




Getting Started, SOA Lifecycles and Wrapup

Don Bagwell
IBM Washington Systems Center
dbagwell@us.ibm.com

© 2007 IBM Corporation



This slide intentionally left blank

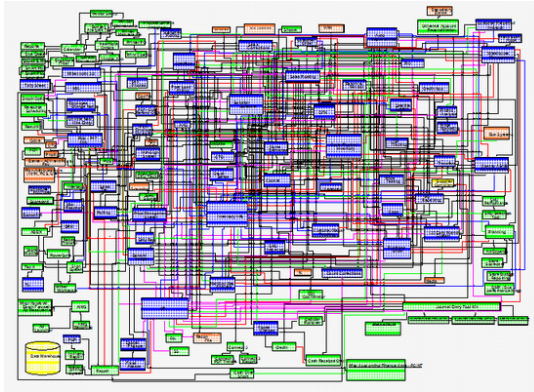
2

IBM Americas Advanced Technical Support
Washington Systems Center, Gaithersburg, MD

© 2007 IBM Corporation

Going Back to Our Original “Problem Statement”

Do you recall the really inflexible application interconnection chart we showed earlier? That serves as the backdrop for this discussion:



How do you approach this?

How do you make sure it's approached in a sensible, controlled fashion?

How do you make sure it works as you hoped?

This is more than just a “use product X” issue. Here we get into the issue of planning for and executing on a plan in a defined, controlled and disciplined manner.

Starting out ...

3

IBM Americas Advanced Technical Support
Washington Systems Center, Gaithersburg, MD

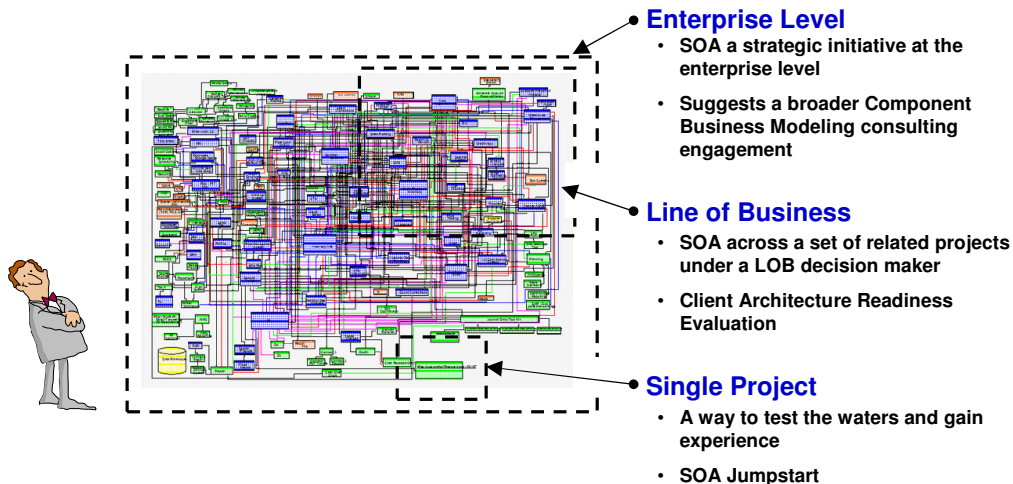
© 2007 IBM Corporation

Way back in the first unit we presented picture that represented the typical “spaghetti code” application design. We mentioned back then that many companies -- IBM included, by the way -- have pictures that look like that. It just something that happened over time. As we contemplate what we've learned so far, and look at this picture, we're faced with some basic questions about approaching this and applying SOA to it. The purpose of this unit is to give you a sense of the issues and thinking in this space.

Before we do that, it should be clear that what we're about to talk about is not just a product usage issue. What we're about to talk about is really more of a high-level control and discipline approach, aided by technology implemented in particular products. Much of this is really related to human behavior -- tools are only as good as our willingness to use them properly.

Starting Out

There are three *basic* ways you can approach this question ...



No one is “best” -- it all depends on the needs of the business and the state of SOA awareness, understanding and acceptance in the organization

IBM SOA lifecycles ...

4

IBM Americas Advanced Technical Support
Washington Systems Center, Gaithersburg, MD

© 2007 IBM Corporation

There are three *basic* ways you can approach the topic of starting out with SOA and implementing it. There are, of course, variations on each of these ... as well as the opportunity to blend these. But these three will serve the purpose of illustrating the basic approaches.

