Web Services: How Do You Actually Derive Value?

David Moskowitz
Productivity Solutions, Inc.
Agenda

- Common vocabulary
- SOA effectiveness: How do you know...?
  - What's really important
  - Security, standards, and services
- Cooperating and collaborating services
- What's the value?
- How do you really make it work?
Architectures…

- Software architecture (50K ft level) defines system components & how they interact
  - Components aren't any specific type of object
  - Abstract "modules" deployed as one or more units on one or more servers

- The architecture defines the externally visible properties and their relationship(s).
  - Structure, relationships **and** patterns
  - Defines expectations expressed as a contract
Architectures…

- Service Oriented Architecture (SOA) formally separates interface & implementation
  - Service-consumer views service as an endpoint
  - Supports contract with defined request & response msgs

- SOA dictates a different paradigm:
  - Find, Bind, and Execute (FBE)
  - Web Services take this one step further and define a formal mechanism for the FBE process

- SOA expresses the definition of loosely coupled (software) business service components/modules that map a business process or problem space
On the Same Page

- At a high level, a service is software that conforms to specific protocols & standards
- More specifically, a service is a stateless "component"
  - It accepts a request &...
    - ...returns 1 to n responses
    - through a set of defined interfaces
- "Architecture" defines how services interact and are sequenced (orchestration)
Architectures

Web Service  SOA  Architecture

Implements
The SOA Roadmap

- **Service Provider**
- **Service Requester**
- **Service Broker**

- **UDDI**
- **WSDL**
- **SOAP**
Web Services Protocol Stack

- **Service Transport:** transports messages between apps
  - HTTP, SMTP, FTP, & Blocks Extensible Exchange Protocol (BEEP).

- **XML Messaging:** Encodes messages in a common format
  - XML-RPC, SOAP and REST.

- **Service Description:** Describes service’s public interface
  - WSDL

- **Service Discovery:** Centralizes services into a common registry/directory
  - UDDI

- The Web service protocol stack also includes a whole range of recent protocols: WSFL, SOAP-DSIG and more
Reasons for SaaS

- SaaS == Software as a Service
- You might not have to develop it, if it is available as a service (on-demand or subscription-based)
  - Reduced startup costs
  - 21st Century version of time-sharing
  - Reduced upgrade costs
- Additional revenue source
  - Piracy prevention???
Reasons for SaaS

- Capitalize on diverse resources including core competencies
  - It's about business, not technology
- Evolve to meet customer needs faster
  - Business needs, not technical
- Easier to sell
- Easier to install
- Easier to integrate
- Cheaper to build

Compared to a classic application
Web Services Advantages

- Web services provide interoperability between various software applications running on disparate platforms.
  - Loose coupling allows either end to change
- Uses open standards & protocols.
  - Protocols and data formats are text-based where possible, making it easy for developers to comprehend.
- Keep pace with global competition
Web Services Advantages

- Web services **should** easily allow software and services from different companies and locations to be combined to provide an integrated service.
- Web services allow the reuse of services and components within an infrastructure.
- Using HTTP, Web services can tunnel through many firewalls
- Immutable interfaces
Web Services Disadvantages

- Learning curve associated with the standards
- Coding (and architecture) learning curve
  - Depending upon the tool set: code complexity.
- Availability (no site is 100%)
- Using HTTP, Web services can tunnel through many firewalls
  - Rules intended to block or audit communication between programs on either side of the firewall.

Immutable interfaces
Web Services Disadvantages

- Poor performance compared to other distributed computing approaches (RMI, CORBA, or DCOM).
  - Common trade-off with text-based formats.
  - Note: Design goals for XML do not explicitly include parsing efficiency.
- Some of the standards aren't standards, yet
- Standards for transactions don’t exist...
  - ...or still in infancy compared to mature distributed computing open standards (CORBA).
Ideas

- Federated identity across services
- Types of services
  - Self-contained (e.g., Amazon) or for internal use only
  - Multi-vendor interoperation
  - Standards
- Web services security in each environment
- Services are about the business
- BPEL (and BPEL4WS) is your friend
  - What about BPMN and BPEL for people: BPEL4People
- Just a thought...
  - What happens if we don't do/use SOA?
How Do You Know It Will Work?

