

SOA Blueprints V0.2 Expert Review

Expert Review Board comment for version 0.2 of the SOA Blueprints Specification

1 DAVE MURRELL, XEDE

I have reviewed the specification - overall, a great job. Honestly, I think it is one of the most readable I have seen. I only have a couple points to consider:

- Under Section 3 (Concepts), you comment about service discovery, but explicitly exclude it as out of scope. I think you should consider including it for completeness, or at least frame a little more discussion around it.

- Under Section 5.3 ((HR Publishing to Security System), I think you may want to consider revising the design to specify a single topic instead of two. With this design, I think you may run the risk of the asynchronous topic messages being processed non-serial when there is that dependency. That is, if the add employee (changeEmployee) message does not get processed before the role selection (rolesChange), the role assignment will fail. As they are two distinct topics and there is no restriction on where or how they must reside (e.g., on separate message broker servers), thus there is no guarantee of the required serial behavior.

- Under Section 5.4 (The Legacy Adaptor), you may want to add more explanation of the getPossibleRoles() callback. It is not clear why this is needed until later in the specification.

That's all comments I have. Overall, I think it is a great example for SOA. Please let me know if I can do anything more.

2 BOLA ROTIBI, OVUM

Apologies for the delay but have unfortunately been unavailable for the past few weeks due to a client engagement. Anyway, I have now managed to do a quick review of your SOA specification document and have the following comments.

On the whole it was very comprehensive with clear instructions and guidelines. I think for an organisation wishing to understand how to implement SOA principles in their applications this specification does the job.

I have a query around section 3.3 i.e. the SOA patterns. Perhaps I am being pedantic when I suggest that you put in a few examples in the Serial and Parallel Service Orchestration. There are examples in all the other sections of that enable the user to quickly understand what is being depicted. I would also like to ask whether you believe that a data service can also be a business service, or is the latter definition only for composite services. An example I think of is whether a data service which for instance retrieves flight times be considered a business service or would it only be a business service if its granularity was raised i.e. a flight booking services, which I guess would contain a number of data services. Perhaps this will be clear to most architects and senior developers but it was one of the few things that caused me to pause and think.

Is section 3.3.9, Interception and Extensibility a reference to Aspect Oriented Programming?

I particularly like how you make reference to which pattern is being used in your use cases.

Whilst this is a technical specification, it does not make mention on the importance of the development environment or direct references to the importance of having to employ a process that employs good software engineering principles (which actually underpins and makes SOA possible).

By the heavy use of models in the specification, I do not suppose you would expect anyone not to use a modelling tool to help build their business processes or model their services.

I would like to go through the document again I do realise that you will already have compiled your list, however, if anything else jumps out I will pass it on for the next revision.

3 SCOTT METZGER, TRUELINK

Overall a great document, nice work! I'm looking forward to seeing the next draft. Most of my comments are minor points and I expect you may have thought of many of these items but here are the things that came to mind...

Section 2.5

Although you don't want to cover performance in this version of the doc you may consider setting the seed of the impact/benefit of asynchronous vs. synchronous interfaces with respect to performance. Another note to consider to point out, as obvious as it may be, is workflow it the heaviest weight (in figure 2).

3.3.1.1

Asynchronous Services states that these services do not return any response to the invoker. Most SOAs have a notion of guaranteed message delivery (i.e. Message Broker, Web Services) and initially respond to the invoker with an acknowledgement that the message has been received and in some cases validated for integrity via XSD ect.

3.3.2 & 3.3.3

A best practice measure we have is these services is that it does not rely on another peer component service to function.

3.3.3.2

A best practice for parallel service orchestration is there is still only one end-point.

3.3.8

You might consider some more definition around compensating transactions because this is a critical element to consider in SOA. In addition to rollback to an error or timeout there are compensating business transactions. For example a payment workflow service is in process where a credit card has been authorized but not captured. The person calls customer support and a separate workflow in CRM application needs to "compensate" the payment work flow that is still in flight is there is cancellation or price reduction. Compensating transactions can get quite in depth. So much so you might consider adding a separate heading under SOA patterns.

This would also be a good spot to talk about the potential for a timeout/watchdog exception mechanism for workflow services.