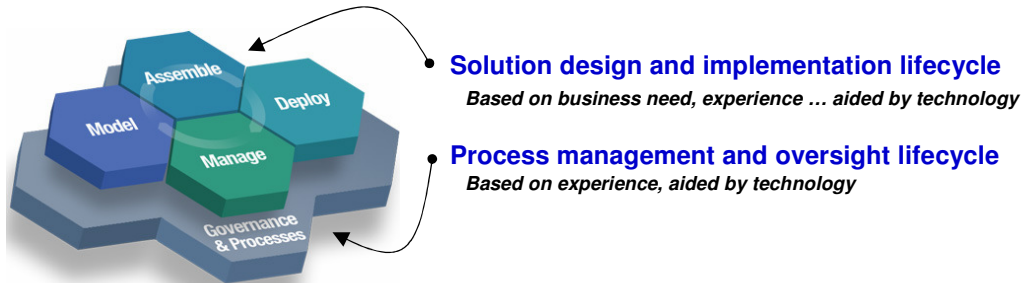
- Enterprise Level** -- here you approach the topic from the top down. This is a kind of “all in” strategy, where the start of the analysis is done at the highest level, and the business is analyzed not from a technology point of view (initially), but from a business component point of view. Clearly this would require buy-in from the highest level. IBM Business Consulting Services has offerings to assist enterprises in assessing their business and determining how, where and why SOA would be beneficial.
- Line of Business** -- a smaller version of the Enterprise Level, here the focus is fenced within a particular line of business (division, organization unit), or possibly related to a particular set of projects within an enterprise (for example, the shipping function). IBM services provides an architectural readiness evaluation to help with this process.
- Single Project** -- when the desire is to start small and build experience from there. This makes good sense in a lot of cases. Here a particular potential service is selected and the solution is built around that. Experience is gained in a small, controlled (and often less risky) way.

The ubiquitous “it depends” answer is given for which of these is best. A lot depends on the business as well as the current state of mind of the business decision makers.

Let's now see how IBM describes the “lifecycles” of SOA adoption.

The IBM Icon

You'll see this icon used frequently in IBM presentations on SOA. What it says is that SOA can be viewed as a set of lifecycle processes.



What this is suggesting is that SOA involves same key things that *any* project involves:

- Proper planning and consideration of the relevant issues
- Actual development of solution and implementation of the solution
- On going management and monitoring, along with updates and improvements
- An understanding of who the decision makers are, and where lines of responsibility fall between different organizations
- Commitment to the effort, support of key sponsors, discipline

SOA is a new concept, but the need for careful planning and discipline is not new or unique to SOA

SOA lifecycle ...

When you scan the IBM websites on SOA, you'll see the icon shown in the chart above over and over again. That icon shows interconnected hexagons, and often two layers of hexagons. The essential message of this is that SOA is not a single event. It is an ongoing process that involves a lifecycle. And in fact two lifecycles are represented there: the one on top related to the design and implementation of the SOA solution; the one underneath related to effective control and discipline over the one on top. Both of these lifecycles are more than pure technology plays ... both are based on experience and the needs of the business, but both can (and are) aided by applied technology.

The concept behind a lifecycle is that the adoption and implementation of SOA is something that occurs over time, involves a set of known and relatively common tasks, and is benefited from a thoughtful, controlled and disciplined approach. There's nothing new in that; pretty much every I/T project from day one had that in one form or another. The message here is that SOA, while very definitely a new approach to business, is not so different that it too can't benefit from a disciplined lifecycle approach.

Let's look at the two lifecycles indicated here ... first the SOA lifecycle, then the governance lifecycle.

The SOA Lifecycle

The four hexagons in the top of the icon do have meaning:
<http://www.ibm.com/software/solutions/soa/gov/>

The diagram consists of four 3D hexagons arranged in a diamond shape, each representing a phase of the SOA lifecycle. The top-left hexagon is blue and labeled 'Model'. The top-right hexagon is teal and labeled 'Assemble'. The bottom-left hexagon is green and labeled 'Manage'. The bottom-right hexagon is teal and labeled 'Deploy'.

Model

- Determine what might make a good re-usable service
- Map out business processes and determine where these services fit within that process

Assemble

- Construct the re-usable services
- Build composite applications
- Construct flows and processes

Manage

- Monitor the environment
- Correct problems
- Collect data on effectiveness
- Perform needs analysis for improvements

Deploy

- Implement the runtime environments
- Deploy the assembled services and composite applications

This ought to appear relatively obvious to you ... that's because the basic concept is the same we've had for many years.

It's useful to remind ourselves of this lifecycle. It's not always followed.

IBM Offerings ...

6 IBM Americas Advanced Technical Support
Washington Systems Center, Gaithersburg, MD © 2007 IBM Corporation

The information on this page comes from the website listed at the top. The SOA lifecycle is represented by the four hexagons labeled, "Model, Assemble, Deploy and Manage." What these relate to are:

- **Model** -- in this phase we determine what our re-usable service will be; what its interface is and what existing I/T assets are involved. From there we analyze where this service will fit within a business flow. For instance, if the service is "update inventory" this service might map to the after the receiving function has acknowledged receipt.
- **Assemble** -- here we actually build the service. If it's a computer-based service, we use the various tools (Rational Application Developer, WebSphere Developer for zSeries, etc.) to construct the code that implements the service.
- **Deploy** -- this is related to all the things needed to ready the runtime environment and to deploy the new service code into the environment. For instance, if it's a CICS web service then we'd do the things we did in the CICS lab.
- **Manage** -- the standard post-installation management tasks ... monitoring the environment, correcting problems, measuring against goals and looking for ways to improve.

