JBoss World 2006 LAS VEGAS

Real-World ESB

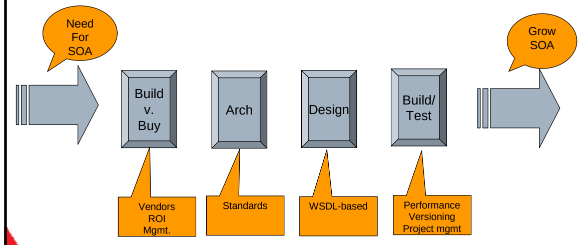
Vijay Venkatesh
Director, EAI
Pearson Education
Soa_arch@comcast.net

Agenda

- Part I – Getting there
  - ✓ ESB & SOA
  - ✓ ESB concepts & features
  - ✓ Initial selection, deployment & maintenance
- Part II – Growing SOA
  - ✓ Increasing scope of the ESB
  - ✓ Emerging standards

Best Practices
&
Lessons learnt

Lessons Learnt / Best Practices



Evolution of the ESB

- 1st phase – Programmatic Integration
  - ✓ CORBA/RMI/Sockets/C++
- 2nd phase – EAI ($1b - $1.5b market in '99)
  - ✓ Non-standard & proprietary API
  - ✓ Non-standard & proprietary adapters
  - ✓ Non-interoperable messaging (unlike JMS)
  - ✓ Expensive & bloated
- 3rd phase – ESB
  - ✓ Support standards, SOA, Reduce $, Increase speed to market
  - ✓ ESB = SOA + WS

ESB key features

- Messaging & adapters
  - ✓ JMS (vendor-agnostic), HTTP(s),FTP,SMTP
  - ✓ JCA & packaged apps
- Orchestration
  - ✓ Workflow creation, modeling & engine - BPEL 1.1 + Process persistence
  - ✓ Process monitoring & controls - BAM
- Policy
  - ✓ WS-Security
- Deep XML Support
  - ✓ Format & content transformation
  - ✓ WSDL design, develop & service implementation
  - ✓ Address, content & meta-data based routing
  - ✓ SOAP, WSDL, UDDI support

ESB Offerings Then and Now

2004

IBM
sonic
CAPE CLEAR
PolarLake

2006 (New)

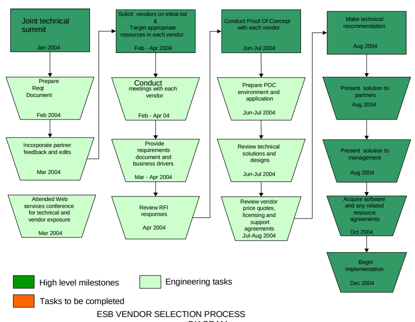
Sun Microsystems
Spiritsoft
TIBCO
ObjectWeb
webMethods.
IBM
bea
IONA
JBoss
Project Open ESB
ORACLE
Fiorano
SOFTWARE AG

Real-World ESB

Part I….Getting There

7

ESB Build vs. Buy Timeline Best Practice


Joint technical summit
Jan 2004
Prepare Reqt Document
Feb 2004
Incorporate partner feedback and edits
Mar 2004
Attended Web services conference for technical and vendor exposure
Mar 2004
Solicit vendors on initial list & Target appropriate resources in each vendor
Feb - Apr 2004
Conduct meetings with each vendor
Feb - Apr 04
Provide requirements document and business drivers
Mar - Apr 2004
Review RFI responses
Apr 2004
Conduct Proof Of Concept with each vendor
Jun-Jul 2004
Prepare POC environment and application
Jun-Jul 2004
Review technical solutions and designs
Jun-Jul 2004
Review vendor price quotes, licensing and support agreements
Jul-Aug 2004
Make technical recommendation
Aug 2004
Present solution to partners
Aug 2004
Present solution to management
Aug 2004
Acquire software and any related resource. agreements
Oct 2004
Begin Implementation
Dec 2004
High level milestones
Engineering tasks
Tasks to be completed

