

Case

Sure, we don't apply the waterfall --- everyone knows it doesn't work.

We've adopted <iterative method X> and are into our first project.

We've been at it for two months and have the use case analysis nearly finished, and the plan and schedule of what we'll be doing in the next iteration.

After review and approval of the final requirements set and iteration schedule, we'll start programming.

Are you really agile?

Michael Franken
mfranken@xebia.com

Introduction

- Agile is hot!
- But why?
- Is it code quality? Testability? Fun?
- What issues is Agile development addressing?
- What is it about?

Agile is all about adding *business* value early

Nature of projects

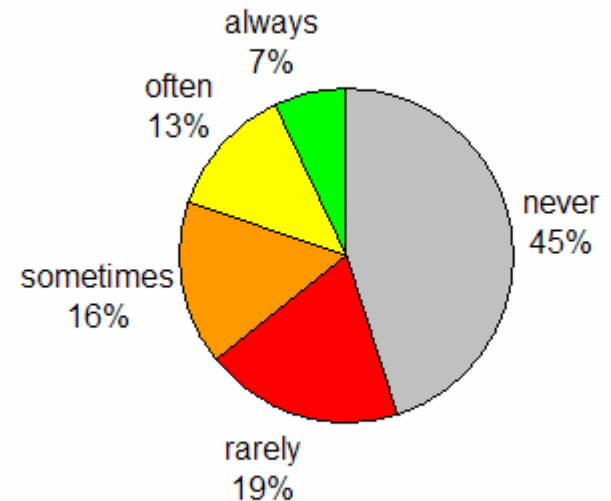
- Predictable or mass manufacturing:
 - Assembly line
- New product development or inventive projects
- *Software development is new product development*

Process Mismatch

- BUFD, estimates, predictive planning are misapplied
- Architecture metaphor does not work
- Waterfall paradigm does not apply to software development
 - Requirements change (> 25%)
 - Most defects origin from requirements (41%!)
 - Cost of fixing defects increases non-linearly with time

Waterfall

- Plan the work, work the plan
- Waterfall promotes BUFD:
 - 45% of all features are never used, 19% rarely
- Think early, fail late



Jim Johnson. The Standish Group International Inc. 2002

Waterfall history

- 70's answer to 60's ad hoc coding
- Author (Winston Royce) stated that waterfall is only applicable for the most straightforward un-novel projects.
- DoD promoted Waterfall in 70's and 80's (DOD-STD-2167). Changed that in 1994 to promote iterative and evolutionary development (MIL-STD-498)

Consequences

- Projects fail (CHAOS report: Standish group)

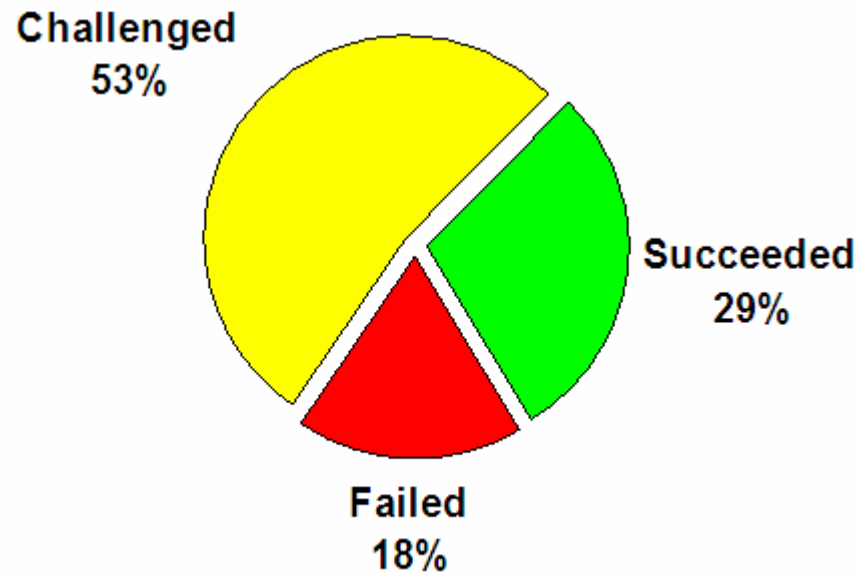
Project Size	People	Time (months)	Success Rate
Less than \$750K	6	6	55%
\$750K to \$1.5M	12	9	33%
\$1.5M to \$3M	25	12	25%
\$3M to \$6M	40	18	15%
\$6M to \$10M	+250	+24	8%
Over \$10M	+500	+36	0%

- Mainly because lack of:
 - User Involvement,
 - Executive Support and
 - Clear Business Objectives.