The concept here is nothing new. The SOA mechanisms behind this are new, and many of the tools designed to assist in this are new, but the *concepts* are not new. It's useful to remind ourselves of this because treating SOA adoption as a lifecycle process, with a focus on proper planning, discipline, execution and follow-through will mean the success of SOA will be greatly enhanced.

The next chart is intended to give you a sense for the IBM offerings in this space.

Some IBM Offerings for the SOA Lifecycle

This gives a flavor ... but other products may fit as well.

<http://www.ibm.com/software/solutions/soa/offerings.html>

IBM SOA Foundation - Model Phase:

- IBM® WebSphere® Business Modeler
- IBM Rational® Software Architect

IBM SOA Foundation - Assemble Phase:

- IBM WebSphere Portlet Factory
- IBM WebSphere Integration Developer ★
- IBM Rational Application Developer ★
- WebSphere Developer for zSeries ★
- WebSphere MQ Broker Toolkit ★

IBM SOA Foundation - Deploy Phase:

- IBM WebSphere DataPower SOA Appliances
- IBM WebSphere Process Server ★
- IBM WebSphere ESB ★
- IBM WebSphere Message Broker ★
- IBM WebSphere Partner Gateway
- IBM WebSphere Adapters
- IBM WebSphere Portal
- IBM WebSphere Everyplace® Deployment
- IBM Workplace Collaboration Services
- IBM WebSphere Information Integrator
- IBM WebSphere Application Server ★
- IBM WebSphere Extended Deployment

IBM SOA Foundation - Manage Phase:

- IBM WebSphere Business Monitor
- IBM Tivoli® Composite Application Manager for SOA
- IBM Tivoli Composite Application Manager for WebSphere
- IBM Tivoli Identity Manager
- IBM Tivoli Access Manager for e-business
- IBM Tivoli Federated Identity Manager

Message: Many tools to use during each of the phases of the SOA lifecycle

★ Talked about or used in lab

Challenges to implementation ...

7

IBM Americas Advanced Technical Support
Washington Systems Center, Gaithersburg, MD

© 2007 IBM Corporation

There are a lot of IBM tools designed to assist in this lifecycle. The chart above comes from the website referenced at the top. The little red stars highlight tools or products we've touched on or used in this workshop.

Caution: this chart is not exhaustive. For instance, the website did not list WD4Z or the Broker Toolkit, both of which definitely fit into this space. The point of this chart is not to show you every possible tool, but rather give you a sense that many tools do exist and that a discussion with an IBM technical sales specialist who focuses on tooling would be in order.

Challenges to Implementing SOA

Successful implementation is not a given. Things may stand in the way:

Establishing Decision Rights

- Who within an organization holds authority to make SOA decisions
- How cross-organization boundary decisions are to be made

Defining High Value Business Services

- What constitutes a service that's worth considering
- Prioritization to be used when multiple service opportunities are competing with one another for resources

Managing the Lifecycle of Service Assets

- When new services get deployed
- Change management policies
- Service retirement policies

Measuring the Effectiveness of Services

- Definition of what constitutes "effectiveness"
- Need policies to define how data is collected and analyzed

What this suggests is that a methodology should be in place to resolve these issues prior to embarking on an ambitious SOA strategy. It gets to the question of SOA *governance*

Governance lifecycle ...

Having a focus on the SOA lifecycle and having tools in hand is all well and good. But it's not the complete story. There still lurks challenges to the process. And those challenges have to do more with human decisions than technical impediments. It is one thing to say "We're going to implement SOA" and another to have everyone acting as a united team in that effort.

This is where the subject of "governance" comes into play. Governance is really just a fancy term for a structured and disciplined process of planning, decision making and control. Without this, the SOA implementation has the very real risk of becoming anarchy. It would be like setting 100 people out to build a house without plans and without a job foreman. Something might get built, but it probably would not be a working house.

The SOA Governance Lifecycle

The focus here is much more on insuring the business gets the most out of its SOA investment. There's a *people* and *organizational* emphasis.

