

Service Oriented Architecture Compliance: Initial steps in a longer journey

Primary Author:
J Falkl (jfalkl@us.ibm.com)
12/19/2005

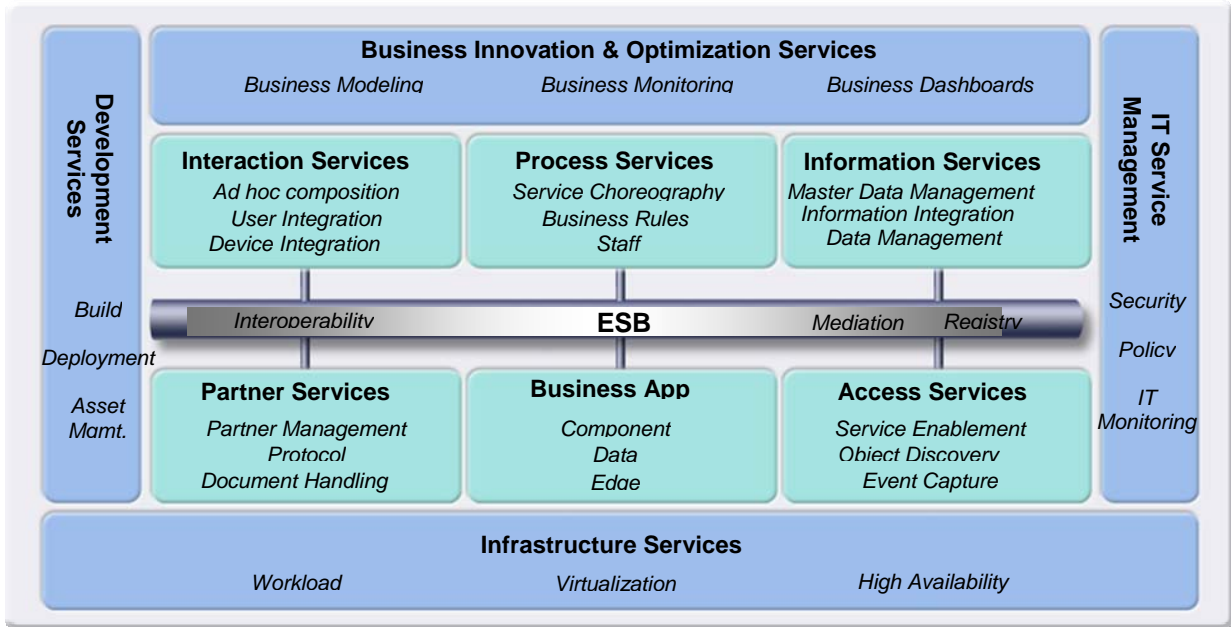
Contributors

Rob High
Christina Lau
Stefan Puehl
Angel Luis Diaz
Anant Jhingran
Don Ferguson
Kelvin Lawrence
Marcia Stockton
Steve Graham
Kareem Yusuf

WHAT IS SOA?

A *Service-Oriented Architecture*, or SOA, is an architectural style for creating an Enterprise IT Architecture that exploits the principles of *service-orientation* to achieve a tighter relationship between the business and the information systems that support the business. By service-orientation, we mean a way of integrating your business as a set of linked services. We define a service quite generally as a repeatable task within a business process. Note, although many people equate SOA with Web services, we regard Web services as one (very important and very popular) approach to building a SOA – certainly not the only approach.

With SOA, you can re-work a business process from the top down (called Business Transformation) based on a new business model or approach to the problem, or start from the ground up with information technology (IT) re-engineering to say, reduce costs. Many organizations use a meet-in-the-middle approach to move to a new realization of a business process by reusing existing assets and incrementally reworking these assets into a collection of services and service enabling components. With that in mind, it's important to realize that you can implement facades for traditional systems (e.g. CICS) and make them appear as loosely coupled services that can be used in an SOA. That alone does not make the CICS application SOA compliant; however it does make the business logic embodied by the CICS system capable of being a service and participating in an SOA.



