WIRTSCHAFTSKAMMER ÖSTERREICH
Unternehmensberatung · Buchhaltung · IT

Agile & Lean

A Quick Overview

Werner Wild
EVOLUTION - Werner Wild Consulting, Innsbruck
Wien, 20. November 2015

Lean Software Development

- **Origins of Lean Development**
  The Toyota Production System
- **Adapted by Mary & Tom Poppendieck**
  (see References)
- **Principles of Lean Thinking**

The Toyota Production System

- **Just-in-time flow**
  - Non stock production
  - Build only what is needed
  - Eliminate everything which does not add value
- **Automation**
  - Stop if something goes wrong
  - Zero Inspection

Principles of Lean Thinking

1. Eliminate Waste
2. Create Knowledge
3. Defer Commitment
4. Deliver Fast
5. Build Quality In
6. Respect People
7. Optimize the Whole

Eliminate Waste

- **Taiichi Ohno (Toyota Production System):**

  All we are doing is looking at the timeline from the moment a customer gives us an order to the point when we collect the cash. And we are reducing this timeline by removing the non value-adding wastes.

Eliminate Waste

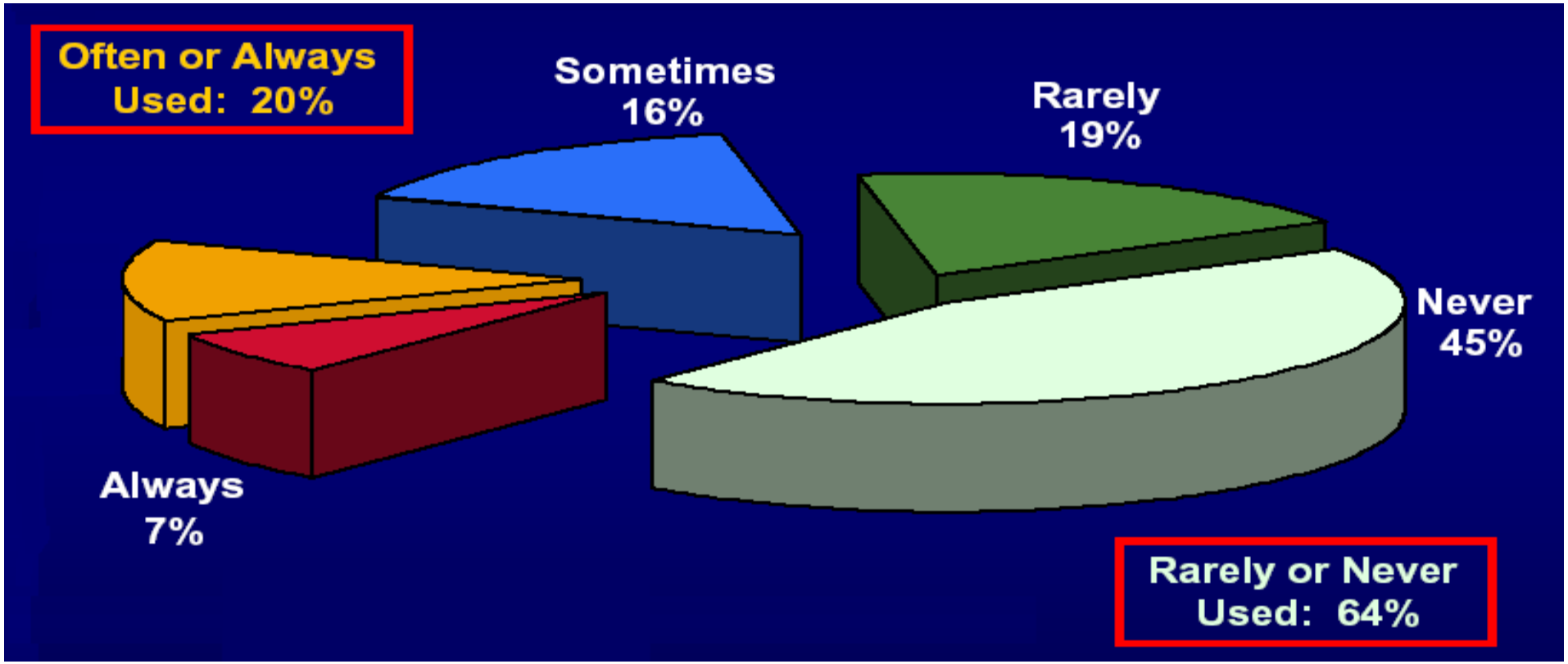
- **Waste**
  - Anything that does not create value for the customer
  - The customer would be equally happy with the software without it