<http://www.ibm.com/software/solutions/soa/gov/>

Plan

- Understand overall scope of the governance needs
- See what governance policies already exist, what doesn't, and which need improvement

Define

- ... governance processes
- ... enforcement processes
- ... success factors and metrics
- ... process owners and who pays for what
- ... decision makers / roles of authority



Measure

- Monitor compliance with policies
- Monitor compliance with governance arrangements
- Monitor IT effectiveness metrics

Enable

- Put in place the policies, enforcement processes and governance infrastructure
- Education people on their roles
- Begin the process of governing SOA

Messages:

- SOA is like any other business process in that one should approach it intentionally, with commitment and an eye on how to make it successful
- Based on customer engagements, IBM has developed a set of proven “patterns” (best practices) and a governance consulting model to assist

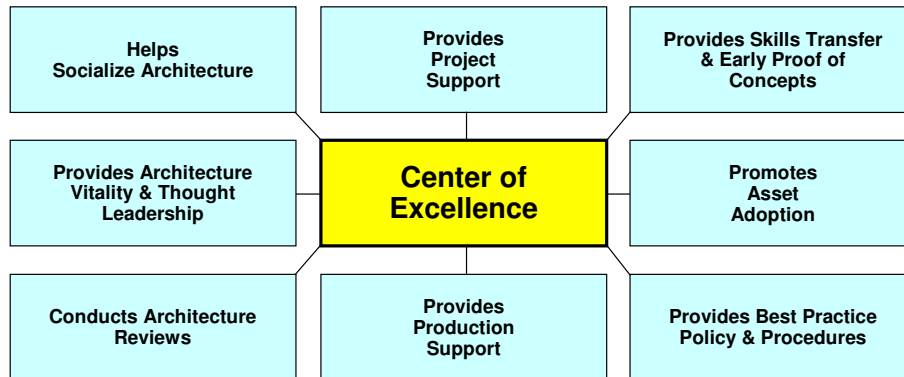
SOA Center of Excellence ...

So here we introduce the lower set of four hexagons. This represents the “Governance Lifecycle,” and it involves the process for planning and implementing the control over the SOA implementation. The chart effectively spells out what each of the four hexagons represents, and what kinds of things are under each. This governance lifecycle is based on experience gained by IBM helping customers with early SOA projects.

One of the things IBM has discovered is that the effective adoption and implementation of SOA is aided by the establishment of an “SOA Center of Excellence” within the organization. That’s next.

SOA “Center of Excellence” Concept

The idea here is to institute an organizational focal point for SOA adoption and guidance. This helps avoid a fragmented, silo approach.



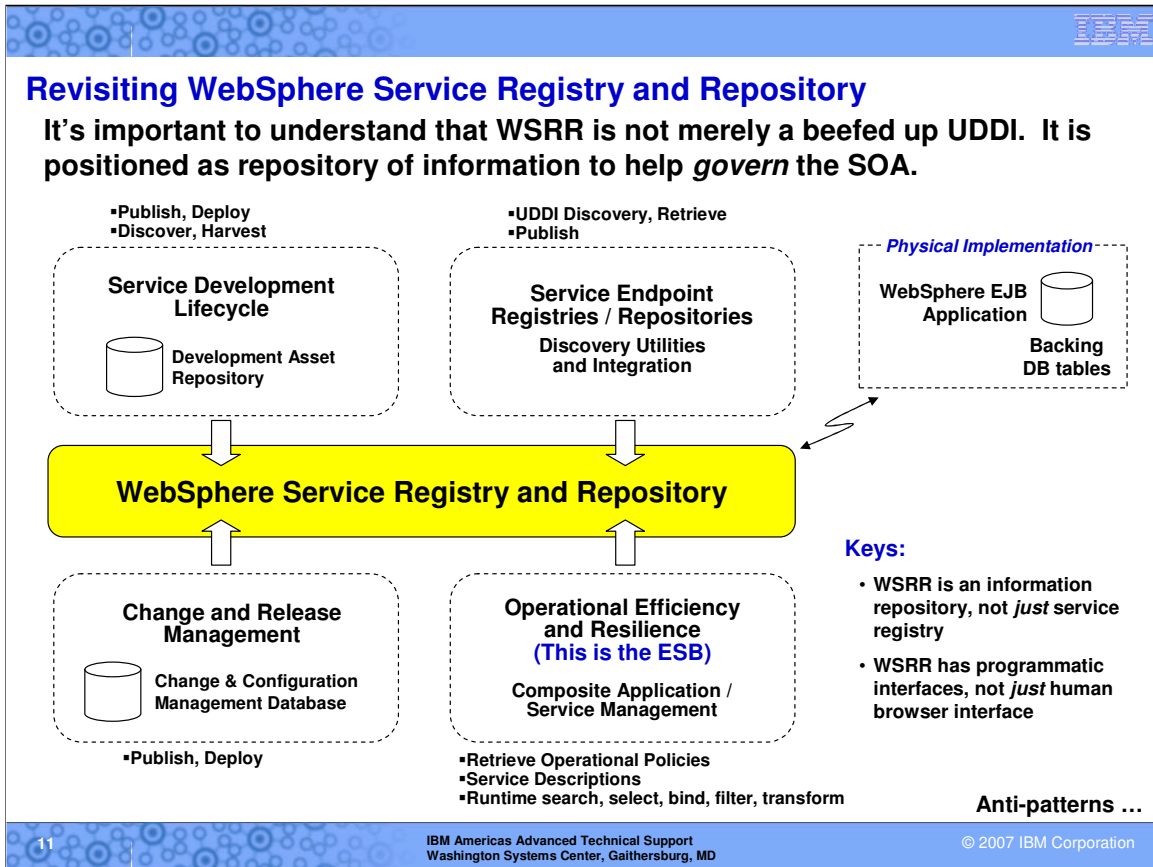
IBM consulting experience in this area has shown this to be an effective means of providing a consistence and centralized view of SOA adoption.