The IBM SOA Reference Architecture

SOA AND STANDARDS

To define compliance, we need to establish a baseline set of criteria and standards to comply against. This is an interesting but difficult assignment because there are many different perspectives to consider (business vs. technical, industrial compliance, etc.). It's fair to say that some minimal characteristics are important in SOA, and the trick is establishing the boundaries surrounding compliance. The following is a short list of standards that could form this baseline.

- Technological: Web services standards, various WS-I profiles such as Basic Profile (BP) and Basic Security Profile (BSP), the IBM Community Centric Profiles (RAMP, etc.) etc.
- Business: Process Modeling formats, service artifact as the modeling artifact, etc.
- Architectural: loose coupling, typed componentization, composability, etc.
- Deployment and management: best practices, ITIL standards

SOA AND INFORMATION MANAGEMENT

SOA significantly drives a maturity model for information management. As a company embarks down an SOA path, it's important to ensure that your business information is clearly defined and managed, since SOA forces data uniqueness as information flows thru the service model. Consider what happens if two services refer to different pieces of data by the same name, or the same piece of data by different names. Planning is needed to ensure a successful outcome. To that end, we need to consider information management maturity and compliance in this effort as well as data governance.

GOVERNANCE AND SOA COMPLIANCE

It's easier to talk about SOA compliance if you first have a governance model defined. A governance model should allow you to:

- Define overall business goals
- Develop policies to achieve those goals
- Define rules/micro policies that are used to enforce policies
- Implement management capabilities to monitor compliance to policies
- Govern the above in a lifecycle model

So, once goals and policies are defined, you have a baseline of compliance criteria to which your services and composite applications can comply.

Since we are just starting definition work surrounding SOA Governance, we would like to position this paper as a generic 'starter set' of compliance criteria that can be used as an initial baseline for SOA compliance. This baseline will grow in detail as governance and compliance mature.

WHY COMPLIANCE AND NOT CONFORMANCE?

The definitions for compliance and conformance are:

Compliance usually implies adherence to a strict standard or regulation. It implies that there is some objective, widely-recognized standard and you either pass or fail. I.e. a company can be in compliance with Sarbanes-Oxley auditing requirements. A company's financial standards can comply with generally accepted accounting standards. A company can comply with a country's Export Regulations. A browser can comply with a specific security requirement by providing 128-bit security and TLS encryption.

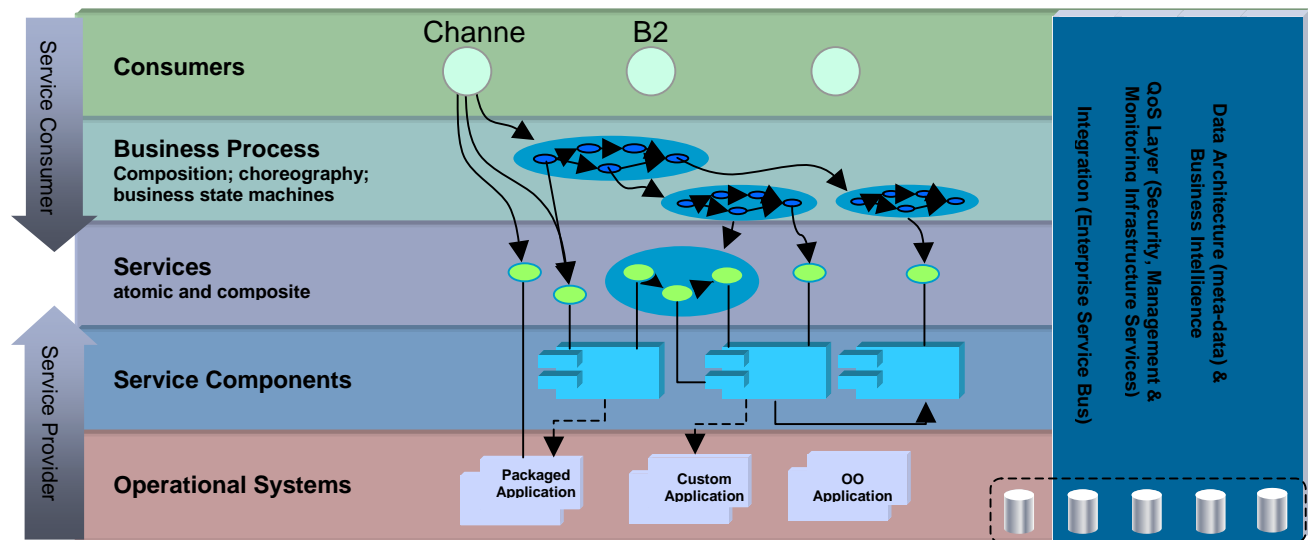
Conformance describes how well a given implementation matches or does not match a standard or a reference. You can have a conformance testing suite that returns results that certain aspects of an implementation match a reference implementation, and certain other aspects do not. A Web services implementation can conform with the WS-I basic interoperability profile. A lack of conformance does not necessarily imply a value judgment the way that lack of compliance with a law or regulation does. An implementation can comply with best practices in some areas and fall short, or differ (possibly for good reasons!) in other areas.