ESB VENDOR SELECTION PROCESS DIAGRAM

8

ESB Build vs. Buy Lessons Learnt

- Corporate standards
  - ✓ ESB did not exist 4-5 years ago, be prepared to have to change corporate standards for MOM or messaging if they exist; **these standards are probably stale**
- Management
  - ✓ True ESB & SOA deployments ARE complex. Make sure management understands why. This will justify longer initial development & testing cycles.
  - ✓ Take the time to prepare ROI (ROI, NPV etc.) analyses & do take into account future development costs saved
    - For e.g. current SOA rollouts cost us 60-70% shorter development cycles
    - BPM & process-modeling could be a skill product managers can use

9

ESB Build vs. Buy Best Practice

4. Meeting Functional and Non-Functional Requirements				
(i)	Reliable messaging – asynchronous	3	2	3
(ii)	Reliable messaging – synchronous	3	3	3
(iii)	Transactions	3	2	3
(iv)	Orchestration – transactions	3	3	3
(v)	Orchestration – monitoring / execution	3	3	3
(vi)	Orchestration – modeling tools	2	3	3
(vii)	Xml / message transformations	3	1	3
(viii)	Standards-based xml transformations	3	1	2
(ix)	Administration – role based	3	3	3
(x)	Administration – queue manipulation (retry/delete)	3	3	3
(xi)	Administration – fault handling	3	2	3
(xii)	Administration – health checks	3	3	3
(xiii)	Administration – audit logging	3	3	3
(xiv)	Performance – projected performance	3	3	3
(xv)	Performance – high availability	2	3	3
(xvi)	Performance – future scalability	3	3	3
(xvii)	Security – IP authentication	3	3	3
(xviii)	Security – other authentication	3	2	2
(xix)	Security – WS Security (planned support)	3	3	3
(xx)	Security – message or transport level security	3	3	3
(xxi)	Interoperability – multi platform support	3	2	2
(xxii)	Interoperability – multi transport/protocol support	3	2	2
(xxiii)	Interoperability – industry standards support	3	2	2

10

ESB Build vs. Buy Best Practice

6. Proof-of-Concept and Demonstrations				
(i)	Technical kick-off meeting	3	2	3
(ii)	Installation and configuration of software	3	2	2
(iii)	Http to Http	3	2	2
(iv)	Http to Web Service	3	2	2
(v)	New Web Service	3	2	0
(vi)	Web Service to Web Service calls	3	2	3
(vii)	Data transformation	3	1	3
(viii)	JMS interoperability	3	2	3
(ix)	Content-based routing	3	1	3
(x)	Asynchronous end-to-end	3	3	3
(xi)	Team effectiveness	3	0	3
(xii)	Ease of use / usability	3	2	1
(xiii)	Flow and simplicity	3	1	1
(xiv)	Ability to meet current requirements	3	0	2
(xv)	Ability to meet future requirements (unstated)	3	0	2
(xvi)	Robustness (Bugs not found)	3	2	2

11

ESB Vendor Mgmt. Best Practice

- Most ESB vendors are still in their infancy
  - ✓ Pay close attention to PO process & contracts;consider language for escrow
  - ✓ Put strong verbiage for support, patch delivery (you will find bugs) & bug fixes;consider penalties for failed patches
- Proof-of-concepts
  - ✓ DO NOT start on an ESB/SOA project without detailed POCs with each vendor
  - ✓ Do extended POC with final vendor before project design begins

12

ESB Vendor Mgmt. Best Practice

- Select a solution based on open standards
  - ✓ ESB should promote standards;**stay away from proprietary solutions**
  - ✓ Risks of not doing so are vendor lock-in (just like the EAI market all-over again), expensive & inextensible solutions
- Avoid custom-adapters
  - ✓ These are often commercial and proprietary;may need to be updated & maintained, & require professional services
  - ✓ Design patterns like asynchronous messaging & SOA can be applied to .NET

13

Design-Time Best Practice

- Create WSDL-first services
  - ✓ Starting with XML Schema provides best reuse & extensibility, & true document-oriented services
    - **Saved roughly 50% new development time**
  - ✓ Can effectively de-couple design & implementation
    - Clients of the service can develop in parallel with the service developers (distributed computing environment)
    - WSDL can serve as requirements to developers & is portable

14

ESB Planning Lessons Learnt

- SOA & ESB development should promote business & development agility
  - ✓ Abandon waterfall development models; **adopt iterative design, build, test models**
  - ✓ Complex services will never be 100% done; **will need to be iteratively tested**

15

ESB Planning Lessons Learnt

- Project Planning
  - ✓ Level of effort estimates for SOA/ESB projects are different than J2EE; **plan accordingly**
  - ✓ Consider use of a dedicated TPM for ESB vendor & SOA project mgmt.

