Agile and Architecture

The Same, Then and Now

May 18th, 2010

Per Lundholm

crisp

Contents

- Architecture
- Project vs Lean
- How to do it
- What to read

Who am I?

- Per Lundholm, Crisp AB
- 30 years of professional programming
- Consulting as Architect, Architect-coach, Lead Developer
- Teacher at Astrakan
- Crisp is an employee owned company known for agile courses with internationally renowned teachers and experienced, agile developers

Architecture

- **A system's architecture determines its qualities**
  - Scalability
  - Performance
  - Testability
  - Cost
  - ...
- **So architecture is important, even if you're agile**

Architecture is About

- **It is how things relate to each other, the system's structures**
  - A chair and a table
    - In a room
      - Rooms in a house
        - Houses in a block connected by roads
        - Blocks in a city region
        - ...
- **The functions of these are important but architecture is about qualities that emerge from the structures**

Architecture is Understood ...

- **All systems have an architecture**
  - Understood or not
- **We understand architecture through several views**
  - The deployment view
  - The logical view
- **Like different plans for house**
  - Electricity
  - Exterior

Traditional Projects vs Agile

Start-Deliver-End One architecture	**Deliver continuously** Changing architecture
Follow plan Changes are expensive	**Deliver customer value** Changing requirements
Architect Role Given mandate	**Team member** Meritocracy

How to do it – state of mind

- **Embrace Change**
  - You're supposed to be *agile*!
- **Accept Darkness**
  - You will know more later on
- **Be the Guru**
  - Be pro-actively helpful
  - Be loud and clear
- **Expect to Learn**
  - Everybody should

How to do it - architecture qualities

- **Change architecture through refactoring**
  - A super-flexible architecture is *not* an answer to changing requirements
  - Refactoring needs automated tests at system level
    - Testability gets high priority
- **Deliver often**
  - The process of deployment should be cheap and fast
    - You pay each round – find the impediments and fight them

Automated Tests

- **At the system level – three kinds of suites**
  - Tests that marks progress
    - The suite that fails until done
  - Tests for functional regression
    - The suite that never fails – or stop!
  - Tests for quality regression (e.g. performance)
    - The suite that report numbers.
      - You may fail on threshold values

How to do it - strategy

- **Go from end to end process-wise**
  - Build the system
  - Run the automated test
  - Deploy
- **Go from end to end system-wise**
  - Build a skeleton
    - Include as many tiers as possible
    - With automated tests

How to do it - documentation

- **It is better if the system's architecture is known to all**
  - But it changes, so how do we keep up?
  - Use the walls
    - In our eyes all the time
  - Use pair-programming
    - Spreads the word
  - Use a wiki
    - For all the details

Technical Debt

(Courtesy of Henrik Kniberg, Crisp)

- **Analyze defects that slips through**
  - Example: Production server crashes when more than 100 users logged in

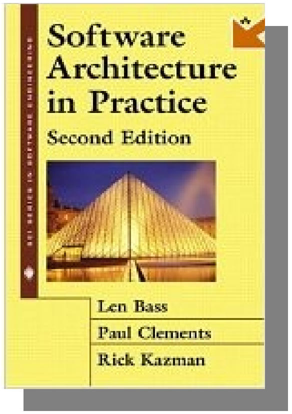
Question	Answer	Improvement
Why did the server crash?	SessionHandler uses a DB connection pool with fixed size 100	Gracefully reject incoming requests when connection pool is full
Why wasn't the bug detected before release?	No automated performance tests	Write automated performance tests
Why weren't the tests written from start?	Nobody knew how to write such tests	Give people more time to learn
Why didn't the team take the time to learn?	Too much pressure on team.	Reduce pressure by reducing scope of project.

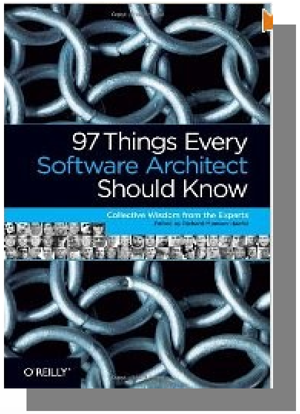
Exercise

- **Pairs: suggest an agile strategy**
  - What to do first and what to do later
- **An online, real-time strategy game**
  - Incomplete information, “fog-of-war”
- **Concerns**
  - Cost (start-up with small funding)
  - Cheating
- **Where is the game-state?**

What to Read

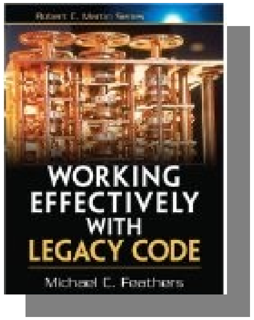
- **Software Architecture in Practice**
  - The principles
  - Some ideas are dated
- **97 things Every Software Architect …**
  - Taken with 97 grains of salt

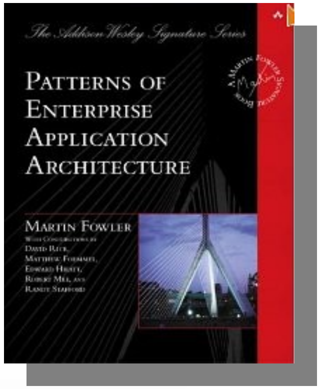



97 Things Every Software Architect Should Know
Collective Wisdom from the Experts
O'REILLY

What to Read

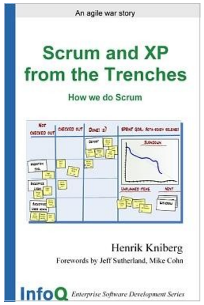
- **Working Effectivley with Legacy Code**
  - Code without tests is legacy code
- **Patterns of Enterprise Application …**
  - If that is what you do




The Addison Wesley Signature Series
PATTERNS OF ENTERPRISE APPLICATION ARCHITECTURE
MARTIN FOWLER

Books from us at Crisp

- **Scrum and XP from the trenches**
  - Practical Scrum
- **Kanban & Scrum, making the most of both**
- **Proritera, Fokusera, Leverera**
  - Executive's introduction to agile



An agile war story

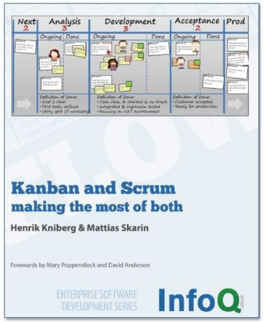
Scrum and XP from the Trenches

How we do Scrum

Henrik Kniberg

Forewords by Jeff Sutherland, Mike Cohn

InfoQ Enterprise Software Development Series

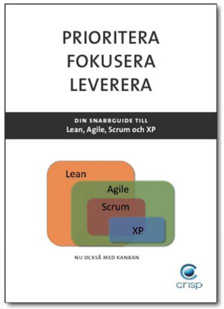


Kanban and Scrum making the most of both

Henrik Kniberg & Mattias Skarin

Forewards by Mary Poppendieck and David Anderson

ENTERPRISE SOFTWARE DEVELOPMENT SERIES InfoQ



PRIORITERA
FOKUSERA
LEVERERA

DIN SNABBGUIDE TILL
Lean, Agile, Scrum och XP

NU OCKSÅ MED KANBAN

crisp

The End

- **Agile means *embracing change***
- **Change the architecture to keep it suitable**
- **Change through refactoring**
- **Refactoring demands automated tests**