From a strict definitions perspective, it could be argued that this paper should be about conformance since the IT industry has not established a set of IT rules/regulations to comply against. However, since IT goals should support business goals and SOA is not just an IT initiative, we felt this paper should follow the well established context of business compliance against industry regulations.

SOA AND END-TO-END LIFECYCLE

Another important aspect of a services model is taking into account how a business service needs to function from its initial conception through withdrawal or replacement of a service (the lifecycle of services). A holistic, a top down view of how your business operates is important to realizing your business goals via SOA.

Let's take a design example. A new business function needs to be developed and introduced into your services environment. It's important for the designers to design the function with services management in mind. The 'design for management' principle provides the designer with a follow on perspective on what the development issue would be, the deployment issues, support, etc. In the recent past designers/developers could potentially create very complex applications that were impossible or at the least very costly to manage. This was done because there was no cross functional accountability; it was simply not the problem of the developers to worry about how difficult an application was to support (unless of course they were the application support organization as well). Today it's more important to have the design, development, deployment and maintenance organizations work together to ensure all aspects of the lifecycle are taken into account when creating new functions or services.



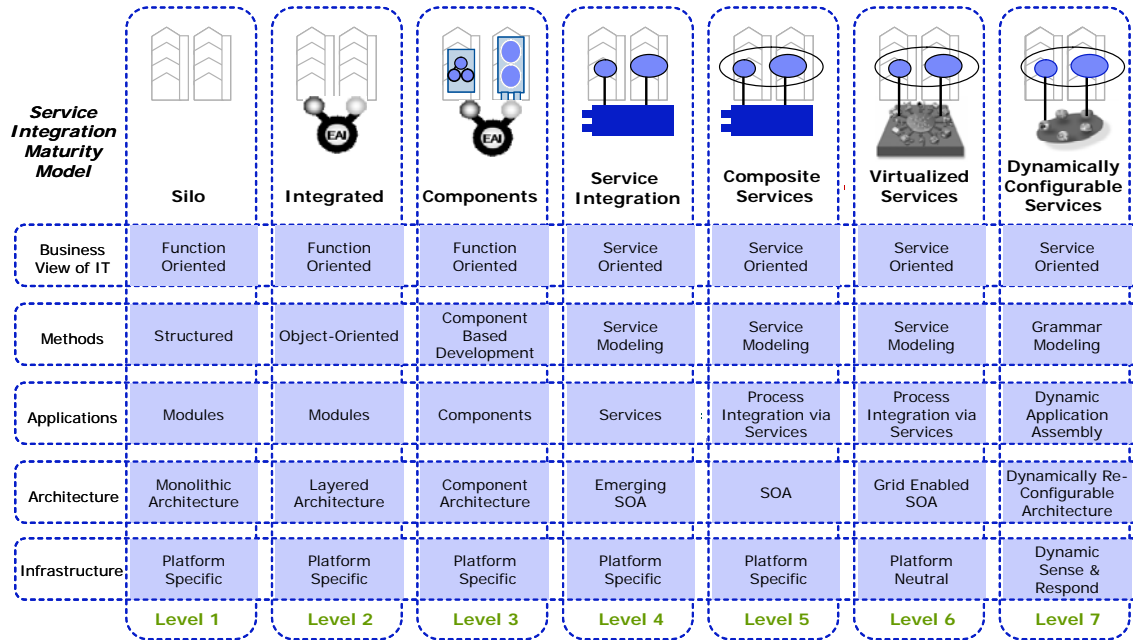
SOA Solution Abstraction Layering

LEVELS OF COMPLIANCE

A business might require nothing more than an ESB to meet their objectives. In that case, they could claim to be SOA compliant and have not implemented any Web services. So we believe a more useful approach is to define *3 levels of compliance* that provide a roadmap to achieving better returns in SOA based investments. With that said, keep in mind that you have the option to stop at any point along the way once you have achieved your goals (you don't have to get to advanced, business transformation compliance to 'claim' compliance; it's what works for you). One could argue that the higher level of compliance a business obtains, the more business value they potentially derive.

It makes sense to gauge compliance based on maturity using either SIMM (the Services Integration Maturity Model from IBM Global Services) or the IBM Software Group integration view as these methodologies define a roadmap for SOA maturity attainment.

BCS (IBM's Business Consulting Services organization) is using SIMM in consulting engagements to define as-is and to-be state for SOA Transformation.



The IBM Global Services Service Integration Maturity Model

This is based on the SOA Assessment from IBM Software Group and BCS on the [IBM Service Oriented Architecture \(SOA\)](#) website.