16

Design-Time Best Practice

- Avoid the auto-generated WSDL from source code (aka “low-hanging fruit”)
  - ✓ Promotes RPC-style services
  - ✓ Schema features such as enumerations, ranges, cardinality may not be supported
    - Some of the inherent power of schema-based design is lost
  - ✓ Design & development cannot be separated
  - ✓ May lead to inter-op issues
    - Java * .NET

17

Design-Time Best Practice

- When designing WSDL-first services
  - ✓ If not using a centralized governance model
    - Document using <xsd:annotation & xsd:documentation> within the schema so that certain schema-design principles may be understood
    - Create an architectural document that outlines your WSDL & schema policies
      - ✓ E.g. Doc/Lit services must be used
      - ✓ E.g. In rev1, no security is required. For rev2, must support xml-encryption of <element1> etc.

18

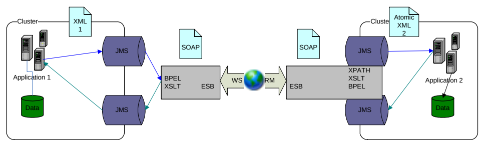
Design-Time Best Practice

- Transactions (Distributed & ACID)
  - ✓ WS-* appear promising; No silver bullet yet
  - ✓ Since vendor/api support will be rare, be prepared to roll your own
    - Investigate what your database can provide;PL/SQL is a vital tool
    - Design transaction support so that they can be called from within your application tier, **these can later be exposed as services**
      - ✓ E.g. PL/SQL that can be called by EJB or POJO, that can be exposed as services, that could be registered as compensation or transaction handlers

Testing Best Practice

- When creating orchestrations or complex services
  - ✓ Create test stubs with the same WSDL as your partner links, populate with dummy data
- SOA & ESB development is iterative – forego the distinct SDLC phases in favor of an iterative approach
- Include separate phases for integration, system & performance testing
  - ✓ High likelihood issues will be found during rigorous performance testing of ESB integration
    - For e.g. locking issues, thread-safety issues,3rd-party XSLT/DOM issues, memory & connection leaks

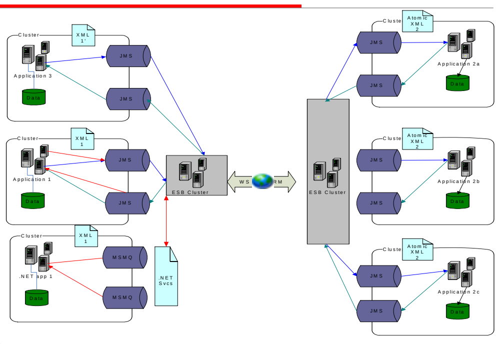
ESB Current State



Real-World ESB

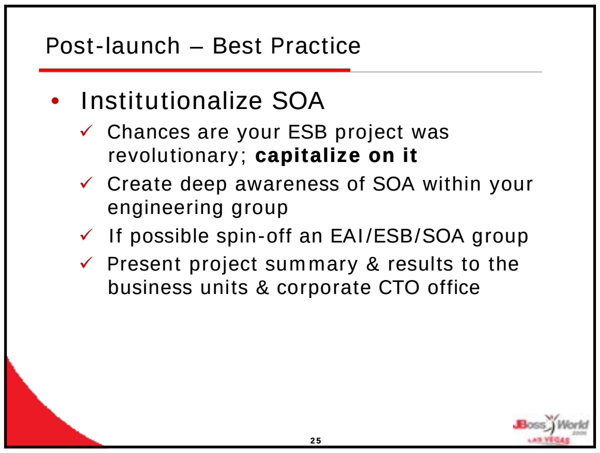
Part II – Increasing scope & growing the ESB….