CHAOS 2004

SURVEY RESULTS

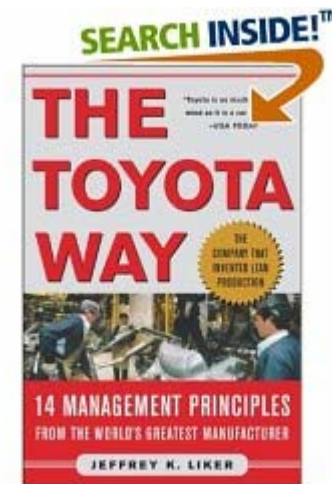
Resolution of Projects



Then What?

The Toyota Way

- Lean manufacturing since 1950's
- High quality, high productivity *and* low cost
- Action
- Inspect and adapt
- Eliminate waste



Lean software development

- Iterative development
- Evolutionary development
- Agile development

Iterative development explained again

- Develop in multiple iterations
 - Each iteration is complete ‘project’
 - Analysis, design, coding *and* test
 - Almost always delivers new functionality
 - Possibly deployed to production
 - *All* parts from *all* teams
- Iteration lasts between 1 and 6 weeks
- Timeboxed, resources and time fixed
- No external changes during iteration

Typical misinterpretation of phases

- Inception phase for Requirements analysis
- Elaboration for Design
- Construction for Coding
- Transition for Testing

*Some say RUP was not intended this way,
although it is hard to read differently*

Examples

- RUP, UP, XP, Scrum, Evo, OPEN, DSDM, Crystal, Spiral, Microsoft Solutions Framework, Adaptive Software Development, Feature driven development, Lean development, Pragmatic Programming

Evolutionary development expl'd again

- Not a method to constantly change your mind, but to constantly improve and learn.
- Forces learning
- Functional and technical challenges tackled early on
- Constantly refine plan, specs, code, etc.
- Late planning of next iteration

*'It is not the strongest of the species
that survive, nor the most intelligent,
but the one most responsive to
change.'*

— Charles Darwin

Agile explained again

- Satisfy the customer through early and continuous delivery of valuable software
 - *Not* about TDD, pair programming
- Iterative and evolutionary
- Lightweight
 - Eliminate waste
- Focus on communication and people
 - Interactive
 - Self directed
- Quality as a means to achieve sustainable productivity

Agile definitions

- Agile manifesto
 - *Individuals and interactions* over processes and tools
 - *Working software* over comprehensive documentation
 - *Customer collaboration* over contract negotiation
 - *Responding to change* over following a plan
- Agile principles
 - <http://agilemanifesto.org/principles.html>

Agile principles

We follow these principles:

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Is this new?

- PDSA 1930 (cycles and wheels)
- SAGE project 1956
- DEMING 1940
- NASA Mercury 1963 (Test First Development)
- IBM Federal Service Division 70's (Harlan Mills, Barry Boehm) including life critical DoD
- Gilb published Evo in 1976

