The Good, The Bad and The Ugly: Mainframe SOA Implementation Best Practices

Mainframe Integration Approaches

SOA software

Enterprise SOA

Migrate parts

Replace

SOA Enable

SOA Goals

- Reduce cost through reuse
  - Build applications faster
  - Use existing business logic rather than rewriting each time
  - Minimize cost of maintenance and upgrade by allowing incremental updates
- Increase agility to better align IT and the Business
  - Allow rapid change through business process management and composition tools
  - Allow incremental updates to enterprise applications
  - Minimize change cycles with business granular interfaces
- Reduce the risk, fragility and complexity of integration by improving interoperability through standards
  - Reduce investment in and risk of brittle proprietary integration techniques and technologies
  - Reduce frequency of data error caused by duplication

SOA Challenges

SOA software

- End-to-end security - trust and protect the privacy of message senders, receivers, and content
- Identify, manage, and repair exceptions as they occur
- Reliability and performance of a distributed set of services and consumers
- Interoperability between different platforms and technologies
- Decoupling of services and consumers
- Measure and prove the business value of SOA to offset cost concerns
- Control of (govern) the proliferation of duplicate or otherwise unnecessary services
- Facilitate the identification of appropriate services by potential users to reduce initial development cost
- Manage the lifecycle of services to minimize the cost and risk of ongoing maintenance and change
- Simplify the actual USE of appropriate services (decoupling location, transport, policy, standards, messaging styles)

Top Pitfalls of Mainframe SOA

SOA software

- Processing off of the mainframe
- Edge interfaces
- Middle tier servers
- Off mainframe security or proprietary security
- Fat client development
- Ignoring human assets
- Uni-directional integration
- Lack of centralization
- Lack of end-to-end monitoring, integrated testing, and error logging
- Doesn’t get the big picture

Mainframe SOA… Can I have it all?

SOA software

Is it possible to incorporate the mainframe into an enterprise wide SOA yet still get all the benefits of the mainframe…performance, reliability, scalability, manageability and cost effectiveness?

Yes.

Here's how…

- Run your SOAP stack on the mainframe.
- Take advantage of the power of the mainframe – reliability, performance, scalability, manageability and cost effectiveness.
- Put your SOA next to your enterprise assets.

Enterprise SOA… theory and practice.

SOA software

Enterprise SOA must address two different sets of issues:

1. Essential Components (what you need to build an SOA)
2. Practical Implementation (what you need to run an SOA)

Essential Components of Enterprise SOA

SOA software

Security

Support for Architectural Standards

Monitoring, Logging & Audit Controls

Policy Management

SOAP & XML Capability

Development Tools

Change & Release Management

Workflow Management

Registry

SOA is an Architectural Framework

SOA software

- Service Oriented Architecture is a framework for the loose coupling of services. All SOA implementations have the following key characteristics:
  - Self-describing interfaces using Web Services Description Language (WSDL)
  - Request/response messages using SOAP
  - The services are documented in a registry (UDDI)
  - The service is associated with a quality of service (QoS) by means of a Policy. The key QoS element is security

Security

SOA software

- You'll need to support Authentication, Authorization, Integrity, Confidentiality and Non-Repudiation. WS-Security provides all of these
  - SAML (Security Assertion Markup Language) implements authentication. Can also use along with RACF or LDAP
  - XACML (eXtensible Access Control Markup Language) implements authorization
  - Integrity is implemented with XML Signature. This is a pre-requisite for SAML
  - Confidentiality is implemented with XML encryption. It allows for flexible encryption (only encrypt what needs to be encrypted) and substantially reduces CPU compared to SSL (which encrypts everything on the wire)
  - Non-Repudiation is dependent on public-key cryptology. You can't deny (repudiate), because your key is half of the only pair able to encrypt/decrypt
- Using WS-Security your mainframe SOA will be able to interoperate with all other SOA participants. Isn't that the point of SOA?
- You'll need to implement WS-Security on the mainframe to eliminate "last mile" vulnerability.

