



Just Enough Early Architecture to Guide Development

Jon Kern

Agile Something-or-Other

TechnicalDebt.WetPaint.com

JonKern@comcast.net



INTRODUCTION

- Doing software since 80s, O-O since ~late 80s
- Aerospace engineer (5 yrs jet engine R&D, 10 years DoD consulting, flight simulation, centrifuge, fighter agility, etc.)
- Formed Lightship Inc in '95, clients like IBM, Ingersol...
- First published agile method in '97
- Co-author Java Design w/ Peter Coad
- Led OO/Java workshops, mentored hundreds
- Joined Peter to form TogetherSoft Sep '99
- Formed the Coad-Certified Mentor group; mentored on OO, FDD
- Led Together Product Development teams in Russia
- Co-author Agile Manifesto, 2001
- After growing to \$56M and 400+, sold TogetherSoft to Borland in late 2002
- Recruited by OptimalJ (MDA) Team 2003, left late 2006
- Consulting since 2006, mentoring teams on architecture, development process, and agile methods to build business solutions

What's on Tap

- **Agenda**
 - Simple process framework for tackling architecture
 - When & how to tackle architecture design
 - How architecture can serve more than just the application
 - Review project examples, one in great depth over two-years
 - Pitfalls to avoid
- **Goals**
 - Describe what we mean by “Architecture” and its flavors
 - Learn how architecture fits into agile development
 - See how proper architecture can lead to a great many benefits
 - Conversely, learn how poor architecture is damaging
 - Learn how to gauge when “enough is enough”

How Is Software Built?

- **People,**
- **Process, and**
- **Tools!**

- **In about that order...**
 - Good people will trump poor/non-existent process
 - Good people will either create tools or use tools in an effective manner
 - Process and Tools cannot make up for inadequacies of the development staff
 - Gray matter is a pre-requisite

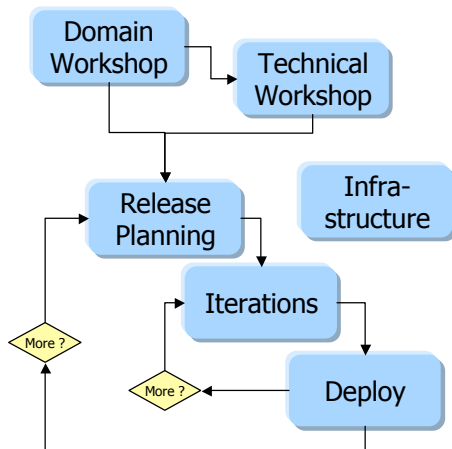


How Do We Start?

- Do we start with Use Cases?
- Do we start with architecture?
- Do we start with domain?
- Just start coding?
- Say
“We’re goin’ agile, step aside!”
and hope no one asks questions?
- Is architecture emergent?
 - It can be... as long as the right amount is “emergent” soon enough, lest you end up with a lot of wasted efforts and excessive refactoring



Development Process Flowchart



- Get enough requirements...
- ...to begin architecture work (if needed)
- Domain + Tech. lead to initial guess at Release Plan
- Architecture & Infrastructure Align
- Set up tasks for tackling architecture
 - Thin slice needed?
- Plan features as the business needs, however, often need to factor architecture into early stages
- “Release” to gain feedback

Requirements Workshop

- **The stakeholders, domain experts, clients describe the business problems they want to solve**
 - We get an idea of the type of application needed
 - We learn about non-functional drivers
- **As features are uncovered by visiting the breadth of the domain...**
 - Features are added to the feature list
 - Elements of the domain are added to a working model
 - Complicated features may require further exploration
 - Uncomplicated feature “depth” can often be skipped over
- **Typically 10% of the project time. Maybe a week or two is sufficient for medium-sized efforts**
- **You are done when you are staring at each other, itching to get back to your own work!**

Architectural Workshop

- **Begin with identifying the application type**
- **Incorporate the non-functional drivers (“-ilities”) into the design**
 - Performance
 - Volume
 - Concurrency
- **Use standard architectural styles, or develop a first-cut architecture if a new one is required**
- **Be sure the technology fits (if need be, prove it)**
- **If needed, do one or more architecture “spikes” (and prove it)**
- **Create a thin slice to disseminate the patterns and coding guidelines to the team**
- **Set clear & measurable goals, make frequent demonstrations, don’t make it analysis paralysis or an academic exercise**

Some Simple Guidelines

- **It's all about balance**
- **Never let one area of activity get too far ahead of others**
- **You don't need *all* requirements prior to defining architecture -- just enough of the right ones**
- **Start building as soon as you have a good enough plan**
- **Many steps will be revisited, so don't get carried away**
 - If possible, consider approaches that are able to be changed
- **You can't build technical architecture without knowing enough of the the overall business needs**
 - Current project
 - Possible future direction
 - Business impetus and drivers
- **Assume things with caution, test assumptions assiduously, chant: "Show Me," think continuously**

Let's Talk About Architecture...

