valtech_

Mit Kernelementen von Scrum zu einer robusten Architektur

With Core Elements of Scrum to a Robust Architecture

Michael Mai – Entwicklertag 2013

Topics

- Robust architecture
  - What is robust?
- Core elements of Scrum
  - What are these?
  - How are they supporting?
- Employ both for best
  - How?

Who is Michael Mai?

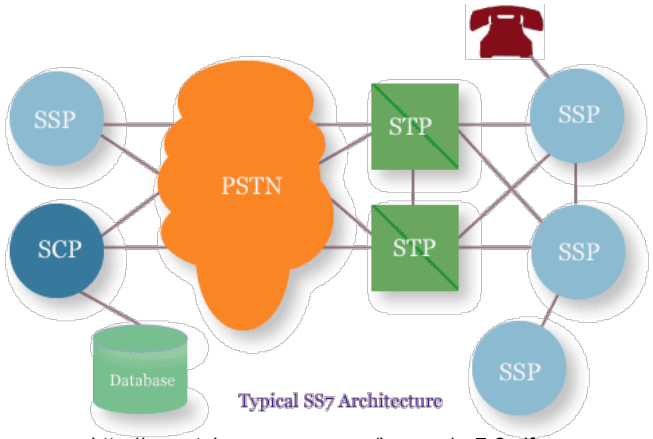
- Senior Consultant
  - At: valtech
  - Division: Agile Consulting
- Before that
  - Scrum Master, Software Architect, Software Developer
  - Scrum Trainer (scrum.org)
- Focus on
  - Business value, Organization, Architecture, People

1. Robust architecture

What is architecture?

→ By man's construction

Signaling System No. 7 (SS7)



http://www.telecomspace.com/images/ss7-2.gif

What is architecture?

→ By nature

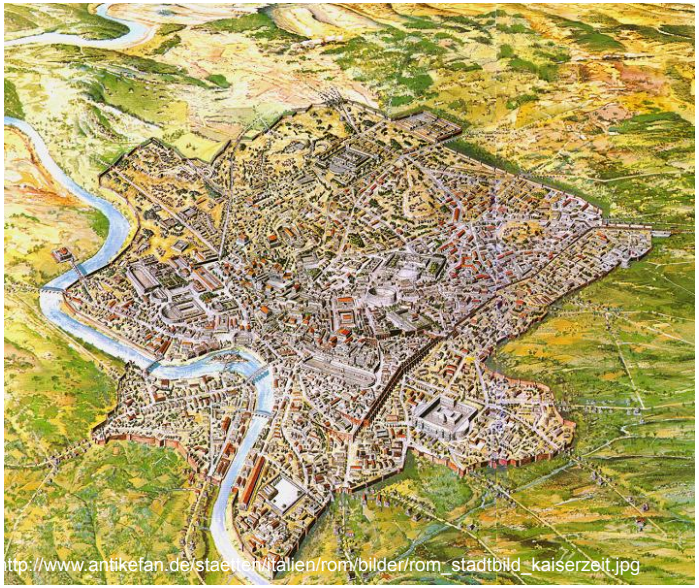
(= evolution over million of years)

http://i.onmeda.de/gehirn_modell.jpg

What is architecture?

→ By adaptive evolution
- Grown
- Decayed
- Destroyed
- Rebuilt
- Modernized

Drivers
- War
- Population
- Time
- Politics



What is architecture?

- Information architecture
  - Which information is where required, in which quality and granularity?
- Data architecture
  - How is data processed and floated through-out the system?
- Code architecture
  - What does the code looks like?
  - Which principles should be obeyed?
  - Which abilities and capabilities does the manifestation have?

What does “robust” mean?

- Think of
  - Does complexity always mean robust?
  - Does robustness always mean complex?
- Think of
  - Does openness always mean not robust?
  - Does robust always mean not open?

> … robustness is the ability of a computer system to cope with errors during execution or the ability of an algorithm to continue to operate despite abnormalities in input, calculations, etc. The harder it is to create an error of any type or form that the computer cannot handle safely the more robust the software is.

http://en.wikipedia.org/wiki/Robustness_(computer_science)

„robust“

- Even thus slight mistakes during component implementation and/or usage, the whole system is able to deal with it in a reasonable way
  - DAU and Super-User
  - Automated attacks
  - Intelligent attacks
  - Communicative hurdlers
  - Interrupts and threaded interactions

Supporting principle – Development practice

- Test driven development
- Pairing within the Team
- Review with out-of-team Team-members
- Continuous Integration
- Detailed testing on contract level

Supporting principle – SOLID Design

- **S**RP, **O**CP, **L**SP, **I**SP, **D**IP

- Liskov Substitution Principle (LSP)

  “Functions that use pointers or references to base classes must be able to use objects of derived classes without knowing it."
  — Robert Martin

  Subtypes must be suitably act for their base type

Supporting principle – Organizational

- Multiple teams working on same product
- Teams are working on same code base
- Teams are delivering whole increments
- Teams are using each others code/components
- Use Scrum for utilizing successful and sustainable people involvement and product development

1. How is Scrum supporting robust architecture

Scrum is more than a framework

- “A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.”

- You need to understand it
  - In order to adjust and align on it
  - In order to surpass the mere letters

Defintion by:
Scrum Guide Oct 2011
scrum.org

(Some) Corner stones of Scrum

1. The Team commits to deliver working software in 30 days or less
2. Time is scheduled to show that software
3. The Team creates the software
4. The Team offers their work for inspection and adapts the plan for the next iteration

Timeboxing

- Time boxes
  - Are reserved time for a specific topic or discussion
  - May not be exceed beyond the allocated time
  - Time to act focused
- Time boxes are implemented all over Scrum
  - Sprint is a time box
  - Daily Scrum is a time box
  - Review is a time box
  - Retrospective is a time box
  - Planning is a time box

Why timeboxing?

- → Coordinate work between many people
- → Allotted space of focused work
- → “Getting the heart of it”
- → Identify difficult and controversial topics
- → Creativity required from solving complex problems requires focus
- → Working with the right people on the right topic

Why are timeboxes good for robust architecture?

→ Brings engineering to the point of maximal value

No “over engineering”

- Keeping the relevant and likely cases in mind

No “under engineering”

- Focused working
- Clear rule in case of identified problems
  - Engagement of the right people to the right time in a focused manner

The Urge to show running software

- Running software is inspectable by any user
  - Requirements can be easily verified and justified
  - Only integrated software make up a use case

- In the light of running software and user encounterment
  - Usability is proven
  - Reasonability is proven
  - Stability is proven
  - Collaboration of components is proven
  - Contribution to business value is proven

Why is running software supporting robust architecture?

- Through running software
  - Exchangeability of components is provable
  - Response of system regarding components is observable
  - Robustness of system regarding creative use, mistakes and error in usage and implementation is attestable

Scrum in a nutshell ...

- **Communication** over specification
  - Written specification are subject to one-way understanding
  - Communication implies vocal exchange and interactions
- Deliver **complete** incremental work
  - Finished and tested increments
- Frequent **increments** and integration
  - Everything is running all together

1. Employ both for best

- Scrum fosters robust architecture indirectly if wisely operated
  - Although it does not guarantee it
- Engage multiple Teams
  - They discuss, collaborate, make mistakes and strengthen your design
- Engaging each Team and support them
  - Support them in their creative work and keep an eye on the goal
  - Advise TDD, CI and distribute a solid understanding of LSP
  - Implement urge for showing software, early testing and early integration

Thank you

michael.mai@valtech.de

valtech_