Policy Management

- A Policy Definition Point (PDP, for example, SOA Software's Policy Manager) is used to define policies for services.
- A Policy Implementation Point (PIP) implements policies.
- WS-SecurityPolicy indicates the policy assertions which apply to Web Services Security.
- WS-PolicyAttachment associates the WS-Policy with the WSDL.
- WS-Policy externalizes the rules, without it your developers are required to hand code your policies, with it a PIP enforces the policies external to your programs.
- WS-Policy is essential for proper SOA governance

Monitoring, Logging and Auditing

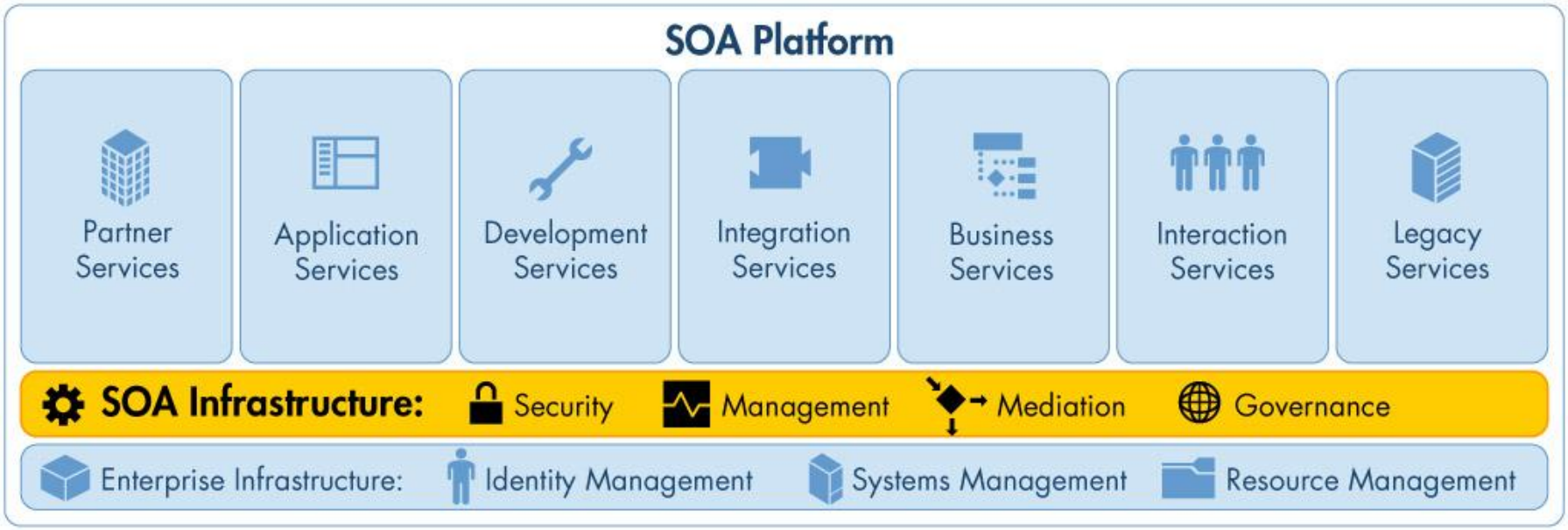
- Monitoring, logging and auditing are necessary to measure and manage performance and diagnose and debug problems.
- Measurement is the key to SLA management. Without monitoring you won't be able to measure your effectiveness.
- Monitoring, logging and audit are essential to make a solution governable.