- **What is it?**
- **Why do we do it?**
- **Where does it fit in?**
- **When do we consider it?**
- **Who should do it?**
- **Who pays for it?**
- **Who benefits from it?**

What is Architecture? The Big View

- **Many viewpoints**
 - Just ask anyone on the extended project team!
 - Major themes behind how the system will be built and how it will work
 - Application architectural is not the same as the technical architecture which is not the same as the underlying component architecture
 - Macro architecture and micro architecture
- **Form should follow function**
 - Product needs to have the right features
 - Application needs to be on the right deployment platform
 - Design needs to make it all happen as effortlessly as possible
- **Dealing properly with architecture yields many benefits**
 - Coding is made easier
 - Clarity of purpose is obvious
 - Extension to support new features is easier
 - Various illities/quality attributes are easier to meet
 - Minimal amount of complexity is allowed into the solution

(Is it) Architecture or Design?


- **Chocolate or vanilla?**
- **It is a bit blurry at times, but...**
 - An architecture is more of a type of approach to a class of problems
 - A design is more of how we seek to implement the architecture
- **Macro architecture is a matter of style**
 - a big, Deployment Diagram-type architecture; e.g., N-tier, Agent-based
 - To implement a narrow, "thin-slice" running system, requires we design out the techniques in code
 - This design is typically applicable across the breadth of similar features
- **Micro architecture is normally for smaller bits**
 - Messaging, authentication, logging, persistence
- **The User Interface Experience can be designed from a visual standpoint, considering style, etc.**
 - But the UI needs to be micro-architected to provide a mechanism to allow the UI to interact with the rest of the system
 - The UI architecture is regardless of the specific content
 - And the UI can be designed from an implementation perspective

Why Should Architecture Delay Coding?

- **For users, as long the app works, it doesn't matter how it is built... (at least not for a while)**
- **But as professionals, we need to:**
 - Ensure that more than just the functional requirements are met
 - Ensure proper quality exists (life support application more critical than a twitter plug-in)
 - See that the business interests are met
- **Leaning Tower of Pisa might be ok**
 - ...if you just rebuild it every 2 years
- **If the app needs to be around for 15 years**
 - Then you better pay close attention to detail!
 - Or you may choose to rebuild frequently the first few years
- **Understanding the impact that architectural decisions have on the business goals is our job**
 - Sometimes we shouldn't care that much about architecture! (for a simple, throw-away demo app for a trade show)

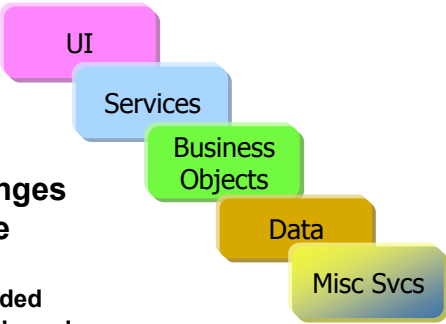
What Happens with Poor Architecture?


- **In Development**
 - Everything is more difficult
 - redundant and repeated code, dead code
 - lousy documentation
 - confusing semantics
 - convoluted workflow
 - Inconsistent architecture/code in various areas
 - Supporting techniques for ensuring quality are more of a challenge
 - Automation, unit testing, regression testing, etc.
- **In Production**
 - Unstable system
 - Some desired business functionality not supported
 - It takes 80% of the effort just to "keep the lights on"
 - App seems "brittle" – hard to change without causing breakage (low modularity/cohesion, high coupling)
 - It is hard to add new features
 - No one can really point to your business model
 - Performance can suffer over time (or immediately)

TheServerSide
JAVA SYMPOSIUM 

Where does Architecture Fit in?

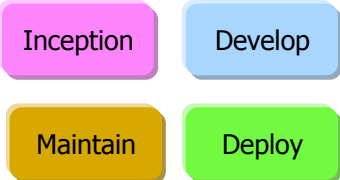
- **Architecture applies (recursively) at many levels,**
 - From enterprise, to...
 - Product portfolio
 - Coding
 - Quality assurance
 - Supporting tools
 - Infrastructure
- **This is one of the challenges we face with architecture**
 - Later, we will see ways to deal with developing the needed architecture to meet the user's and system's business requirements



TheServerSide
JAVA SYMPOSIUM 

When Does Architecture Get Done?

