Architectural Refactoring
PLURALSIGHT

Dave Adsit

Architect

@davidadsit
codeobsession.blogspot.com
Architectural Evolution at

PLURALSIGHT

a mostly true story
What is Software Architecture?

Architecture is the stuff that’s hard to change.

Martin Fowler
What is Refactoring?

Refactoring is a changing the structure of code without changing its behavior.
What is Architectural Refactoring?

Intentionally changing the structure of a system without altering its features.
Why would anyone do that?

The technology fails to meet the needs of users.

The team fails to deliver on the needs of the organization.
Architectural Drivers

Quality Attributes

- Performance
- Compatibility
- Reliability
- Scalability
- Extensibility
- Availability
- Maintainability
- Usability
Architectural Drivers

Constraints

Money
Hosting
Environment
Experience
Skills
Knowledge
Capacity
Patterns of change

A generally reusable solution to a commonly occurring problem within a given context
Build a monolith

Fast and easy
Can be done with a small team
Allows for quick changes
No need for QA handoff
No need for Ops handoff
Leverage a distributed web cache (CDN)

Offload traffic from a web server
Improve response time for users far away
Cluster the servers

Distribute load
Improve reliability
Move from physical to virtual machines
Eliminate session state

No more sessions pinned to servers
Replace session data with user data
Replace feature branches with feature toggles

Integrate continuously
Decouple deploy from release
Ability to disable features that don’t work out
Separate data and static assets

Reduce server-side rendering load
Leverage client processing power
Improve quantity and granularity of caching
Shard the database

Partitioning data to distribute load
Introduce a polyglot datastore

Document database for fast reads
SQL database for ad hoc reporting and research
Miscellaneous special purpose databases
Replace database queue(s) with real queue(s)

Don’t implement a queue in a database
Guaranteed at least once delivery
Hire more developers
Organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations.

Melvin Conway
Embrace stack diversity

.NET
NodeJS
Python
PHP
Ruby
Redundant data centers

Increase reliability
Simplify disaster recovery
Reduce response time
Strangle the monolith

Repeat extract context until no more than a single context remains behind.
Distributed system of record

One bounded context is the system of record
Others have local caches of relevant information used to make local decisions
Implement async communication pattern

Publish/Subscribe
Leverage the queue for reliable delivery
Implement blocking communication pattern

Request/Response
RESTful APIs
Circuit breaker
Orchestration and choreography

Conductors direct orchestras
Dancers in a ballet act independently
Trade total complexity for local simplicity
How long will it take?
Questions?

Dave Adsit

Architect @ Pluralsight

@davidadsit

codeobsession.blogspot.com