Registry

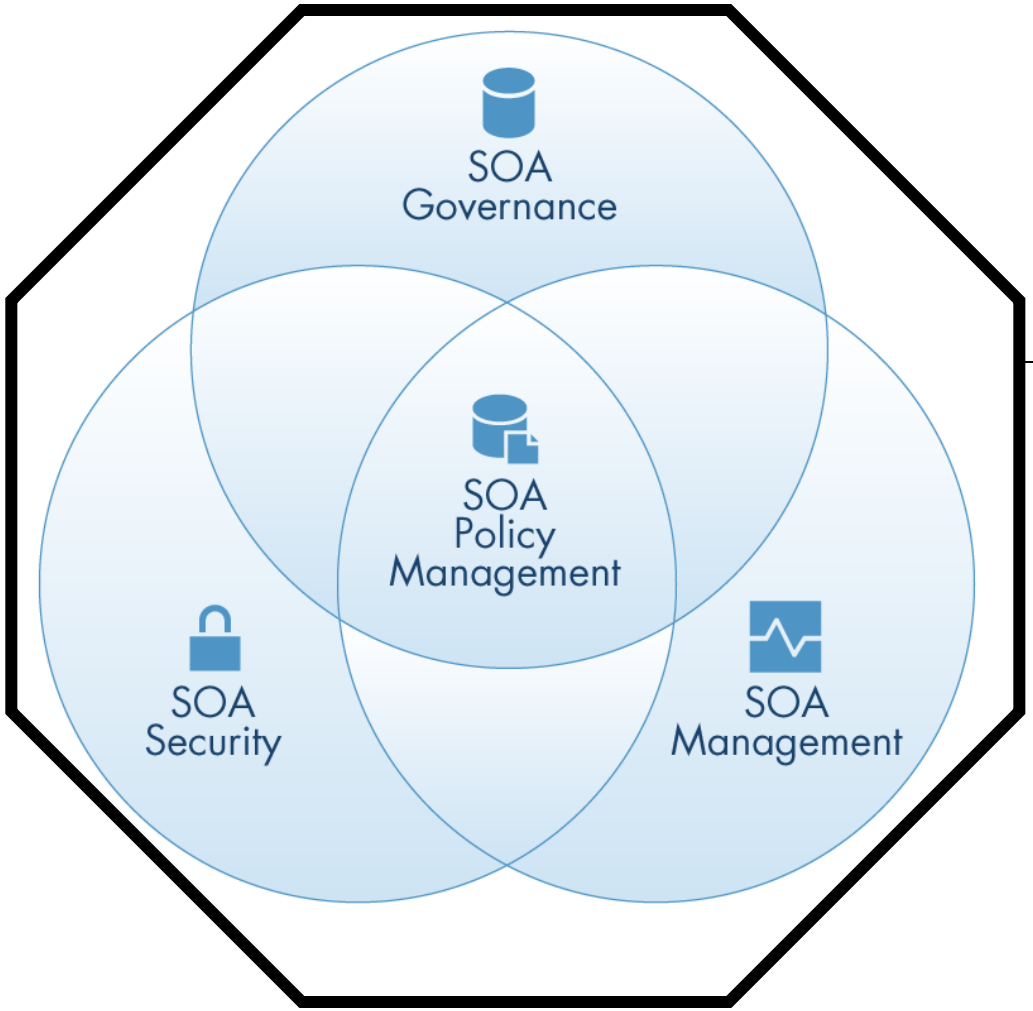
SOA software

- A Registry enables governance (you can't govern what you don't know).
- A Registry enables reuse. You can't reuse services that you can't find.
- All services need to be automatically documented in a registry.
- A registry is the foundation of a Service Oriented Architecture, an architect won't design a building that lacks a foundation