- **Prime Directive of Lean Thinking**
  - Create Value for the customer
  - Improve the Value Stream by removing non value-adding activities

- **=> Value Stream Mapping !**

Huge Source of Waste

Features and Functions Used in a Typical System



Seeing Waste

Manufacturing	Software Development
In-Process Inventory	Partially Done Work
Over-Production	Extra Features
Extra Processing	Relearning
Transportation	Handoffs
Motion	Task Switching
Waiting	Delays
Defects	Defects

Principles of Lean Thinking

1. **Eliminate Waste**
2. **Create Knowledge**
3. **Defer Commitment**
4. **Deliver Fast**
5. **Build Quality In**
6. **Respect People**
7. **Optimize the Whole**

Defer Commitment

- The longer we defer decisions, the more we can learn
- Challenge to make decisions Just-in-time
- 2 Approaches for learning:
  - *Building a system which is change tolerant*
    - Refactoring
  - *Building several options*
    - Set-Based Design

Principles of Lean Thinking

1. Eliminate Waste
2. Create Knowledge
3. Defer Commitment
4. Deliver Fast
5. Build Quality In
6. Respect People
7. Optimize the Whole

Queues

- Cycle Time
  - Average End-to-End Process Time
    - From Entering The Airport
    - To Arriving at the Gate
- Time Spent in a Queue is Wasted Time
- The Goal: Reduce Cycle Time

Queuing Theory

- Little's Law

  Cycle Time = Things in Progress / Average Completion Rate

- 2 ways to reduce cycle time
  - Do things faster
  - Reduce the number of things in the process

=> use a **Kanban**

A Quick Overview

XP - 12 Practices

- **The Planning Game**
  Quickly determine the scope of the next release. Priorities
- **Small releases**
  Release new versions in very short cycles
- **Metaphor**
  Find a simple metaphor describing how the system works
- **Simple design**
  Make the design as simple as possible
- **Test Driven Development**
  Test the code continuously. Write the tests before the production code. Both Acceptance and Unit tests.
- **Refactoring**
  Mercilessly restructure the code to remove duplications, improve communication, simplify, or add flexibility - Intention Revealing Coding
- **Pair programming**
  Two brains at one machine
- **Collective ownership**
  Everyone owns and can change any code anywhere in the system
- **Continuous Delivery / DevOps**
  Integrate, build, test and deploy the system many times a day
- **Sustainable Pace**
  Don't work more than 40 hours a week
- **On-site customer / Whole team**
  Include real users in the team – they must speak with one voice though
- **Coding standard**
  Use a coding standard to improve communication

Experiences

Mobile Payment System

Safety Critical Software

(Ask me during the break)

Lessons Learned

- **Support** from the **top** is essential
- **Corporate culture** might be incompatible with the **Agile value system**
- **Users** and **Ops** must be involved from the start
- Contrary to popular belief, **Agile is highly disciplined !**
- As always, start with the **low hanging fruits for quick wins** to get as many on board as possible, early on

Lessons Learned

- Use **Kanban** to create **Flow** and **Limit Work in Progress**
- **Measure** and **Optimize Cycle Time**
- Scrum by itself is not enough, focus on **Technical Practices** at least equally, right from the beginning, even if it seems harder!
- **Automate**, **Automate**, **Automate!**
- Never let a **good crisis** go to waste :-) !

References

Lean Software Development:

- Mary and Tom Poppendieck: *An Agile Toolkit for Software Develop*
- Mary and Tom Poppendieck: *Implementing Lean Software Develop*

XP: http://c2.com/cgi/wiki?ExtremeProgrammingCorePractices

Contact

Werner Wild

EVOLUTION(R) - Werner Wild Consulting
Jahnstrasse 26
A-6020 Innsbruck

Email: werner.wild@uibk.ac.at