in their efforts to adopt SOA best practices to further their organizations' mission outcomes, meet increasingly demanding compliance requirements, and optimize their IT architectures."

Figure 12: Federal SOA Guidelines

6 Appendices

6.1 Glossary

Below are key terms this whitepaper uses – or are part of the FAA and SOA focus area of this whitepaper – along with the definitions we intend them to have.

Term	Definition
Business Activity Monitoring	End-user capability to observe the "state of the business" in near-real time based on a live connection to the flow of business transactions flowing through an enterprise. For the FAA an example could be monitoring flights via a "tap" into the SWIM flow of surveillance information.
Business Process Management	Three stages typically comprise this: (1) *modeling* an organization's business processes, (2) *executing* those processes via some workflow technology, and (3) monitoring these processes and their business impacts.
Communities of Interest	These represent key stakeholder groups related to a particular discipline: e.g., Weather or Flight-and-Flow. The SWIM program is using these COIs to help define requirements for SWIM.
Enterprise Service Bus	A technology capability to exchange messages among a pool of software applications to facilitate net-centric operations. The FAA could deploy one ESB *enterprise-wide* or the FAA could deploy *multiple* ESBs (per SIP, for example) in a more federated architecture.
Federated Architecture	A solution structure that is not centralized but rather has a variety of distributed programs (e.g., SIPs) that cooperatively implement a set of "local" capabilities that collectively work together to support an enterprise goal.
Granularity	The "size" of a service in a Service Oriented Architecture. Typical levels include "course-grained" (big) and "fine-grained" (small). Course-grained services typically map to top-level business functions whereas fine-grained services typically map to task- or system-level activities.
Next Gen	The FAA long-term vision for modernization of the National Air Space (NAS) to support a greater volume of flights via cooperative net-centric operations of NAS systems.

Orchestration	The coordination of discrete tasks into a higher-level business process (e.g., approval of a new FAA flight plan.) Orchestration often has decision-trees and knows how to dispatch the flow of activity to the underlying people and systems fulfilling the discrete tasks.
Registry	A SOA component that lists the "pool" of available services. Developers can use a registry to save cost via re-use of an existing service, and certain runtime environments use a registry to dynamically find the best service to fulfill a certain requirement.
Service	The "unit of work" within a Service Oriented Environment. Software programs can advertise the set of services that they provide, and then other FAA programs can use the SOA solution to find and use these services.
Service Container	An FAA concept for logically grouping certain SOA capabilities within each SWIM Implementing Program (SIP) and using this Service Container as a vehicle for SIP applications to access SWIM Core Services.
Service Oriented Architecture	A software framework that can let the FAA develop modular and re-usable software by identifying required capabilities, developing these capabilities and placing them into a shared "pool", and then using these pooled services in higher-level NAS applications.
SWIM	System Wide Information Management – an FAA program targeted at laying a SOA foundation to enable the long-term NextGen vision of net-centric operations for moderations of the National Air Space.

6.2 FAA SWIM Acronyms

Below is a list of key acronyms the FAA uses to discuss operations and the environment SWIM will support. Not all of these terms arise in the current whitepaper, but they appear here for completeness and to support future discussions.

ADDS.....Aviation Digital Data Service
ADOC .....Airline Direct Operating Cost
AIM .....Aeronautical Information Management
AOC .....Airline Operating Center
ARMT.....Airport Resource Management Tool
ARTCC .....Air Route Traffic Control Center
AS .....Application Server
ASDE-X .....Airport Surface Detection Equipment – Model X
ATC.....Air Traffic Control
ATCT .....Air Traffic Control Tower
ATO .....Air Traffic Operations
AWC .....Aviation Weather Center
BPEL .....Business Process Execution Language
BPEL4WS .....Business Process Execution Language for Web Services
BPM .....Business Process Management
CA .....Certificate Authority
CBR .....Content Based Routing
CDM .....Collaborative Decision Making
CERAP .....Center Radar Approach Control
CIWS .....Corridor Integrated Weather System
CMP .....Configuration Management Plan
COI .....Community of Interest
CORBA .....Common Object Request Broker Architecture
COTS .....Commercial off-the-Shelf
CP .....Central Processor
CPMP.....Commercial Product Management Plan
CSIRC .....Computer Security Incident Response Center
DAFIF .....Digital Aeronautical Flight Information File
DNS .....Domain Name Service
DOTS .....Dynamic Ocean Track System
EAP.....Extensible Authentication Protocol
EFSTS .....Electronic Flight Strip Terminal System
ERAM .....En Route Automation Modernization
ESM .....Enterprise Service Management
ESP .....Encapsulating Security Payload
ETE .....End-to-end
EVM .....Earned Value Management
FAA .....Federal Aviation Administration
FBWTG .....FAA Bulk Weather Telecommunication Gateway
FDIO .....Flight Data Input Output
FID .....Final Investment Decision
FSS .....Flight Service Stations