This of course requires buy-in from a decision maker somewhere up the chain. The broader the scope of SOA adoption, the higher this would need to be.

The role of the WSRR once again ...

The idea here is that optimally there is a set of people within an organization who are given the mission of overseeing the implementation of SOA. Think of this as a kind of “orchestra leader” who directs a consistent set of practices across the enterprise. There is, of course, no magic to this concept at all. It’s pure common sense.

But that doesn’t mean it’ll automatically happen. Establishment of such a center of excellence (or whatever you wish to call it) is going to require approval from a key decision maker; it’ll likely require funding, or at least sponsorship of its existence; and it’ll require that others within the organization look to it for guidance. It’s not a foregone conclusion that even if one is established it’ll be effective. But the point of this chart is still valid -- that IBM has found that adoption of SOA within an enterprise is best served by a centralized staff who sees “the big picture” and is maintaining the consistency across the enterprise.



We mentioned the WebSphere Service Registry and Repository (WSRR) once before. We're going to re-introduce it here ... because its role is more than just a place where service interface information is kept. It is that as well, but it's not *just* that.

Recall that what WSRR is -- physically -- is an EJB application that runs inside of WebSphere Application Server. It has a set of defined database tables backing it up. It is a storehouse for information about services. Because it runs in WebSphere it has a both human interface (a web browser and Eclipse plugin) but also a programmatic interface (SOAP and RMI). That programmatic interface allows other tools to interface with it and store/retrieve information about services.

And that's the key to its role in the governance lifecycle. Governance is all about applying control and discipline to the SOA environment, and that's best done when the information being used to guide that control is current and reliable. WSRR, acting as a repository for that information, will allow the governance tools to better advise the people on how best to govern the environment.

The four dashed-line boxes in the chart above show four major disciples of the overall lifecycle picture (both SOA lifecycle and governance lifecycle) interacting with the WSRR product to store and retrieve the information.

Note: the little database icons in the "Service Development Lifecycle" box and the "Change and Release Management" box are somewhat important to note ... WSRR is not intended to be a code repository, or a version control mechanism for software components. Lots of other tools do that today and do it very well. What WSRR does is maintain information about those service; not the actual code that implements the services.

SOA Adoption “Anti-Patterns”

Think of a “pattern” like a “best practice,” or a “good thing to do.” An *Anti-pattern* is the opposite ... something that hinders successful adoption

<http://www.ibm.com/developerworks/webservices/library/ws-antipatterns/>

Table 3. List of antipatterns

Category - ID	Antipattern	Description
Adoption - A1	Technology Bandwagon	A noted common trend where corporations decide to jump on the new technology bandwagon without careful consideration whether this technology have any contribution to the business.
Adoption - A2	So, What's New?	Lack of understanding of the differences between SOA and previous computing paradigms drives skeptical to claim that SOA is just a name for same old techniques. The result is opposition to the adoption of SOA even if such adoption benefits the business.
Adoption - A3	The Big Bang	This antipattern is also known as "Bite more than you can chew." It is observed when SOA is viewed as a panacea, leading to a push to change all the enterprise systems and architecture at once. Such a big bang adoption could result in failures that are then unjustly blamed on SOA.
Identification & Design - I1	Web service = SOA	When architects equate SOA with Web services they run the risk of replacing existing APIs with Web services without proper architecture. This will result in identifying many services that are not business aligned.
Identification & Design - I2	The Silo Approach	In this antipattern, services are identified based on isolated applications, thus same services are identified by different groups with different names. As a result, no common services or service sharing are realized.
Identification & Design - I3	Misbehaving Registries	Duplicate service registries and overlapping, unclear ownerships result in governance nightmare and runtime confusion, potential bad performance, and unplanned costs due to duplication.
Realization - R1	Chatty Services	This antipattern describes a common mistake developers usually make when they realize a service by implementing a number of Web Services where each communicates a tiny piece of data. This will result in the implementation of a large number of services leading to degradation in performance and costly development.
Realization - R2	Point-to-point Services	Using XML or SOAP over HTTP between applications replacing middleware by point to point web services results in the emergence of point-to-point integration as the de facto integration pattern.
Realization - R3	Component-less Services	By jumping into development and implementation of Web services without having a clear association with owning components results in unstructured and undisciplined architecture (non strict layering). This leads to inflexibility behind the service interface and in preserving the legacy limitations of existing applications and packages.


