The Past, Present and Future of Software Architecture

Eoin Woods
UBS Investment Bank

Eoin.Woods@ubs.com
www.eoinwoods.info

About me

- I’m a working software architect
  - Enterprise and software architecture
- Always worked as a software engineer
  - 8 years of products, 7 of applications
- Recently moved to end-user company
  - Stream architect in ETD area
  - Major re-engineering effort
- Co-author of software architecture book
  - With Nick Rozanski, Addison-Wesley, 2005
- Participant in IFIP 2.10 WG
Topics

- Introducing Software Architecture
- The Past
- The Present
- The Future

What Is Software Architecture

The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible qualities of those elements, and the relationships among them

- Bass, Clements, Kazman (SEI)
  Software Architecture in Practice
What is Software Architecture

- The set of design decisions which, if made incorrectly, will cause your project to be cancelled
  - Eoin Woods (heads the SEI definitions list!)

The SEI definitions list:
www.sei.cmu.edu/architecture/definitions.html

Just Design, Surely?

- All architecture is design, not all design is architecture [Paul Clements]
- Not all design decisions are equal
  - Some have "architectural significance"
- Architectural design is outward looking
  - Focus on stakeholder needs, not pure technology
- Architecture more fluid than design
  - Context, scope, success criteria all unclear
Architecturally Significant

- concern, problem, system element; having
- wide impact on structure of the system; or
- wide impact on an important quality property
  (performance, availability, …)

*(Philippe Kruchten, Intro to RUP, 2nd Edition, 2000)*

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Global</th>
<th>Intentional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design</td>
<td>Local</td>
<td>Intentional</td>
</tr>
<tr>
<td>Implementation</td>
<td>Local</td>
<td>Extensional</td>
</tr>
</tbody>
</table>

*“Architecture, Design, Implementation”,
Amnon Eden And Rick Kazman, ICSE 2003*

Is It Really “Architecture”?

<table>
<thead>
<tr>
<th></th>
<th>Civil Architect</th>
<th>Software Architect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>Arts based</td>
<td>Engineering based</td>
</tr>
<tr>
<td>Focus</td>
<td>Aesthetics, usability</td>
<td>How it works</td>
</tr>
<tr>
<td>Key Responsibility</td>
<td>Vision, coordination</td>
<td>Delivery of solution</td>
</tr>
<tr>
<td>Key Talent</td>
<td>Feel for client need</td>
<td>Technical design</td>
</tr>
</tbody>
</table>

- But some similarities, both:
  - need a “big picture” focus and awareness
  - act as a bridge from customer to builder
- Software architect = civil architect + structural engineer?
- Don’t push the analogies too far!

The Essence

- Software architecture is concerned with:
  - Stakeholders
  - System-Level Structures
  - System Qualities

- Software architecture involves:
  - Understanding domains, problems and solutions
  - Making design decisions & tradeoffs
  - Delivering working systems
Architecture in Context

The crucial bridge between requirements and design
“Three Peaks” model – after Bashar Nuseibeh

Why Architecture is Important

- Stakeholder focus
  - who cares and why
  - more than just users and sponsors
- Ensuring that the right system is built
  - critical functions, crucial qualities
- System-wide (or cross-system) consistency
  - what should be constant, what should vary
- Early identification of risk
  - what is going to go wrong and how to avoid it
Types of IT Architect

Enterprise Architects

- Software Architect
- Infrastructure Architect

Operations Architects

- Software Architect
- Infrastructure Architect

Single System Focus
Technology Specific Focus
Production Environment

Cross-System Decisions

Topics

- Introducing Software Architecture
- The Past
- The Present
- The Future
Where Did it Come From?

- Dates from mid-1980
  - roots as far back as late 1960s
  - mainstream interest from mid-1990s
- Began largely as an academic interest
  - from studying how systems were built
- Enthusiastically transferred into industry
  - little interchange between the two since!
- Mirrors the rise in importance and status of the technical IT professional

The Past - People

- 1985 – David Parnas and modules
  - 1987 – Zachman Framework
- 1992 – Dwayne Perry and Alex Wolf
- 1994 - Witt/Baker/Merritt book (IBM FSD)
- 1995 – Philippe Kruchten and 4+1
- 1996 – Shaw and Garlan’s book
- 2000 – Team produce IEEE 1471
- 2003 – Martin Fowler admits it exists!
The Past - Process