FTI .....FAA Telecommunications Infrastructure
FTP.....File Transfer Protocol
FY .....Fiscal Year
GCNSS .....Global Communications, Navigation, and Surveillance System
HADDS.....Host Automation Data Distribution System
HIDS .....Host-based Intrusion Detection Sensor
HT .....Hypertext and Transfer Protocol
IETF .....Internet Engineering Task Force
IKE .....Internet Key Exchange
ILS.....Integrated Logistics Support
ILSP .....Integrated Logistics Support Plan
IOC .....Initial Operating Capability
IOT&E .....Independent Operational Test and Evaluation
IOTRD .....Independent Operation Test Readiness Decision
IP .....Internet Protocol
IPCP .....Internet Protocol Control Protocol
IPS .....Internet Protocol Service
IPSec .....IP Security
ISD .....In-Service Decision
ISR .....In-Service Review
ISS .....Information Systems Security
IT .....Information Technology
ITWS .....Integrated Terminal Weather System
J2EE .....Java 2 Platform, Enterprise Edition
JMS .....Java Messaging Service
JPDO .....Joint Planning and Development Organization
LAN .....Local Area Network
LDAP .....Lightweight Directory Access Protocol
MADE .....Military Airspace Data Entry System
MOU .....Memorandum of Understanding
MQ .....Message Queuing
NACO .....National Aeronautical Cartographic Organization
NAIMES .....NAS Aeronautical Information Management Enterprise System
NAS .....National Airspace System
NASE .....NAS Adaptation Services Environment
NASR .....National Airspace System Resources
NextGen .....Next Generation Air Transportation System
NGATS .....Next Generation Air Transportation System
NIDS .....Network Intrusion Detection Sensor
NIST .....National Institute of Standards and Technology
O&M .....Operations and Maintenance
OT&E .....Operational Test and Evaluation
PDC .....Pre-Departure Clearance
PDR .....Preliminary Design Review
PHA .....Preliminary Hazard Analysis
PIREP .....Pilot Report
PKI .....Public Key Infrastructure
POC .....Point of Contact
PP .....Protection Profile

PTR .....Program Trouble Report
PVT .....Passenger Value of Time
QAP .....Quality Assurance Plan
QoS .....Quality of Service
RFC .....Request for Comment
RMI .....Remote Method Invocation
RVR .....Runway Visual Range
SA .....SWIM Adapter
SAML .....Security Authorization Markup Language
SAMS .....Special Use Airspace Management System
SD .....Situation Display
SDP .....Service Delivery Point
SEC .....Systems Engineering Council
SIG .....Security Incident Group
SIP.....SWIM Implementing Program
SLA .....Service Level Agreement
SMTP .....Simple Mail Transfer Protocol
SOA .....Service-Oriented Architecture
SOAP .....Simple Object Access Protocol
SOW .....Statement of Work
SNMP .....Simple Network Management Protocol
SRVT .....Safety Requirements Verification Table
SMS .....Safety Management System
SRMGA .....Safety Risk Management Guidance for Acquisitions
SSAR .....System Safety Assessment Report
SSD .....System Specification Document
SSH .....System Safety Handbook
SUA .....Special Use Airspace
SWIM .....System Wide Information Management
TBD .....To Be Determined
TCP .....Transmission Control Protocol
TDDS .....Terminal Data Distribution System
TDLS.....Terminal Data Link System
TFM-M .....Traffic Flow Management – Modernization Program
TFMS .....Traffic Flow Management System
TRACON .....Terminal Radar Approach Control
TSG .....Telecommunications Service Group
UDDI .....Universal Description, Discovery, and Integration
UDP .....User Datagram Protocol
URI .....Uniform Resource Indicator
URL .....Uniform Resource Locator
USNS .....United States Notice to Airmen (NOTAM) System
VNTSC .....Volpe National Transportation Systems Center
VPN .....Virtual Private Network
VRTM .....Verification Requirements Traceability Matrix
WAN .....Wide Area Network
WARP .....Weather and Radar Processor
WINS .....Weather Information Network Server
WJHTC .....William J Hughes Technical Center

WMSCR .....Weather Message Switching Center Replacement