- Ask critical questions in 5 areas
  - Architecture/design
  - Other technical
  - Organization
  - Market
  - Regulatory

- Developing and deploying services takes time & coordinated effort.
  - Governance

- Your situation might call for weighting
Will It Work: Considerations

- Define risk for each of 5 areas (description)
- How likely (low med high – lmh)
- How bad would it be (lmh)
- What would trigger it
- What would happen if...
- Metric to determine if & how you will know...
- What are you going to do about it?
The 5 Area Questions

- **Architecture/Design**: What factors prevent the architecture from meeting specific goals, or expectations? (performance?) (targets?)
- **Other technical**: If architecture/design works, what other technical issues could prevent it from working?
- **Organization**: Which people (or department) could obstruct, slow, or kill the project?
- **Market**: What could competitors do that reduce effectiveness of the result?
- **Regulatory**: What changes in law, organizational guidelines or mandates could adversely impact the project?
The Right Questions

- How to reduce integration costs, increase both reuse and business agility?
- What standards will you use?
  - How will you select them?
- How will you assure adherence to standards?
  - Corporate and regulatory compliance?
- How will you discover services...
  - ...and assure they are re-used?
- What services are already available (inside or...)?
More Right Questions

- What do you do about SLAs?
  - How do you manage/negotiate them?
- H2 connect/integrate internal users?
  - External?
  - Is "how quick" an issue – if so, how...?
- Who authorizes "publish" for internal use?
  - External?
  - ...and how – what is the process?
- How will we deal with "system" governance?
The Ultimate Question(s)

- How do/will you measure success?
  - On-time
  - Under budget or profitable
  - Number of people using it
  - Team was agile
  - Team learned what works and what doesn't
  - Serves as a good bad example
  - does it map business need or business value
  - Better ROI, ROC, or competitive position
  - ???
SOA Deployment Considerations

- Impacts of/on corporate culture
- Business justification required
- Regulatory compliance == SOA opportunity
- SOA designed to operate in heterogeneous federated environments
  - Blur distinction between network & software from the perspective of functionality
- Security issues...
Web Services and Security

- Requests (can) pass through gateways that either aggregate or split messages
  - Add multiple points of identity scrutiny and message integrity
    - Possible to spoof identify without proper safeguards
- Most WS security focuses on authenticate requestor, validate message integrity, and provide security policy headers
- Is this enough?
Web Services and Security

- The previous issues are resolvable **PROVIDED** you don't assume one size fits all
  - SSL or XML signature
  - PKI (could bog down in high volume)

- Layered defenses
  - Outer firewall for perimeter, inner gateway to decrypt and authenticate messages & sender
  - Check policy – allowable request from source
  - Virus/Trojan/worm protection
Web Services and Security

- Business context
  - Trusted source
  - Number of endpoints increases complexity
- Unless you have access to a private network (WAM!net?) then you might not want to offer sensitive WS in high volume, today.
- Every standards iteration gets better
  - WS-I Basic Profile
  - SAML (Security Assertion Markup Language)
  - Still need standards for trust exchange
Follow the Yellow Brick Road

- The goal is a framework that makes future applications faster, easier, cheaper
- Open the doors to reuse of services and data across...
  - ...lines of business
  - ...B2B and B2C
- Provides information sharing & collaboration
- Single sign-on
Yellow Brick Road

- Determine how to wrapper existing applications to work in & within SOA
  - Map services to data silos
- Separate data and logic – but it's not about CRUD
- Aggregation via orchestration
  - Not one-off, but façade
- Eventually, services need to be extensive
- SOA allows/provides enterprise governance
Service Architectures as Patterns

- Patterns have descriptive conditions, entry exit conditions, *etc.*
  - What service patterns exist in your organization?
- Service architectures as APIs – define the parameters of the service and what it does
  - If you can describe in WSDL...
- Re-use isn’t free, it takes work to discover what is there and how to use it.
  - Search for “it”
Not the Best Approach…

- NetWeaver rollout, SAP announced 10,000 services for 1,000 applications in 30 industry groups?
  - Is this a plus or still a legacy architecture wrapped up to look like SOA?

- If architectures define structure, relationships and patterns, where is the structure in 10,000 services...?
  - Frankenstein's Monster or...
Things to Consider

- WS still evolving, so are best practices
- Service definitions can be volatile
  - Needs to be a change management process without propagating volatility
- Consider WSDL, UDDI, & SOAP
  - SOAP and WSDL have inconsistencies
  - Vendor implementations differ
  - Interoperability not guaranteed
- Both Java and .NET offer some protection and are getting better.
Final Thoughts…

- KISS still works!
- It's Service-oriented, not tech-oriented
  - Service linked to business objectives and business processes
- SOA is not just about software…
  - It's architecture: everything that surrounds s/w
    - New infrastructure layer(s) address SOA requirements
    - Hardware accelerators and security for XML
  - Business analysts: what can SOA do for me?
    - This is part of the key to change!
Questions ???

If you don't ask, who will?
If not now, when?

There aren't any dumb questions.
The only dumb question is the one not asked!
Thank You

For more information:
David Moskowitz
Productivity Solutions, Inc.
147 Ashland Avenue
Bala Cynwyd, PA 19004
+1-610-664-7726
davidm2@usa.net
SkypeID: davidmosk