The SOA Platform

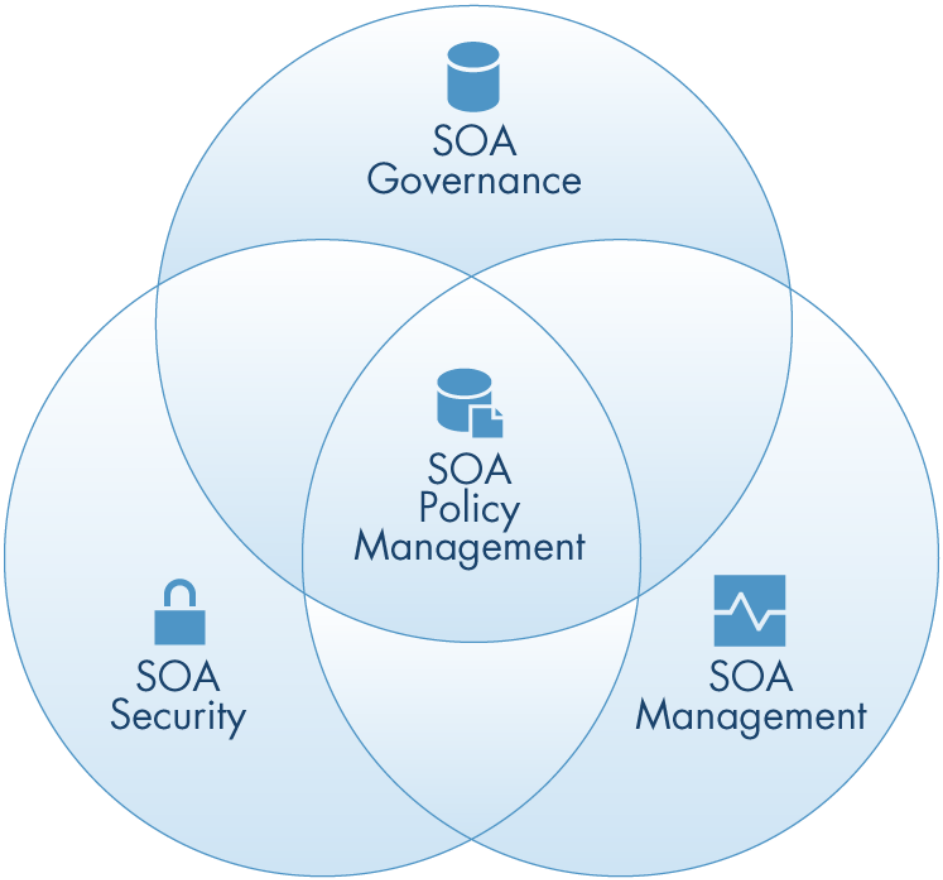


SOA Infrastructure Solutions



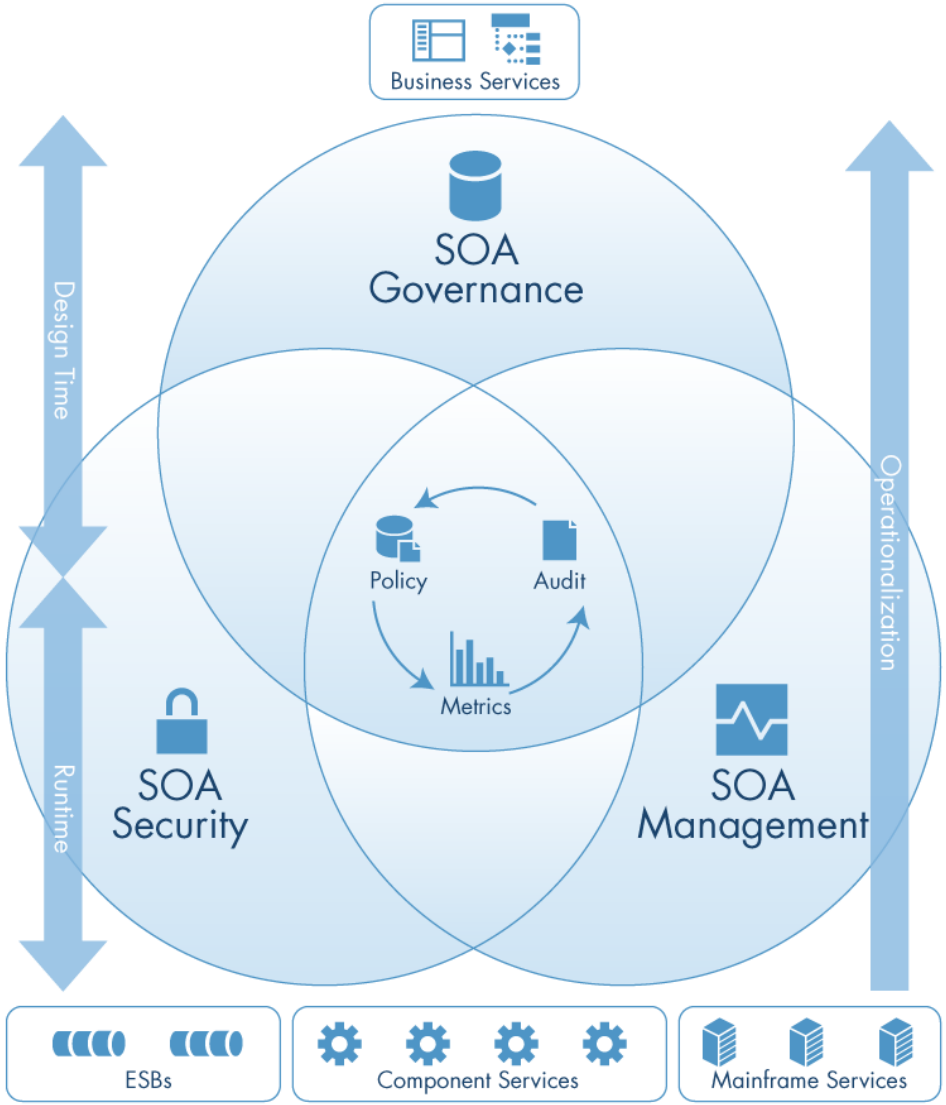
This is the Governance ring

SOA Infrastructure Solutions



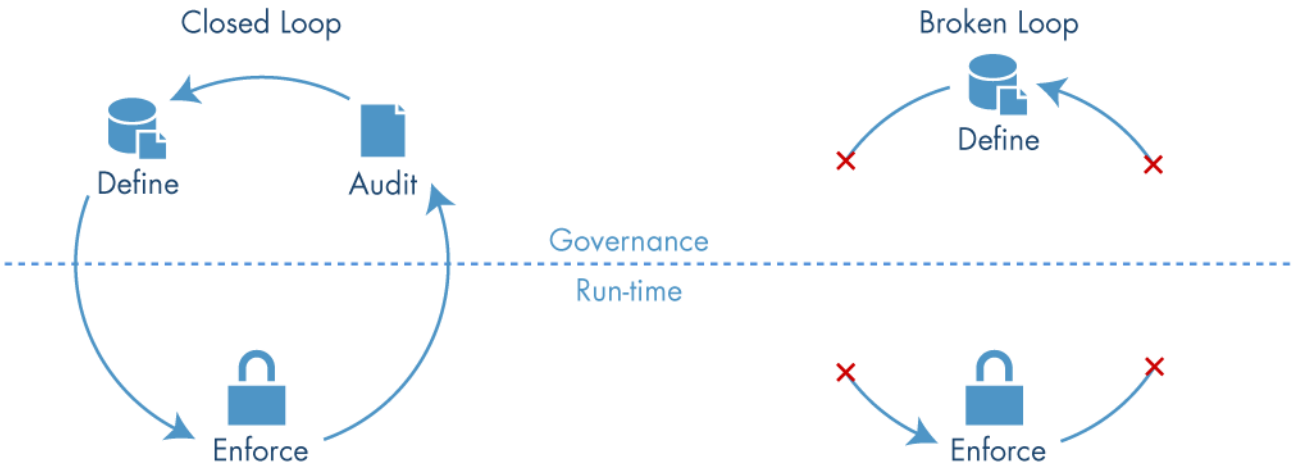
- SOA Infrastructure includes Governance, Management and Security linked together through SOA Policy Management
- Governance offers no value without a runtime solution to enforce policies and feed back metrics and compliance data
- Runtime solutions (security and management) offer minimal value without central policy control and value-added service governance capabilities

Standards-based Closed-loop SOA Infrastructure

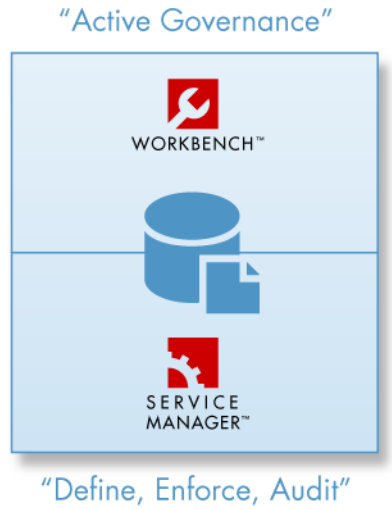


- Closed loop means:
  - Defining and managing actionable policies in a governance solution at design-time
  - Enforcing these policies via deep integration with a management solution at run-time
  - Auditing that these policies are being enforced
  - Using industry standards (WS-Policy, WS-MEX) where appropriate for metadata interchange
- Closed loop infrastructure enables demand and Value Management
  - Collect performance, usage and exception statistics at run-time
  - Track these statistics via the governance solution
  - Use live, audited information to drive value-based decisions about the effectiveness of different services and organizations
  - Provide developers with up to the minute information about a service in runtime to inform their decisions about which services to use
  - Manage supply and demand to ensure maximum efficiency and benefit from SOA

Closed-loop vs Broken-loop



- Integrated (closed-loop) solutions are best-of-breed
- There are no examples of integrated standalone solutions in production



Enterprise SOA…Practical Implementation

SOA software

Now that we understand the essential components, let's examine some practical aspects of Enterprise SOA.

How do you keep the mainframe's speed and reliability?