ESB – Current state +



Post-launch Best practice

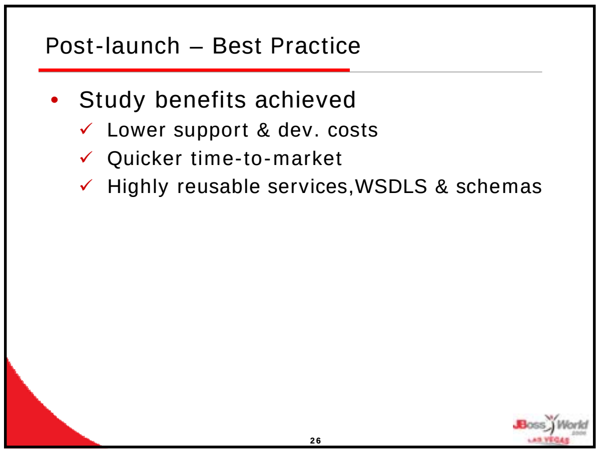
- Develop ESB run-book
  - ✓ Details on BPEL/RM data-models, SOP, manual message/orchestration resubmissions, logs etc.
  - ✓ Check on open issues,patches en-route & future requirements
- Do current gap analysis
  - ✓ Focus on features you needed but couldn't use in your current ESB; get this list to your ESB vendor asap



Post-launch – Best Practice

- Institutionalize SOA
  - ✓ Chances are your ESB project was revolutionary; **capitalize on it**
  - ✓ Create deep awareness of SOA within your engineering group
  - ✓ If possible spin-off an EAI/ESB/SOA group
  - ✓ Present project summary & results to the business units & corporate CTO office

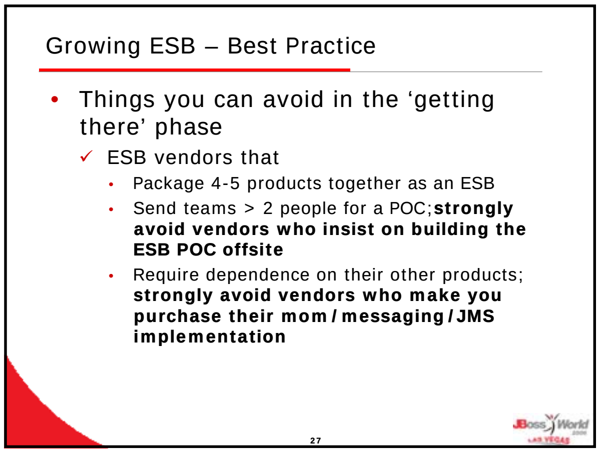
25



Post-launch – Best Practice

- Study benefits achieved
  - ✓ Lower support & dev. costs
  - ✓ Quicker time-to-market
  - ✓ Highly reusable services,WSDLS & schemas

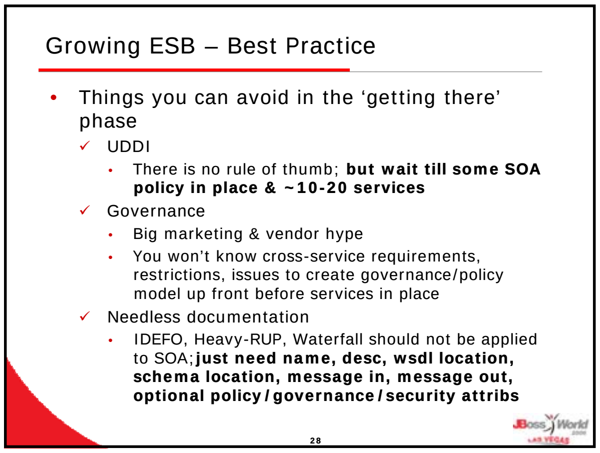
26



Growing ESB – Best Practice

- Things you can avoid in the 'getting there' phase
  - ✓ ESB vendors that
    - Package 4-5 products together as an ESB
    - Send teams > 2 people for a POC;**strongly avoid vendors who insist on building the ESB POC offsite**
    - Require dependence on their other products; **strongly avoid vendors who make you purchase their mom / messaging / JMS implementation**

