SOA Best Practices

Jason Bloomberg
ZapThink LLC

zapthink

What is Architecture?

The fundamental organization of a system embodied by its components, their relationships to each other and to the environment and the principles guiding its design and evolution. (IEEE P1471/D5.3)

What is Enterprise Architecture?

- An aggregated architecture of all the individual IT systems within an organization
- The human element within the enterprise
- Systems, people, and organizational constructs at other companies that have relationships with the enterprise
- Individual consumers who are that enterprise's customers

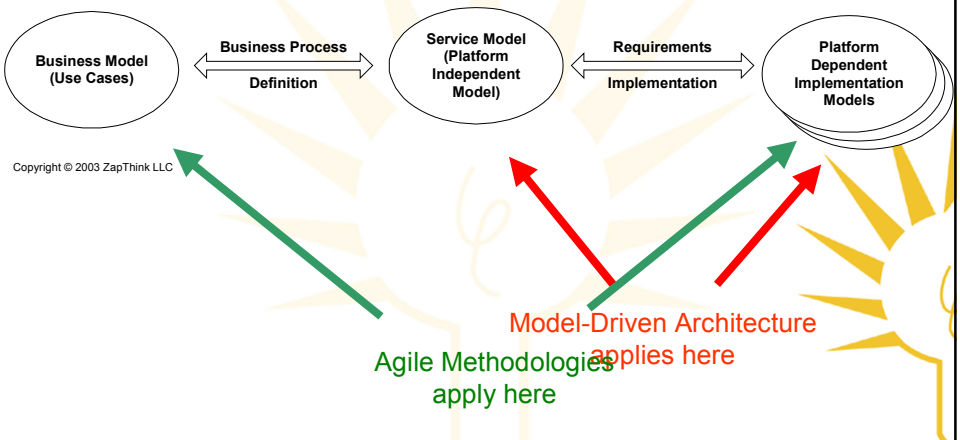
SOA Foundation: Model-Driven Architecture

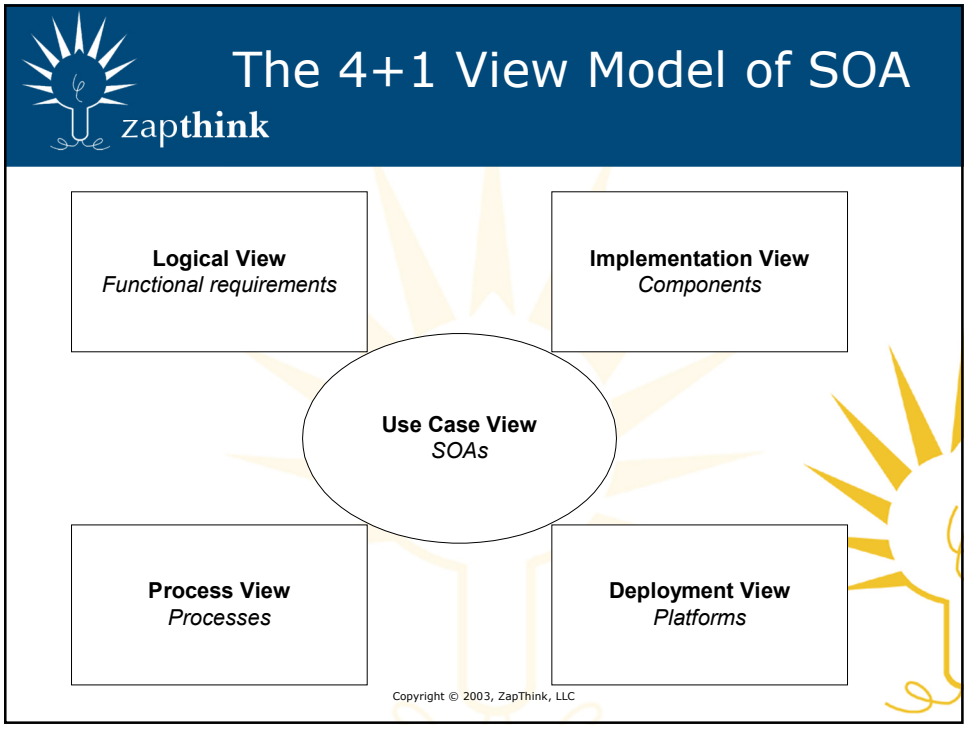
- Object Management Group (OMG) initiative
- Concepts of *models*, *metamodels*, and the Meta-Object Facility, which is a *meta-metamodel*
- Platform independent model and platform dependent models
- Model-driven development
- Weakness: Doesn't (yet) take into account changing requirements