- **It starts from the very first utterance of an idea for an application -- and it never really completely ends**
- **Architecture & agile techniques should pervade the entire SDLC**
 - But there are points in the project where you (should) pay more attention
- **Since architecture can make or break some applications, this "risk" should not be ignored**
- **Architecture should be tackled**
 - As a macro decision from the start
 - Details for a thin slice, to provide code patterns
 - To guide infrastructure
 - When considering testing
 - As part of fulfilling cross-cutting concerns



Who Does the Architecture?

- **Varies across companies and teams...**
 - Some have official architecture groups that create designs, but may or may not actually create implementations for a given project
 - Others simply have senior, wiser, gifted architects that are also developers
- **Might be multiple folks that specialize in different areas**
 - UI
 - Server
 - Persistence
 - Specialties (SemWeb, SOA, etc.)

How can we Tackle Architecture in an Agile Way?

What is one of the foundational architectural principles?

Separation of Concerns

- **A great way to drive much of the development “thought”**
- **An architectural best practice, the “father” of many patterns**
- **Keep business logic in one place/layer**
- **Architect with the future in mind, by capturing the Domain of the business**
- **Keep technology at arm’s length and “in its place”**
- **Isolate change to the smallest place(s)**
- **Treat different concerns differently!**
 - User Interface
 - Business Logic
 - Persistence
 - External systems/services
 - Logging/Security
 - Performance
 - Testing
 - Etc.

Architecture is More than “Meets the Eye”

- **Architecture is a key to agile projects on many fronts**
- **“Thin Slice Architecture” is an imperative to doing any new technical architecture**
 - A “Reference Implementation” serves a guideline for all
- **Allows for ease of refactoring, accepting change**
- **Performance test harness on top of architecture; e.g.**
 - Individual method timing
 - Load testing
 - Tools to publish & analyze test results, and to highlight bad trends
- **Automating builds, tests, performance tests, notifications of exceptions, publishing results**
- **Build a culture of checks and balances**

The Two Primary Architecture “Concerns”

- **Two of the biggest “drivers” for the project are:**
 - Application architecture/design
 - Technical architecture
- **The basic flavor of the application will be based on the requirements**
 - RIA/web client? Desktop app? Mobile?
 - Disconnected? Distributed? SOA? Object-oriented?
 - Critical banking app?
 - Real-time factory automation?
 - Role-playing game

Multi-dimensional Architecture & Design

- **Need to know enough about the requirements ...**
 - Rough ideas on the performance, scalability, etc.
 - Sketch out the basic UI ideas that might work or are desired
 - What sorts of integration we might face
 - Any “corporate” aspects to consider
- **To consider the various “concerns” that might be impacted the most:**
 - Lifetime of the product
 - Likelihood of enhancements
 - Types of users and amount of usage
 - Complexity of domain
 - Complexity of workflow
 - Possible solution archetypes
 - System integration challenges
 - Ability to test

Architecture & Orthogonality

- **Application Architecture**
 - All about the users -- not about the technology
 - Represents client-valued features
 - Organized to meet the needs and workflow of the user
 - Includes “-ilities”
 - Features are bundled into major components
 - Frequent changes:
 - Brand new features
 - Feature enhancements
 - Bug fixes
 - Reuse standard components and User eXperience techniques
 - Helps drive decisions regarding the technical architecture!
- **Technical Architecture**
 - Not usually “paid” for directly
 - Needs to meet the assorted non-functional/cross-cutting concerns
 - Performance
 - Security
 - Logging
 - Can be based on standard architectures, or could be custom
 - Should be applied consistently
 - Across all similar parts of the application
 - Should have flexibility to change over time, though tends to change more slowly than features
 - Able to support the features!

TheServerSide
JAVA SYMPOSIUM

Consistent Construction - Putting it Together

Application Arch. + **Technical Arch.** + **Construction Guidelines** = **Working App!**

Annotations in Technical Arch. include: **SOA, Distr., Web App** and **Patterns, Conventions**.