3.3.11 Security

You may want to contrast authentication, authorization, signing, and encryption.

3.3.11.1

HTTP authentication. Consider changing user/password to credentials since certificate-based authentication applies to this also.

3.3.11.2

HTTPS Encryption. What actually happens the public/private (asymmetric) keys are used to exchange symmetric key which used for the actual encryption/decryption of the data. This symmetric key can be changed during the session.

3.3.11.3

XML signing is also used to validate the identity of the sender. I would also add XML encryption and another bullet to draw out the topic of transport vs. message/payload encryption.

5.3

I should point out the UML notation difference to synchronous and asynchronous interactions since this is a subtle point in UML.

5.3.1.2 & 5.3.2.2

In the patterns covered it looks like 3.3.11.3 also applies. I'd also consider encrypting at the transport layer or data payload layer given the nature of the data.

5.4.1

In the patterns covered it looks like 3.3.11.2 and 3.3.11.3 also apply. I'd I think you want to specify https for security: This applies to most of the following examples as well.

5.4.3

This (figure 8) would be a good example to call out the need for a timeout exception.

5.5.1.7

Figure 13&14. I try to avoid the pattern of multiple exit/finish nodes. I prefer to update a data member on the status and have a single exit point. In my experience debugging workflow code is even more difficult with multiple exit points.

General comments:

One of the values of WSDL is using the underlying XSDs for data validation. This capability is unfortunately overlooked by many Web Services developers.

When there is more than one "CRUD" application running you need consider concurrency (locking, dirty reads/writes, caching etc.) in an SOA.

Summarize the characteristics that would guide the decision of what type (workflow, business, component) service is appropriate.

Introduce the pattern for synchronous to asynchronous adapters via a polling mechanism ect. Since this scenario can come up frequently when integrating with legacy systems.

It would be helpful for this target audience to provide an overview of items to consider in your SDLC for SOA development. This could include: decomposing requirements into SOA primitives validation, programming by contract (WSDLs)

4 DORON SHERMAN, INDIVIDUAL CONTRIBUTOR

General comments:

- Address the challenge of proper partitioning of functionality (especially existing apps) into services.
- More motivational support for investing in SOA as compared to using traditional approaches can be useful.
- The level of detail provided in describing the apps and schemas requires a lot of scrutiny, which frankly, I did not have a chance to go into with sufficient depth.

1.- The assertion of an Application Platform Suite consisting of portal and integration as add-ons to an App Server (still lacking definition of the Application Platform Suite) and making the implicit statement that the App Server is the core foundation for SOA... don't seem supported at this stage.

- Is it intended for the set of interconnected apps to make up an enterprise? How do these apps relate to existing apps, such as ERP implementation?

2.2 - the list is somewhat of a laundry list, best if items are grouped under categories, e.g. separate service types from technical behaviors.

- conversational workflow services? I believe that a business process is central to discussion of SOA, not just as a pattern but more so as a fundamental concept that justifies the investment in designing application infrastructure based on SOA.

- composite web services need not get broken down into serial and parallel... these are just two types of composition activities.

- need to list "event handling" as behavior of services

- not sure why Message Broker is in the diagram as a transport, I'd expect protocols only there.

- In general, I prefer the view conveyed by Figure 6 in <http://www.bijonline.com/PDF/Oct03Lublinsky.pdf> to the Figure 2 here.

2.3

- Use of services (by definition loosely-coupled) where tight coupling is called for

3.2

- add: support for multiple forms of human interaction (e.g. portal, email, wireless, etc)

3.3.1.2

- mention having to deal with message correlation for callbacks (e.g. WS-Addressing)

3.3.3

- it's not clear whether composite services are not multi-step (aka conversational) like workflow services are, or are not long-lived. Introducing transaction here can be confusing since it's opening a whole set of other issues. Are there no async invocations involved with composite services?

3.3.4

- comment about the type methods available for carrying out this implementation (e.g. BPEL).

3.3.7

- if running sync or async? Notification midway are normally only possible with async invocations.

3.3.11.3.

- I like the word "tampering" in that context

5.1

- It's implied that using a MessageBroker is superior to using Web services. Is that assertion intended? Use MessageBroker where possible rather than Web services?