SOA software

Run your SOAP stack on the mainframe.

It doesn't make sense to integrate the mainframe into an SOA by using off-mainframe middleware and/or middle tier servers. Implementing mainframe SOA doesn't have to mean introducing bottlenecks and points of failure.

How do you ensure a low MIPS overhead?

- Be fanatical about performance – your SOA will fail if it drives MIPS usage up by more than a small percentage.
- Use compiled languages.
- Take advantage of hardware assistance for cryptology.

Scalability and Failover

It’s easy to scale down. Scaling up takes planning.

- Exploit SYSPLEX/CICSPLEX architecture.
- Run under WorkLoad Manager.
- Use IP port sharing and SYSPLEX Distributor to handle workload.
- Think of your SOA as supporting your business. Use monitoring tools to monitor your SOA environment through a “dashboard”.

Operations

- How will you diagnose production outages?
- What was in that failing SOAP request?
- How will you manage changes?
- How is the environment running?
- Are you meeting your SLAs?
- How will you prevent unauthorized access?

Standards

Q: How do you make sure your implementation doesn't get left behind?

A: Standards.

- Standards reduce the risk of committing to a technology that will be unsupported in future.
- Standards allow greater choice of vendors and the availability of a larger talent pool.

Human Assets

What good is leveraging technology assets if you ignore human assets? Task the right people with the right jobs.

- Mainframe developers expose mainframe applications as services.
- Distributed developers incorporate services into composite applications.
- Ignore this at your own peril…training, development and support costs can far exceed the cost of the solution itself.

Summary

Do it the right way from the start.

- Avoid pitfalls
- Avoid piecemeal solutions
- Leverage all of your assets, technological and human
- Make your SOA work for you