A Study of Architectural Decision Practices

Thomas D. LaToza, Evelina Shabani, André van der Hoek
University of California, Irvine

What are the social dynamics of architectural decision making in practice?

Method

11 semi-structured **interviews** (26 - 44 mins)
- 5 developers at small **health** information technology company
- 6 developers at small **telecom** startup

focused on architecture, important decisions, knowledge sharing practices, code reviews

data analysis: **affinity diagramming** of transcripts

Results

Making Architectural Decisions

"Everyone gets involved, it's not just one person making the decision."
- meetings build **consensus**, but architecture is primarily a product of the senior developer

"I think by choosing something like [Apache] Wicket, it kind of ***enforces*** *a pattern on you."*
- **technology** decisions were reported to be the key architectural decisions
- technology decisions imposed architectural styles
- process driven less by requirements and more by a range of **factors**

Factors developers reported considering in making technology choices

Factor	Example
Scalability	“It is easier to **scale** Tomcat out vertically than JBoss.”
Extensibility	“It is easier to **plugin** open source tools”
Popularity	NoSQL databases are the “**hot** thing”.
Personal bias	**Preference** to put logic in the database
Corporate bias	Corporate requirement for **in-house** frameworks
API usability	SQL provides more **abstraction** than NoSQL
Learnability	Preference for middleware that looks **learnable**
Expected longevity	Preference for technologies that **endure**
Reduce coupling	JSON supports optional params that can be **ignored**
Simplicity	J2EE “**bloated**” because much of it is not needed.
Deployment	Operations experience supporting MySQL deployment

Communicating Architectural Decisions

"Only 10% of the design decisions and constraints made it to the Wiki, because ***who has time*** *to write into the wiki"*
- time significant barrier
- rapidly changing code made explanations out of date
- felt that small teams made **verbal** communication particularly important

Code reviews ensured **compliance** and **communicated** architecture to new developers
- important when past habits conflicted
- e.g., batch oriented styles rather than project's event-driven style

Revisiting Architectural Decisions

Developers learned technologies **after** adopting them
- many architectural decisions revisited through two complete rewrites of codebase
- discovered **Achilles heel** of technologies - use case it could not support

Technologies and patterns developers reported revisiting

Technology or Pattern	Achilles Heel
J2EE version 1	Entities stored as a database row are stored as a CORBA object, which has much **unnecessary** data
SQL databases	Cannot **scale** to billions of rows
Annotation-based AOP	Cannot insert calls in all cases
Unnormalized database	Schema changes require changes to all **consumers**
In-memory state persistence	When deployment node goes down, state lost

Design Implications

Architecture driven by **technology** decisions
- impose important constraints
- have Achilles heels
- **revisited** at great cost

existing resources help **learn** technology, not evaluate adoption (books, forums, tutorials, Q&A sites)
technology websites makes case **for**

What if developers could make a technology **adoption** decision by visiting a **website**?
- browse **competing** technologies
- compare **ratings** of adoption factors
- see Achilles heels
- learn potential workarounds
- read technology developer responses
- **report** their own experiences

SDCL Software Design and Collaboration Laboratory

UCIRVINE