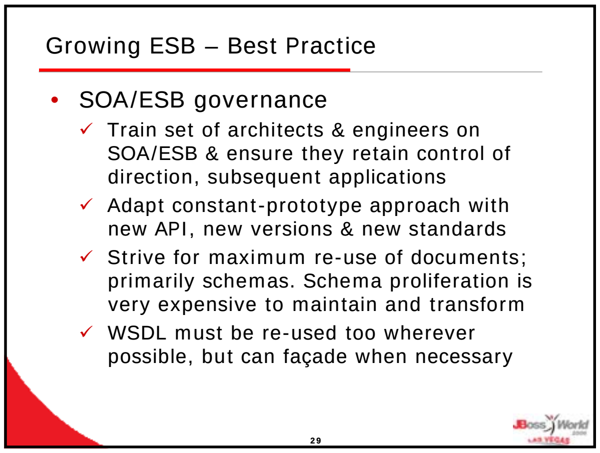
27



Growing ESB – Best Practice

- Things you can avoid in the 'getting there' phase
  - ✓ UDDI
    - There is no rule of thumb; **but wait till some SOA policy in place & ~10-20 services**
  - ✓ Governance
    - Big marketing & vendor hype
    - You won't know cross-service requirements, restrictions, issues to create governance/policy model up front before services in place
  - ✓ Needless documentation
    - IDEFO, Heavy-RUP, Waterfall should not be applied to SOA;**just need name, desc, wsdl location, schema location, message in, message out, optional policy / governance / security attribs**

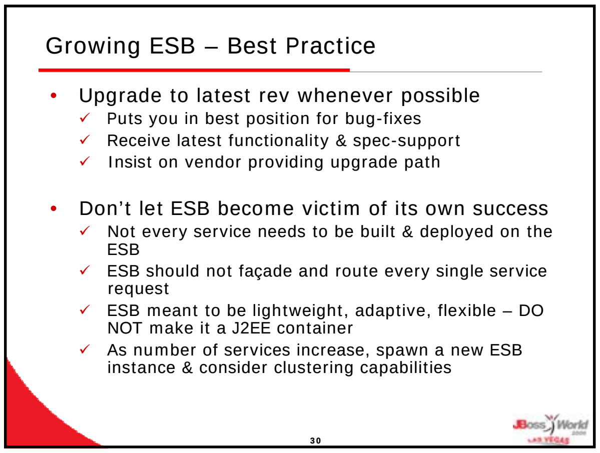
28



Growing ESB – Best Practice

- SOA/ESB governance
  - ✓ Train set of architects & engineers on SOA/ESB & ensure they retain control of direction, subsequent applications
  - ✓ Adapt constant-prototype approach with new API, new versions & new standards
  - ✓ Strive for maximum re-use of documents; primarily schemas. Schema proliferation is very expensive to maintain and transform
  - ✓ WSDL must be re-used too wherever possible, but can façade when necessary