- Constant arguments over “what is architecture”
- No shared understanding of what to do & when
- Pioneering architects ploughed ahead
  - Brooks (of course) – S/360
  - Witt/Baker/Merritt – IBM FSD
  - Kruchten - CATS
  - Cutler – Windows NT
  - Booch – methods and analysis
  - Shaw/Garlan/Bass/Kazman/Clements/Obink/Ran/Muller/… – research and study

The Past - Technology

- Some hopeful technologies never took hold
  - ADLs
- Formalisms have been and gone
  - numerous architectural calculi
- Architects largely imposed architecture on resisting technology
  - components of COBOL!
- Little or no architectural thinking in many product lifecycles
  - ad-hoc structure and extension
  - little support for architectural practice
The Past - Politics

- Constantly answering the question “what is a software architect and why do I need one?”
- Early architects often had no recognition of role
  - constant explanation and education
- Little understanding of the benefits in most organisations
  - and conversely, limitations

Topics

- Introducing Software Architecture
- The Past
  - The Present
- The Future
The Present - People

- Today architects have some industry support
  - Organisations like IASA, conferences like WICSA
- Organisations are seeing value
  - IBM, Microsoft, Hartford Financial, UBS, BP, …
- Some technology independent training
  - SEI, Open Group, various degree modules
- “Architect” appears in career frameworks
- But no clear route to architect jobs
  - Software developers, BAs, PMs, hybrid paths
  - Reflects confusion over nature of the role

The Present - Process

- Small number of reliable techniques
  - Viewpoints & views, ATAM, perspectives
- Focus on simple techniques to organise
  - Boxes and lines in PPT do tend to dominate!
  - Little real analysis of descriptions is possible
- Overall process is quite ad-hoc
- Little or no domain focus in approaches
- Reasonably large set of (basic) books
- Some standard terminology exists (e.g. 1471)
  - but not widely used
ATAM

**Phase 1**
- Understand Business Drivers
- Understand the Architecture
- Identify Architectural Approaches
- Generate Quality Attribute Trees
- Analyse Approaches

**Phase 2**
- Brainstorm / Prioritise Scenarios
- Analyse Approaches
- Present Results

**Stakeholder-centric activities**

**Architect-centric activities**

Architecture Tradeoff Analysis Method
http://www.sei.cmu.edu/ata

Viewpoints

- Functional Viewpoint
- Information Viewpoint
- Concurrency Viewpoint
- Development Viewpoint
- Deployment Viewpoint
- Operational Viewpoint

[Rozanski & Woods, 2005]
Viewpoints Example

- A statistics management system
  - Data bulk-loaded into the database
  - Derived measures calculated automatically
  - Statisticians view and report on the data
  - Deductions recorded and reviewed manually

Viewpoints Example

- Described through 5 views
  - Functional
  - Information
  - Concurrency
  - Development
  - Deployment
  - *(Operational view omitted)*
Concurrency View

Development View

Domain
- StatDate Library
- Java Numerical Toolkit

Utility
- Apache Axis
- Hibernate 2.1

Platform
- Java 1.4 Library
- Oracle JDBC Driver 9.0
- Servlet 2.2 API
Deployment View

**Client PC**
- Memory >= 500MB
- CPU >= 1.8GHz

**Primary Server**
- Model: DellSC430
- Memory: 8GB
- CPU: 2x 3GHz

- **Stats_Client**
- **Calculator**

**Database Server**
- Model: SunFire V440
- Memory: 16GB
- CPU: 2x 1.6GHz
- IO: FiberChannel

**Disk Array**
- Model: StorEdge 3510FC
- Capacity: 1000GB

**Data Centre Resident**
- **Primary Server**
- **Database Server**
- **Disk Array**

**Viewpoints and Perspectives**

- Security Perspective
- Accessibility Perspective
- Performance Perspective
- Location Perspective
- Availability Perspective
- Regulation Perspective
- Maintenance Perspective
- etc.