For the purposes of this paper, we will use the following categories to identify the potential levels of compliance:

- **Basic**
- **Intermediate**
- **Advanced**

That does not mean these categories are rigid and inflexible. In fact it is expected that they will grow and evolve as more acceptance of standards and best practices takes hold.

SCOPING FOR SOA

Working within the definition that SOA compliance is an environment where loosely coupled, IT agnostic services are consumed in a standards based interoperability model that is easily supported and managed, IBM's strategy is to provide SOA capability based on the SOA Foundation. IBM uses a simple four level lifecycle approach in its SOA Foundation Strategy:

- **Model**
- **Assemble**
- **Deploy**
- **Manage**

So let's take a look at potential compliance guideline criteria for the SOA lifecycle.

Model

Modeling is the process of capturing your business design from an understanding of business requirements and objectives and translating that into a specification of business processes, goals and assumptions – creating an encoded *model* of your business. As we indicated above, many businesses do not go through a formal modeling exercise. Those that do business modeling often capture their design using primitive techniques, such as drawing out Visio® diagrams or writing out the prose in a text document.

However, capturing your business design using a rigorous approach offers the potential to gain better insight into your business, by, for example using tools to reason about the design and its supporting rational. In particular, we can use the model to simulate how your business processes will actually run. A sophisticated modeling approach lets you perform “what-if” scenarios that reflect your understanding of the actual number of process instances, contacts, quantities, incoming traffic, etc. that you may experience in your business. The process can then be simulated using those parameters to predict the effect that process will have on your business and on your IT systems. If you don’t achieve the hoped-for results then you can change your process definition to try to improve your results. You can go on to refine your processes as you have modeled them to optimize your business performance even before ever investing in an implementation of those processes. Deep insights of this type are hard to achieve using primitive business modeling tools.

Your model will also capture key performance indicators – business metrics that are important measurements of your business. This could include, for example, a measure of the new accounts that you have opened in a given month. These key performance indicators are input to the assembly of your application and later, when the application is in production, collected and reported back to you. You will be able to use that information to determine how well your business is performing. You can use the correlation between your business design in your actual implementation in the information system to determine whether bottlenecks in your performance are due to limitations in your business design or limitations in the information system that automates your design.