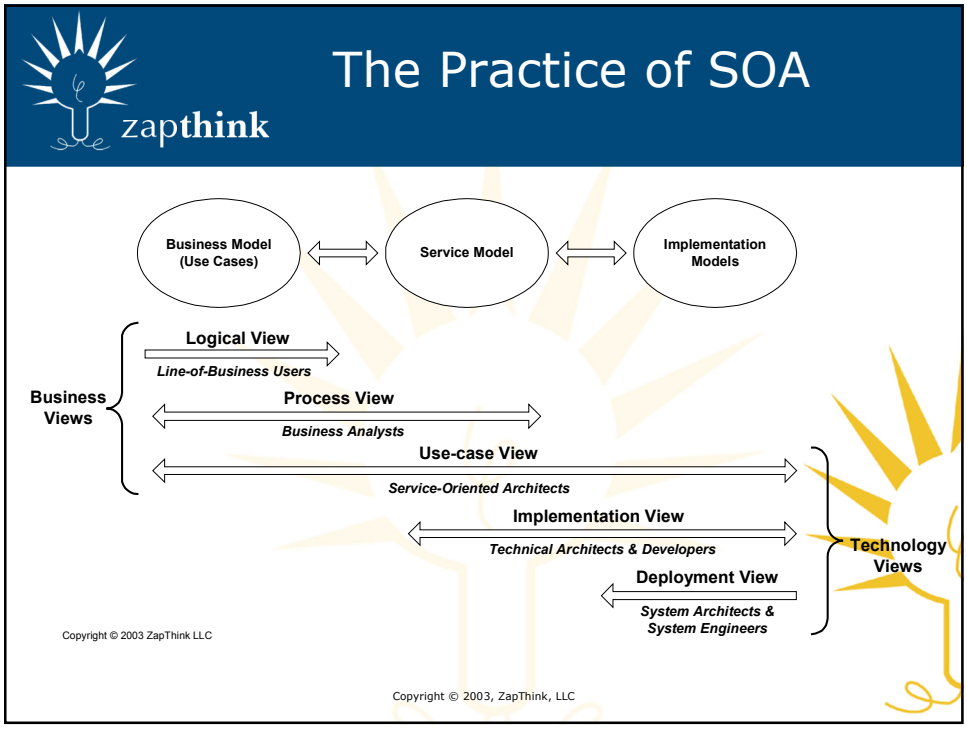
SOA Foundation: Agile Methodologies

- *Extreme Programming* the most notable example
- Applicable in environments of constant change
- Direct end-user involvement
- Test-first, iterative process
- All team members responsible for refactoring
- Focus on people & code, not process & documentation
- Weakness: doesn't scale well

The SOA Metamodel




zapthink
The 4+1 View Model of SOA
Logical View
Functional requirements
Implementation View
Components
Use Case View
SOAs
Process View
Processes
Deployment View
Platforms
Copyright © 2003, ZapThink, LLC


zapthink
The Practice of SOA
Business Model (Use Cases)
Service Model
Implementation Models
Business Views
Logical View
Line-of-Business Users
Process View
Business Analysts
Use-case View
Service-Oriented Architects
Implementation View
Technical Architects & Developers
Deployment View
System Architects & System Engineers
Technology Views
Copyright © 2003 ZapThink LLC
Copyright © 2003, ZapThink, LLC

SOA Best Practices

- These best practices are emerging
- Based on existing distributed computing best practices

BP#1: Develop a top-down, extended enterprise SOA

- A "big picture" roadmap for SOA
- Develop the vision (but not the details) ahead of time
- Think beyond the firewall
- Think about security from day one

BP#2: Build & maintain a platform independent Service model

- Avoid "shelfware" models
- Large projects with clear lifecycles give way to ongoing, *ad hoc* projects
- Service model acts as clearinghouse for information about IT environment
- Requires sophisticated modeling and asset management capabilities

BP#3: Maintain feedback at all points of the architecture

- Feedback must be *active*
- Feedback must be *automatic*
- Between Service model and Implementation models
- Between Business models (Business requirement and Process) and Service model

BP#4: Follow Agile Methodology principles & techniques within the context of the Service model

- Maintain a focus on simplicity
- Operate with small teams focused on speed & efficiency
- Write tests first and work until all tests pass
- Refactor when necessary
- Maintain a focus on users & requirements in context of Service model

BP#5: Encapsulate existing/legacy functionality

- Encapsulate as Web Services with eye toward reuse
- Squeeze more value out of legacy systems
- Find new uses for old data

BP#6: Embrace heterogeneity/follow a federation model of software

- Move from suites to best-of-breed
- Federation: interoperability on abstracted level
- WS-Security example
- Changes approach to package recommendation

BP#7: Compose atomic Services into coarse-grained business Services

- SOA enablement: the "real work" of building & running an SOA
- In domain of Service-Oriented Management and Service-Oriented Integration
- Composition = virtualization

BP#8: Build for consumability/ broad applicability

- "Reuse" typically refers to code reuse
- Refactoring promotes broad applicability
- Asset management promotes consumability

BP#9: Perform *ad hoc* upgrades

- The end of the monolithic, waterfall SDLC
- Constant stream of new requirements crossing the Service model
- Lifecycle management critical
- No "rip & replace"

BP#10: Prioritize SOA transition activities on the fly

1. *Start with "bang for the buck" pilot projects with solid ROI to build expertise and acceptance.*
2. *Once you're up to speed, identify highest value IT problem suitable for SO solution.*
3. *Solve it the SO way.*
4. *Reevaluate business needs to decide on next project.*

- All within context of SOA roadmap
- No fixed timeline
- ROI at every step
- No deadline; may never finish
- Architecture gradually stabilizes, but remains flexible

Final Thoughts…

- SOA is architecture; architecture ≠ integration
- Web Services represent a standards-based approach to integration and architecture
- Getting SOAs right is difficult
- Architecture should be done by architects!

Thank You!

ZapThink is an industry analysis firm focused exclusively on XML, Web Services, and Service-Oriented Architectures.

Jason Bloomberg

jbloomberg@zapthink.com