Based on
collective
experience of
IBM SOA
architects and
designers



It's valuable to review these because by pointing out the “bad thing to do” the better thing to do becomes more clear

Some detail ...

As part of IBM's ongoing work with customers in this area, IBM has developed a set of “patterns” -- best practices -- for SOA. The SOA lifecycle is a simplified representation of some very serious consulting and implementation “patterns” for SOA adoption and implementation. At the same time, IBM has seen lots of what **not** to do. Those IBM calls “anti-patterns.” It's an interesting topic to explore because what it does is highlight, by contrast, what *should* be done. The URL at the top of the page shows the URL where this topic is explored in much greater detail. We'll offer some insight over the next couple of charts.



Anti-Patterns, Part 1

Technology Bandwagon

Where corporations decide to jump on the new technology bandwagon *without careful consideration whether this technology have any contribution to the business.*

At the end of the day, it's a business decision. Never adopt technology for the sake of technology.

What's New?

Lack of understanding of the differences between SOA and previous computing paradigms drives skeptical to claim that SOA is just a name for same old techniques. The result is opposition to the adoption of SOA even if such adoption benefits the business.

We've claimed there are some *concepts* in common with efforts in the past, particularly as they relate to requiring structure and discipline. SOA shares a common *objectives* with efforts of the past (re-usable assets). But SOA is different due to enabling technologies not available in past.

The Big Bang

Occurs when SOA is viewed as a panacea, leading to a push to change all the enterprise systems and architecture at once. Such a big bang adoption could result in failures that are then unjustly blamed on SOA.

One of the reasons why starting with a small, controlled approach is sometimes best. Also why having a good handle on the business value is good, and why having thought about the governance issue is important.

More ...

13

IBM Americas Advanced Technical Support
 Washington Systems Center, Gaithersburg, MD

© 2007 IBM Corporation

The first three anti-patterns. The chart says what needs to be said ... about the anti-pattern and what we've talked about in this workshop.

Anti-Patterns, Part 2

Web Service = SOA

When architects equate SOA with Web services they run the risk of replacing existing APIs with Web services without proper architecture. This will result in identifying many services that are not business aligned.

Web services may well be a starting point towards SOA, but always remember that SOA is more than web services, and more than just technology.

Silo Approach

Services are identified based on isolated applications, thus the same services are identified by different groups with different names. As a result, no common services or service sharing are realized.

This is where proper up-front thought to governance comes into play. Having answers to questions of scope and authority and who's looking at the "big picture" -- or at least a sense of understanding -- is helpful to successful SOA adoption.

Misbehaving Registries

Duplicate service registries and overlapping, unclear ownerships result in governance nightmare and runtime confusion, potential bad performance, and unplanned costs due to duplication.

Again, we come back to the question of discipline and control. Not a new concept, and SOA isn't unique in benefiting from it. However, it's also true that SOA isn't immune from needing it (see "The Big Bang").

More ...

The next set of anti-patterns.

Anti-Patterns, Part 3

Chatty Services

Where services are created that are too granular. Each returns only a small bit of data. To get usable unit of information, dozens or more services required. This results in the implementation of a large number of services leading to degradation in performance and costly development.

There's a bit of a tricky balance here between going too granular and making an exposed service so big that it's reusability is limited to only a few users. No magic formula here ... based on experience and careful thought.

Point-to-Point Services

Using XML or SOAP over HTTP between applications replacing middleware by point to point web services results in the emergence of point-to-point integration as the defacto integration pattern.

We pointed this out in our intro to ESB ... Web Services by itself is still a point-to-point model. There is a decoupling of the interface, but the integration between user and service is still point to point.