[\(Source: SOA Foundation Whitepaper\)](#)

Potential Compliance Criteria:

Basic:

- Use of open standards based business modeling tooling.
- Identification of relevant business data.
- Categorization of relevant business data.
- Representing your business processes in a structured fashion (by using WebSphere Business Modeler as an example).
- Management focus towards Integration between business, application and infrastructure modeling.
- The use of WS-I Basic Profile constructs.
- Avoidance of sharing databases between applications.

Intermediate:

® Visio is a registered trademark of Microsoft Corporation

- Mapping of business information to information schemas in the lifecycle model.
- Integration between business modeling and application modeling: This can be accomplished by using a tooling capability to allow for the export and import of modeling formats (BPEL to UML, as an example by using IBM's Websphere Business Modeler and Rational Software Architect).
- Enabling the processing and management of data in a 'request/response' model (in other words, establish data sourcing/management in a web services type model where the underpinning implementation of say the database schema is not important for the requesting service). This could be viewed as enabling an "Information as a Service" model.

Advanced:

- End to end business data identification, normalization and integration.
- Establishment of business processes in a codifiable format (e.g. BPEL).
- The execution of business processes (perhaps BPEL based) in an automated fashion.
- End to end lifecycle governance so that design considerations are included in the lifecycle management process.

Assemble

You can use your business design to communicate with the IT organization – to *assemble* the information system artifacts that will implement the business design. The enterprise architect working with the business analyst can begin to convert the business design into a set of business process definitions and activities deriving the required services from the activity definitions. They can work with the software architect to flesh out the design of the services.

During the process of resolving a design and implementation of your modeled business processes and services, you should search your existing asset inventories – your legacy programs – to find application components that already meet your needs. Some application components will fit perfectly; some will have to be re-factored; and some will have to be augmented to meet the requirements of the design. These existing assets should be rendered as services for assembly into composite applications.

It is also possible that some of the your legacy is so heavily bound into a tight relationship with presentation and data logic and other business functions that you simply can not extract any re-usable componentry from those programs. In these cases you will have to decide whether and how to re-write these functions as new services, and how to migrate the processes that depend on those old programs.

Any new services required by the business design will have to be created. Software developers should use the SOA programming model to create these new services.

Final assembly includes applying the set of policies and conditions to control how your applications operate in your production environment. This might include, for example, business and government regulations, but can also include critical operational characteristics such as packaging, localization constraints, resource dependency, integrity control, and access protection.

[\(Source: SOA Foundation Whitepaper\)](#)

Potential Compliance Criteria:

Basic:

- Isolation of business applications as coarse grained ‘services’ exposed via a service interface inside the enterprise (such as a web services API).
- Use of a messaging bus.
- The use of SOAP and XML within your IT Infrastructure.
- The use of WS-I Basic Profile constructs, if using Web services.

Intermediate:

- Integration of business process and information management technologies
- Exposure of services externally outside your enterprise (using security constructs such as WS-Security in a Web services model)
- Decomposition and recomposition of existing applications into coarse grained services
- Reuse of existing services
- Adherence to a larger collection of the more basic industry standards (as an example for Web services: WS-I Basic Profile, WS-Security, etc.)
- Use of WDSL
- Basic use of mediation for services
- Use of more advanced Community Centric profiles (such as RAMP, Reliable Asynchronous Messaging Profile)

Advanced:

- End to end lifecycle compliance criteria, management.
- Following a composite application model where new business functions are generated by the underlying assembly of existing services.
- The ability to model, design, deploy and manage business functions through an end-to-end governance capability. (One example of this might be the use of an ESB as a service mediation engine).
- The inclusion of a services registry where services can be registered and located (via WSDL in a Web services example).
- Full integration of business information and business process accessible thru the services registry (for developers).
- The use of WS-Federation and brokered trust to enable Federated Identity interoperability.