Annotation in Application Arch.: **Features**

Annotation in Construction Guidelines: **Working App!**

ID	PROJECT	TITLE	DATE	STATUS	USER
12101001	ORION	Project	23-Apr	Accepted	John
12101002	ORION	Test	23-Apr	Done	John
12101003	ORION	Test	23-Apr	Done	John
12101004	ORION	Test	23-Apr	Done	John
12101005	ORION	Test	23-Apr	Done	John
12101006	ORION	Test	23-Apr	Done	John
12101007	ORION	Test	23-Apr	Done	John
12101008	ORION	Test	23-Apr	Done	John
12101009	ORION	Test	23-Apr	Done	John
12101010	ORION	Test	23-Apr	Done	John
12101011	ORION	Test	23-Apr	Done	John
12101012	ORION	Test	23-Apr	Done	John
12101013	ORION	Test	23-Apr	Done	John
12101014	ORION	Test	23-Apr	Done	John
12101015	ORION	Test	23-Apr	Done	John
12101016	ORION	Test	23-Apr	Done	John
12101017	ORION	Test	23-Apr	Done	John
12101018	ORION	Test	23-Apr	Done	John
12101019	ORION	Test	23-Apr	Done	John
12101020	ORION	Test	23-Apr	Done	John
12101021	ORION	Test	23-Apr	Done	John
12101022	ORION	Test	23-Apr	Done	John
12101023	ORION	Test	23-Apr	Done	John
12101024	ORION	Test	23-Apr	Done	John
12101025	ORION	Test	23-Apr	Done	John
12101026	ORION	Test	23-Apr	Done	John
12101027	ORION	Test	23-Apr	Done	John
12101028	ORION	Test	23-Apr	Done	John
12101029	ORION	Test	23-Apr	Done	John
12101030	ORION	Test	23-Apr	Done	John
12101031	ORION	Test	23-Apr	Done	John
12101032	ORION	Test	23-Apr	Done	John
12101033	ORION	Test	23-Apr	Done	John
12101034	ORION	Test	23-Apr	Done	John
12101035	ORION	Test	23-Apr	Done	John
12101036	ORION	Test	23-Apr	Done	John
12101037	ORION	Test	23-Apr	Done	John
12101038	ORION	Test	23-Apr	Done	John
12101039	ORION	Test	23-Apr	Done	John
12101040	ORION	Test	23-Apr	Done	John
12101041	ORION	Test	23-Apr	Done	John
12101042	ORION	Test	23-Apr	Done	John
12101043	ORION	Test	23-Apr	Done	John
12101044	ORION	Test	23-Apr	Done	John
12101045	ORION	Test	23-Apr	Done	John
12101046	ORION	Test	23-Apr	Done	John
12101047	ORION	Test	23-Apr	Done	John
12101048	ORION	Test	23-Apr	Done	John
12101049	ORION	Test	23-Apr	Done	John
12101050	ORION	Test	23-Apr	Done	John

TheServerSide
JAVA SYMPOSIUM

Examples

- **TogetherSoft UML Modeler**
 - Automated testing, even for model diagrams
 - Performance trend tracking
- **Desktop Photo organizing and sharing app**
 - Headless testing bot
- **RIA Insurance product sales portal**
 - Automated regression test & scenario coverage analysis
 - Import actual policies as test cases
 - Manually create quotes, save as test cases
 - Policy entry simulator
 - Migrator & QCreator

TogetherSoft UML Tool

- **Java-based**
- **UML Modeling tool**
- **Performance tracking was key to catching any degradation of the user experience**
- **New platform was architected**
- **Show details...**

Tidepool Photo Organizer


- **Java desktop**
- **JEE Server component**
- **Developed “headless” client-side test “robot,” able to simulate multiple users to put load on server and test core functionality**
- **Show details...**

RIA Sales Portal - A Walkthrough

- **Initial Requirements**
 - Agent Sales Tool
 - Create insurance quotes
 - Live on top of System of Record (SOR)
 - Capture and use additional business process and logic
- **Results**
 - Application Style: RIA
 - Architectural Style: Layered Approach
 - A constellation of integrated tools
- **See Details...**
 - Evolution of architecture (application, technical, tools, UI)
 - How choices led to benefits for design refactoring, tool creation, growing over time to meet new demands

The Balancing Act

- **There is no trick... but you might feel at times like you are spinning plates in a circus**
- **At many points along the way, be sure to carefully consider risks**
 - Have we done enough, or not?
 - Do we think there is a looming gotcha?
 - Can we do more to learn more?
 - Can we gauge return on effort?
 - Can we suffer the worst outcome if we missed something?
 - If not, what do we do about it?
 - Can we mitigate risks early and often?
- **Architecture can make or break the project**

TheServerSide
JAVA SYMPOSIUM 

Questions

TheServerSide
JAVA SYMPOSIUM 

Thank You!

- **Contact me at jonkern@comcast.net**
- **Visit <http://TechnicalDebt.wetpaint.com> for latest slides**