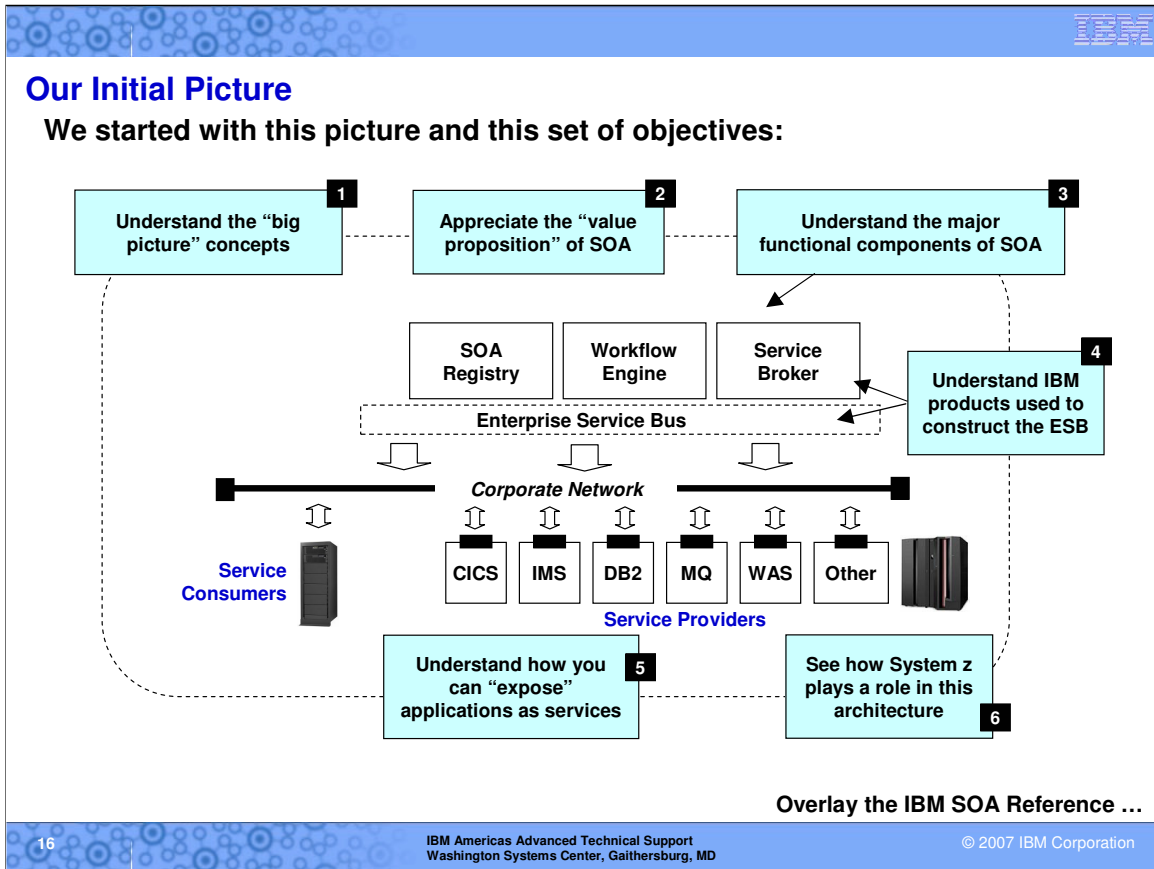
Component-less Services

Implementation of Web services without having a clear association with owning components results in unstructured and undisciplined architecture (non strict layering). This leads to inflexibility behind the service interface and in preserving the legacy limitations of existing applications and packages.

Gets back to the question of approaching services identification and architecting as a disciplined practice, rather than an ad hoc thing.

Cycle back to the initial picture ...

The final set of anti-patterns.



This is our initial picture from way back in the first unit. We used it to represent a more physical representation of the concept of service consumers and service providers using the corporate network to interact. We represented the ESB as middleware function mapped onto the corporate network and providing function to help connect consumer with provider. We covered in some detail two IBM products that implement the ESB -- WebSphere ESB and WebSphere MQ Broker. We talked about the workflow engine (WPS) and the SOA registry (WSRR).

The blue boxes represent some of our initial education objectives, and we hope and trust that by this point in the workshop you've got a sense for how each of those things works.

But this picture is intentionally simplified. The full SOA Reference Architecture picture shows IBM's bigger vision.

The Bigger IBM SOA Reference Architecture Picture

We've pointed out that our picture was intentionally simplified ... more a "physical" than "logical." Here's IBM SOA Reference Architecture:

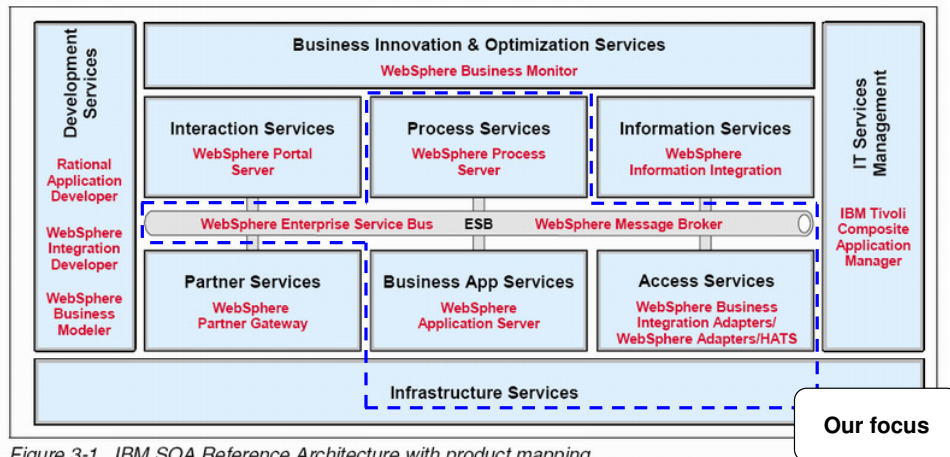


Figure 3-1 IBM SOA Reference Architecture with product mapping

This is a super-set of our starting picture. Our focus has been more limited, but in this last unit we've branched out and touched on other parts of this picture.

zIAW ...