Deploy

The *deploy* phase of the lifecycle includes a combination of creating the hosting environment for your applications and the actual deployment of those applications. This includes resolving the application’s resource dependencies, operational conditions, capacity requirements, and integrity and access constraints.

A number of concerns are relevant to construction of the hosting environment – including the presence of the already existing hosting infrastructure supporting legacy applications and pre-existing services. Beyond that, you need to consider appropriate platform offerings for hosting your user interaction logic, business process flows, business-services, access services, and information logic.

You need to consider the techniques you will employ for ensuring availability, reliability, integrity, efficiency, and service ability.

[\(Source: SOA Foundation Whitepaper\)](#)

Potential Compliance Criteria:

Basic:

- Manual deployment capability (using ANT scripts as an example).
- Implementation of deployment patterns.
- Integrated development/deployment procedures.

Intermediate:

- Some level of automated provisioning and virtualization within the IT infrastructure.
- Pre-integration of new services via automated exploitation and tooling – An example of this might be to build patterns or templates for designers/developers to use in the initial stages of service development so management requirements for those services are automatically captured and enabled at deploy time. In essence, the service could automatically notify the management subsystem of its existence and inform the management engine about what its service management requirements are.
- Auditability of the aforementioned deployment ‘pre-integration’ criteria.
- Web services management capability (monitoring, mediation, performance, SLO/SLA attainment).

Advanced:

- Use of ITSM (IT Services Management) based services to model and manage your entire IT management process.
- Pre-integration with existing Services that have already been deployed and made available in the environment.

Manage

Turning now to the *manage* phase of the lifecycle, you need to consider how to maintain the operational environment and the policies expressed in the assembly of the SOA applications deployed to that environment. This includes monitoring performance of service requests and timeliness of service responses; maintaining problem logs to detect failures in various system components; detecting and localizing those failures; routing work around them; recovering work affected by those failures; correcting problems; and restoring the operational state of the system. The manage phase also includes managing the business model – tuning the operational environment to meet the business objectives expressed in the business design, and measuring success or failure to meet those objectives. SOA is distinguished from other styles of enterprise architecture by its correlation between the business design and the software that implements that design, and its use of policy to express the operational requirements of the business services and processes that codify the business design. The manage phase of the lifecycle is directly responsible for ensuring those policies are being enforced, and for relating issues with that enforcement back to the business design. Managing the system also involves performing routine maintenance, administering and securing applications, resources and users, and predicting future capacity growth to ensure that resources are available when the demands of the business call for it.

(Source: SOA Foundation Whitepaper)

Potential Compliance Criteria:

Basic:

- Some level of diagnostic capability.
- Auditability of deployment processes/criteria.

Intermediate:

- The ability to ‘design for management’ as part of your development cycle and integrate with existing management services.
- Auditability of deployment ‘pre-integration’ criteria.

Advanced:

- Establishment and enforcement of business and management policies in real time.
- Ability to alter/change policies real time.
- Automated changes to management policies based on automatic monitoring of IT environment.
- Dynamic modification (e.g. provisioning, deprovisioning) of your runtime environment based on monitoring.

IN CONCLUSION

It’s apparent that there is no canned answer for SOA compliance today. The important perspective to keep in mind is the importance of a converged business and IT governance model that aligns to achieve business goals. Once the goals/objectives are set in place, then the compliance criteria are relatively easy to define and manage. SOA compliance will then evolve into a more well-defined set of business and IT criteria that enable business functions to be seamlessly designed, developed, deployed and managed within an enterprise’s established service lifecycle model.

We intend to mature this paper with further detail as the governance models mature.

REFERENCES:

IBM’s SOA Foundation Website, <http://www-306.ibm.com/software/solutions/soa/>
”IBM’s SOA Foundation: An Architectural Introduction and Overview”,
<http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-soa-whitepaper.pdf>
“Introducing the RAMP Profile”, Chris Ferris, <http://www-128.ibm.com/developerworks/webservices/library/ws-ramppaper.html>