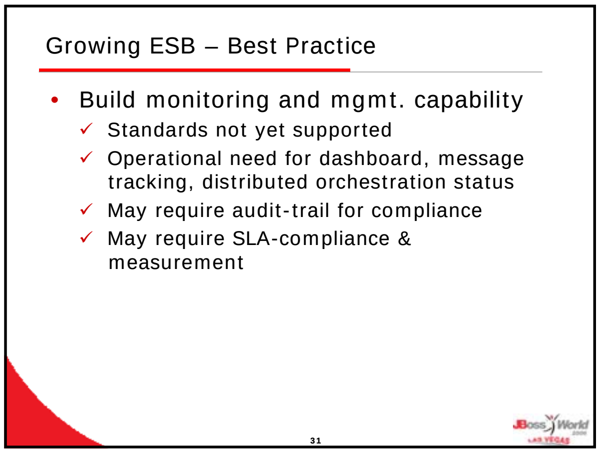
29



Growing ESB – Best Practice

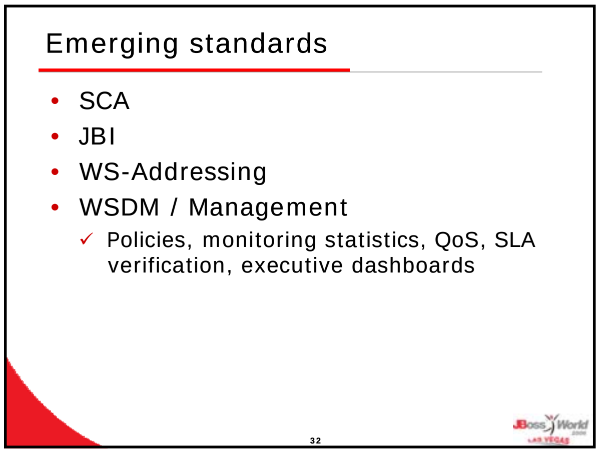
- Upgrade to latest rev whenever possible
  - ✓ Puts you in best position for bug-fixes
  - ✓ Receive latest functionality & spec-support
  - ✓ Insist on vendor providing upgrade path
- Don't let ESB become victim of its own success
  - ✓ Not every service needs to be built & deployed on the ESB
  - ✓ ESB should not façade and route every single service request
  - ✓ ESB meant to be lightweight, adaptive, flexible – DO NOT make it a J2EE container
  - ✓ As number of services increase, spawn a new ESB instance & consider clustering capabilities

30



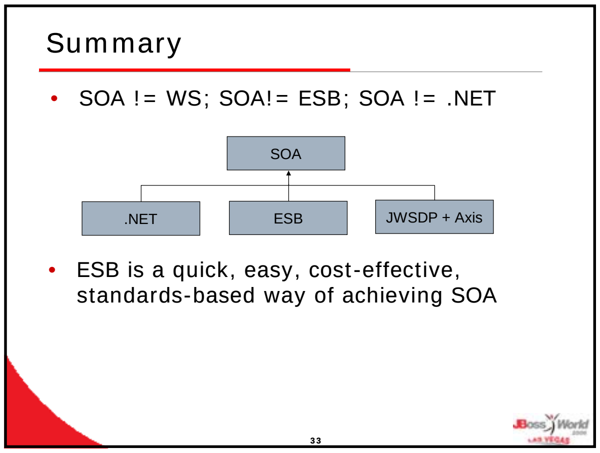
Growing ESB – Best Practice

- Build monitoring and mgmt. capability
  - ✓ Standards not yet supported
  - ✓ Operational need for dashboard, message tracking, distributed orchestration status
  - ✓ May require audit-trail for compliance
  - ✓ May require SLA-compliance & measurement



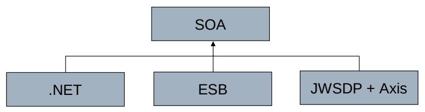
Emerging standards

- SCA
- JBI
- WS-Addressing
- WSDM / Management
  - ✓ Policies, monitoring statistics, QoS, SLA verification, executive dashboards

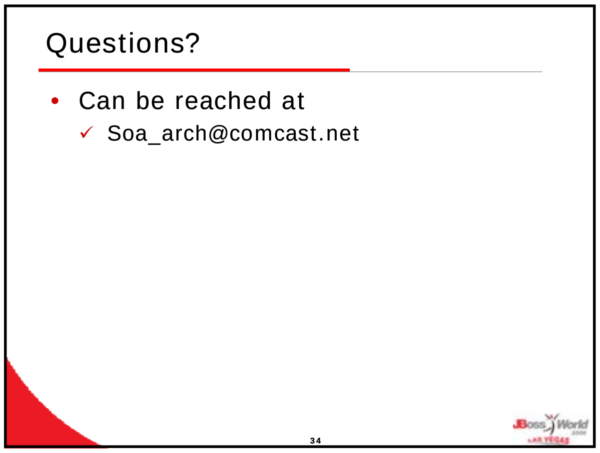


Summary

- SOA != WS; SOA!= ESB; SOA != .NET



- ESB is a quick, easy, cost-effective, standards-based way of achieving SOA



Questions?

- Can be reached at
  - ✓ Soa_arch@comcast.net