We once again introduce IBM's SOA Reference Architecture to illustrate that what we showed in our initial picture was intentionally less than the total picture. Our focus throughout most of this workshop has been on the area enclosed by the thick dashed blue line. In this unit we've started to touch on topics that fall in other parts of this overall SOA architecture picture. And we've given hints along the way of IBM product mappings and where they relate to the various architectural blocks.

Note: product mapping is a challenge because there's a lot of products and because many of the products do many things. So they don't necessarily map perfectly onto the architectural chart. Take any product mapping picture as giving a sense rather than being the definition last answer on mapping.

All this may be more than you care to digest at this point. What can IBM do to help?

The zIAW Offering

zSeries Infrastructure Architecture Workshop

A multi-day interactive workshop in which IBM and customer interact to evaluate existing architecture and come to understanding of next steps

Who should attend ...

<p style="color: #4F81BD;">Customer</p> <ul style="list-style-type: none"> Enterprise Architects Business Analysts Information Technology Managers Line of Business Managers 	<p style="color: #4F81BD;">IBM</p> <ul style="list-style-type: none"> Business Integration Solution Specialist Business Integration Sales Specialist Software IT Architect or Subject Matter Experts Value Assessment Specialist (optional)
---	--

Deliverables ...

<p style="color: #4F81BD;">Workshop Results Summary</p> <ul style="list-style-type: none"> Executive Overview Requirements Solution Architecture Product Information Next Steps 	<p style="color: #4F81BD;">Strategic Outlook</p> <ul style="list-style-type: none"> Addresses out of scope topics Forward looking observations and recommendations <p style="color: #4F81BD;">5-year Total Cost of Ownership (optional)</p>
---	---

Contact ...

David McCorkle, IBM Solutions Architect
 1-816-556-6097
 dmmccor@us.ibm.com

Other SOA services from IBM ...

18
© 2007 IBM Corporation

IBM Americas Advanced Technical Support
Washington Systems Center, Gaithersburg, MD

One thing IBM can do to help is to conduct what's called a zIAW -- a zSeries Infrastructure Architecture Workshop. This is where IBM subject matter experts and customer key players get together in a room and work through the existing environment and develop an understanding about an SOA solution that would meet their business needs.

Other SOA Consulting Services

<http://www.ibm.com/software/solutions/soa/services.html>

SOA Consulting Services

Service oriented architecture (SOA) helps clients create business value by helping them develop a more on demand IT infrastructure. But the demands of an increasingly competitive global marketplace — driven largely by the networked world — are driving companies to find new ways to grow their businesses, while at the same time they must operate as efficiently as possible. Responding to competitive pressures and market opportunities requires business systems — and supporting technology infrastructures — to be quickly adaptable and flexible in structure. (If this sounds familiar, it should: IBM introduced the idea of On Demand Business.) To address these challenges, IBM provides professional SOA services in three major areas.

Business Services

The benefits of an SOA and Web services solution include increased flexibility, responsiveness, speed and efficiency. SOA and Web services have become part of the new corporate agenda; however, many clients do not know how to plan, design, implement or manage an SOA and Web services. IBM Service-Oriented Architecture Business Services has a series of offerings that leverage IBM expertise to help with every stage of the SOA process.


- IBM Design Services for Service-Oriented Architecture
- IBM Business Enablement Services for Service-Oriented Architecture
- IBM Implementation Services for Service-Oriented Architecture
- IBM Management Services for Service-Oriented Architecture

Bringing a sizeable storehouse of knowledge to the table

Finally ...

And still more service offerings.

One thing should be noted here. There's a temptation to think this is just a marketing pitch. But the truth is IBM has an enormous consulting and services division, and that division does lots of work with lots of different customers. Further, that division is *very good* about capturing lessons learned and re-using knowledge gained. The various patterns and the “anti-patterns” we spoke of earlier are all part of this collective knowledge. So to say IBM brings a “sizeable storehouse of knowledge to the table” is not just marketing fluff. There is reality behind that statement.



+ SOA / ESB

- Proven platform
- Proven middleware
- Viable and powerful combination

Questions? Comments?

20

IBM Americas Advanced Technical Support
Washington Systems Center, Gaithersburg, MD

© 2007 IBM Corporation

We close down the workshop. We close with a final statement about the role of Z in this SOA / ESB picture. All along we've been pointing out how System z is a perfectly viable platform in this arena, and in fact is more than just viable, it is proven. Much of the SOA and ESB solutions are based on proven z/OS middleware solutions. While SOA/ESB is definitely not "just for z/OS," it is true that z/OS can and should play as appropriate in your solution design.

That's it. We're done talking. ☺

End of Unit