**Stakeholders**
- Architecture
- Development View
- Information View
- Operational View
- Deployment View
- Information View

**Architecture**
Perspectives Example: Security

- The architecture we’ve described is credible
- What would happen if the system needed to protect the system’s information?
  - Justice community system for criminal intelligence

Considering Security

- Sensitive Resources
  - The data in the database
- Security Threats
  - Operators stealing backups
  - Administrators querying data, seeing names
  - Bribing investigating officers
  - Internal attack on the database via network
Considering Security

- Security Countermeasures
  - Backups: encrypt data in the database
    - How about performance?
    - Does this make availability (DR) harder?
  - Seeing names: use codes instead of names, protect codes at higher security level
    - More development complexity
    - Possible performance impact

- Network Attacks: firewalls, IDS
  - More cost
  - More deployment / administration complexity
  - Operational impact if IDS trips

- Bribery: add audit trail for data access
  - Possible performance impact
  - More complexity
  - Protecting / using the audit trail
Considering Security

Information View Impact

- DerivedMeasure
  - Deduction
  - StatsSet
  - Variable

Isolate names

Identifier Code

Observation

Add audit when accessing data

Considering Security

Development View Impact

- Domain
  - Controlled StatAccess Library
  - StatDate Library
  - Java Numerical Toolkit

- Utility
  - Apache Axis
  - Hibernate 2.1
Considering Security

- Deployment View Impact
  - Added network model making network security clear

- Other Impact
  - Need IDS added to Development view
  - Need to capture impact on Operational view
  - Need to consider impact on availability
  - Need to re-work performance models to allow for database encryption, audit, …

- Note the need to change many views
- This is “architecturally significant”
The Present - Technology

- Today it’s a design time game
  - Most architecture lost as we move to code
  - Even with so called MDA – the “A” doesn’t survive

- Some styles codified in technologies
  - Grid, P2P, client/server, tuple space

- First-class connectors in some places
  - Messaging oriented systems, ESBs, …

- First tools for architects appearing
  - Lattix, Sotograph, Aspects, Troux’s Metix (for EA)

The Present - Politics

- Everyone’s an architect
  - 10 years technical experience => “architect”

- Language of software architecture widely used
  - Even if rarely defined or understood

- Products sold through architecture
  - .NET, J2EE, SOA, distributed caches
  - Leads to mass confusion about "architecture"

- Agile / architecture tension is fairly high

- Competing professional organisations
  - IASA, WWISA, GEAO, AEA, …
Topics
- Introducing Software Architecture
- The Past
- The Present
- The Future

The Future - People
- Better defined role
  - Architect vs. developer, technologist, tester or PM
  - Helps relationships with others
- Career track recognition
  - Agreement of key skills and responsibilities
  - Possibly certification (e.g. Microsoft, IASA, GEAO)
- One or more key professional bodies emerge
  - Probably IASA, perhaps others
- Education
  - MSc in software architecture perhaps?
The Future - Process

- Standardised practice
  - domain specific approaches
  - real time vs. transactional vs. data centric vs. …
- Different levels of architectural process in use
  - from “agile” to “SEI” (or “kennel” to “skyscraper”)
- Design decisions will become first class
  - move focus from *structure* to *rationale*
- Fundamental agreed definitions
  - need to decide if necessary or even desirable?

The Future - Technology

- Moving architecture to runtime
  - Views in the running system
- Better description languages & tools
  - Executable and queryable architecture description
- Architecture in the running system
  - ADLs / DSLs / Aspects
- Architect-specific tool support
  - Lattix and Sotograph are just early examples
  - Modelling and construction ripe for development
The Future - Politics

- Someone will win the hearts and minds
  - IASA?
- Selling to architects will get more intense and effective
- The agile / architecture tension will settle down
  - Both will realise where their strengths are
- More research / practice alignment?
  - WICSA6 and WICSA7 perhaps!
Topics

- Introducing Software Architecture
- The Past
- The Present
- The Future
- Summary

Summary (i)

- Software architecture is still young
  - really a product of 1995 – 2005
- Mainstream since about 2002
- Good core of knowledge emerging
  - approaches and techniques
- Some agreement on fundamentals
  - stakeholders, structures, qualities
  - understanding, designing, trading-off, delivering
- Much to do to raise level of sophistication
  - description, analysis, runtime representation
Summary (ii)

- Finally research & practice meeting
  - WICSA 5, IASA, Microsoft, …
- Better design time tools will come
  - describe and analyse architectures
  - create architectures using dedicated languages
  - beyond boxes and lines
- The future is architecture in the running system
  - too much is lost today
  - architecture description as an executable deliverable

To Learn More

*Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*

Nick Rozanski & Eoin Woods
Addison Wesley, 2005

http://www.viewpoints-and-perspectives.info