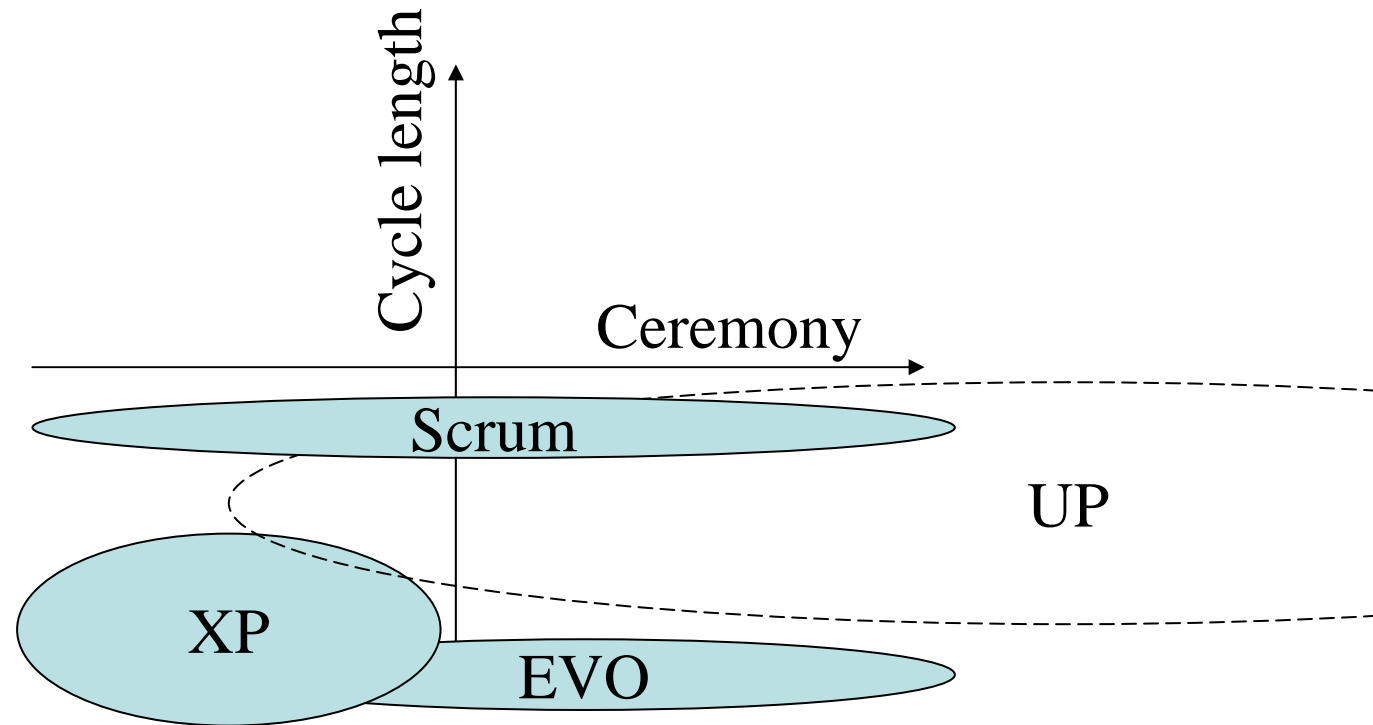


4 popular methods

- Scrum
- XP
- Evo
- Unified Process

Classification

- By Ceremony (barely sufficient)
- By Cycles (short)



Scrum

- By Ken Schwaber, Jeff Sutherland
- Self directed and self organizing teams
- No external addition of work to an iteration, once chosen
- Daily stand-up meetings with special questions
- 30 calendar day iterations (sprints)
- Demo to external stakeholders at end of each iteration
- Each iteration: client-driven adaptive planning

Scrum more

- Flexible on ceremony: workproducts not defined
- All types of projects: life critical and more than 100s of people
- Empirical process: planning, staging, development and release.
- Product, Release and Sprint Backlog: tasks of 4 to 16 hours max.
- Sprint backlog updated constantly (at scrum meeting)

Scrum some more

- Scrum master and Product Owner
- Team of 7 max
- Chickens and pigs (team)
- Scrum meeting:
 - What have you done since the last Scrum?
 - What will you do between now and the next Scrum?
 - What is getting in the way (blocks) of meeting the iteration goals?

- By Kent Beck
- For small to medium projects (< 20 people, not life critical)
- Low on ceremony
- Explicit methods for programmers
- Complete and consistent, not a pick and choose

More XP

- Simple things taken to the Extreme:
 - Code reviews -> pair programming
 - Testing -> unit (TDD) and acceptance testing automated: red, green
 - No upfront design -> refactor
 - Continuous integration
 - Short iterations 1 – 2 weeks
 - Customer involvement full time
 - Constant communication, same room, one team
 - Sustainable pace -> no overtime

Some more XP

- Values:
 - Communication
 - Simplicity
 - Feedback
 - Courage
- Story cards
- Planning game

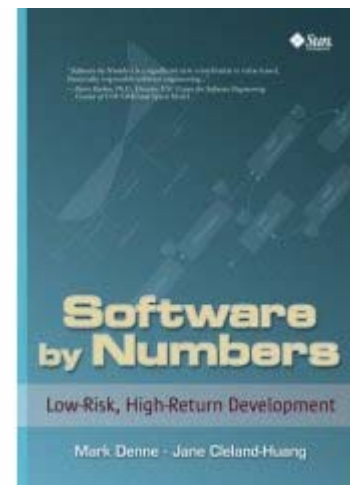
Unified Process

- Iterative
- Client and Risk driven
- Deliver value to customer
- Accommodate change early
- Work as one team
- Scale up and down on Ceremony
 - Most (RUP) workproducts are optional

- By Tom Gilb
- Short iterations, with evolutionary delivery
- Evolutionary requirements and design
- Adaptive client driven planning
- Quantification
- Planguage

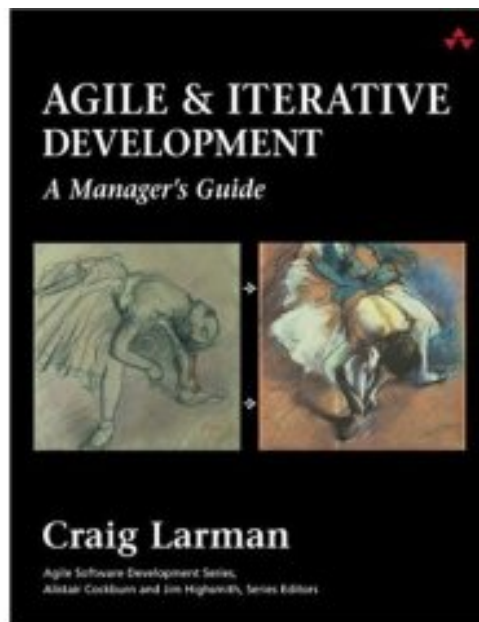
What's missing and what's next?

- Scrum will take off, PM will change
- Experience in distributed agile development in offshoring projects
- Quantification
 - Bringing software engineering practices back into Agile development
- Financially backed decisions
 - Software by numbers, Mark Denne and Jane Cleland-Huang



Want more?

- Read this:



- Mail me: mfranken@xebia.com