Will the promise of Service Oriented Architecture ever be realised?

Stephen Smith
State Director NSW
Enterprise Architects

stephen.smith@enterprisearchitects.com
The development of “Service Oriented Architecture” based on Web Services Standards has generated considerable excitement within the consultant and vendor communities. These groups consistently assign enormous benefits to these technologies and techniques, but how easy will it be to achieve these benefits?

The available techniques for designing and managing SOA based systems assumes that these activities will occur within complicated management systems when in fact we operate in increasingly complex management systems.

Without a change in the way we approach design and management, realisation of the SOA promise will be limited.
What is SOA?

- An *implementation agnostic* architectural style based on the formal *separation of concerns*.

- Breaks down automation into distinct logical units that can be *distributed and recombined to solve different problems*.

- An approach to architecting business operations into *discrete services* in such a way as to promote their recombination to support business agility and flexible automation.

- Based on key principles…
# Principles of SOA

<table>
<thead>
<tr>
<th>Principle</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Loose coupling</strong></td>
<td>Services maintain a relationship that <em>minimises dependencies</em> and only requires that they retain an awareness of each other.</td>
</tr>
<tr>
<td><strong>Service contract</strong></td>
<td>Services adhere to a communications agreement, as defined collectively by one or more service descriptions.</td>
</tr>
<tr>
<td><strong>Autonomy</strong></td>
<td>Services have control over the logic they encapsulate.</td>
</tr>
<tr>
<td><strong>Abstraction</strong></td>
<td>Beyond what is described in the <em>service contract</em>, services hide logic from the outside world.</td>
</tr>
<tr>
<td><strong>Reusability</strong></td>
<td>Logic is divided into services with the intention of promoting reuse.</td>
</tr>
<tr>
<td><strong>Composability</strong></td>
<td>Collections of services can be coordinated and assembled to form composite services.</td>
</tr>
<tr>
<td><strong>Statelessness</strong></td>
<td>Services minimise retaining information specific to an activity.</td>
</tr>
<tr>
<td><strong>Discoverability</strong></td>
<td>Services are designed to be outwardly descriptive so that they can be found and assessed via available discovery mechanisms.</td>
</tr>
</tbody>
</table>
Services can encapsulate varying amounts of logic

Services use messages and contracts

- name
- location
- data exchange
- requirements

service description for service B
What is SOA Nirvana?

- The state of “Nirvana” described by SOA proponents is one where components of business logic, provided as services, can be discovered, negotiated, bound and execute in real-time, allowing the creation of rapidly reconfigurable, low cost automation solutions.
Benefits of an SOA approach

- Reduced Costs
  - Re-use existing assets
  - Amortise over more users

- Reduced Risks
  - Invest in better management
  - Lower development risk
  - Invest in better DR
  - Lower "modernisation" risk.

- Increased Agility
  - Enable new channels faster
  - Enable new processes faster
  - Enable new partners faster
  - Enable business control of BPM
  - Modify existing processes faster

- Lower "modernisation" risk.
  - Lower development risk.

SETE 2007 Sydney
Standards - we still have a way to go...

- “It is very difficult (and some would say impossible) to include sufficient information about the meaning of a message such that a computer can understand it, regardless of its language or data format”.

- “We now have the technology to build loosely coupled agile approaches to IT, yet the struggles to automate the understanding of messages between providers and consumers remain”. (Bloomberg & Schmelzer, 2006).
What is the event driven model?

This architectural pattern may be applied by the design and implementation of applications and systems which transmit events among loosely coupled software components and services.

Building applications and systems around an event-driven architecture allows these applications and systems to be constructed in a manner that facilitates more responsiveness, because event-driven systems are, by design, more normalized to unpredictable and asynchronous environments.
An event orchestrator ties together services originating from diverse sources.
Adopting an SOA changes the way that we need to manage all types of services.
SOA requires that multiple service suppliers are managed. How does this happen across multiple management domains?
Welcome to Cogworld

- Function stable and understood
- Components and relationships stable and understood
- Behaviour predictable
- System can be optimised for primary productive value(s)
- It's complicated but understandable

Welcome to Bugworld

- Function and models constantly need changing.
- Components and relationships change and adapt
- Behaviour not reliably predictable
- Attempts to optimise usually short-lived
- It constantly adapts to internal and external feedbacks.
Where does your design process operate?

- We think and solve problems as if we live in Cogworld
- But the organisations that we work in are more like Bugworld
Most architecture operates in complex environments

Customers, products & markets are changing
In-sourcing, outsourcing and globalisation
New channels and distribution
Changing demographics
Changing technology & techniques
Changing environment
Regulatory changes
Competing, changing & un-resolved objectives & priorities.
Changing personalities and politics

GETTING MORE COMPLEX

SETE 2007 Sydney
What are the barriers to SOA implementation?

- If we do not have mature approaches for designing and defining semantic alignment between services we are required to manage a set of common semantic models over time and across a complex environment so that we can:
  - Coordinate multi-team and multi-year investment and development agendas
  - Manage service provision in a complex multi-sourced environment.
  - Manage projects when build depends on assembling components from across a complex program.
  - Establish monitoring and system management processes in a highly interconnected, distributed and heterogenous environment.
  - Improve the robustness of our application development process so partial (component) failure conditions do not cause catastrophic system failures
  - Improve the availability of systems that are used in an enterprise wide fashion including ensuring scalability of shared components.
  - Resolving on-going issues of functional gaps and overlaps; Ensure ongoing alignment of needs.
What is the future of SOA?

- SOA techniques may offer value to organisations where the architecture for the entire system is closely managed over its entire lifecycle. However, this approach will not deliver many of the benefits identified by proponents of SOA!

- In organisations where solutions are assembled from multiple sources over long and complex projects, the current level of design tool and technique maturity does not support the realisation of SOA, “Nirvana”.

- I feel it is too early to know if SOA will thrive in the way envisaged by SOA Nirvana proponents.
Questions?