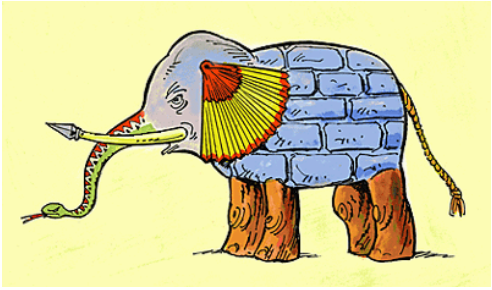
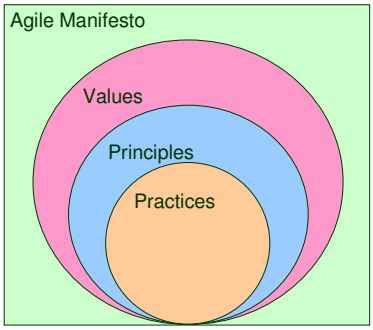
Introduction to Agile

Presented by:
Aaron Korver



What is Agile?



Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through that work we have come to value:

Individuals and interactions over **processes and tools**
Working software over **comprehensive documentation**
Customer collaboration over **contract negotiation**
Responding to change over **following a plan**

That is, while there is value in the items on the right we value the items on the left more.

Agile Values

- Honesty
- Trust
- Cooperation
- Respect
- Educate
- Listen
- Involvement
- What values does your team embrace?

Agility is…
Successful Agility - Principles

- Successfully adopting agile means adopting the principles and philosophies
- "Agile projects succeed when the team gets the spirit of agility." – Ron Jeffries (APM Group)
- "In agile development, we value following the ***principles*** over following specific ***practices***." – Alistair Cockburn (Agile Use Cases Presentation)

Principles of Agile Development

- **Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.**
- Agile focuses on the *delivery* of software
- Delivering software early and often allows for quick wins and early feedback about the requirements, the team, and the process
- The emphasis is to deliver the most important parts first
  - If the project ends, there are still valuable artifacts
  - Development proceeds from most important to less important
  - Note that this may (and will!) change during the course of the project

Principles of Agile Development

- **Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.**
- Most projects run 1 to 3 month cycles [Cockburn]
- Within the cycle, work is done in short iterations
  - Iterations usually last from 2 to 4 weeks
  - The iteration is *complete* at the end of the iteration, at the expense of moving requirements
  - XP iterations are 2 to 3 weeks
- Users review progress throughout the process

Principles of Agile Development

- **Working software is the primary measure of progress.**
- Code is the most important deliverable
- Agile methodologies place a premium on getting something up and running early and evolving it over time
- Agile does ***not*** discard the need for design documentation
  - Diagram to understand, then move to code
  - Documentation is secondary to working code

Principles of Agile Development

- **Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.**
- This works because of
  - Early and frequent delivery of running software
  - Use of iterative and timeboxing techniques
  - Continual attention to architecture
  - Willingness to update the design
- If you can gracefully accommodate changing requirements and your competition cannot, you win

Principles of Agile Development

- **Business people and developers must work together daily throughout the project.**
- There is a strong correlation between links to users and project success [Frakes 1995]
- The best links are through onsite business expertise and daily discussion
  - The longer it takes to get information to and from the developers, the more damage will occur in the project
  - Information flow is similar to convection currents
- XP insists that business stakeholders are part of the development team

Principles of Agile Development

- **Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.**
- "It is better to have motivated, skilled people communicating well and using no process at all than a well-defined process used by unmotivated individuals." [Cockburn 2002]
- "We hire good people, give them the tools and training to get their work done, and get out of their way." [Thomas]

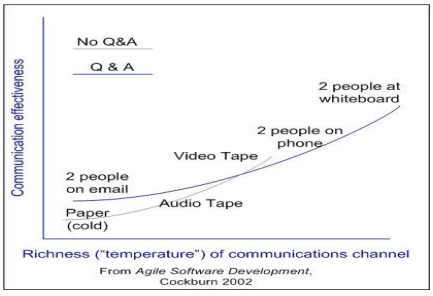
Developers are the Key

- You cannot overemphasize this point: developers, project managers, and business analysts make or break the project
- "All other factors being equal, a 90th-percentile team of analysts and programmers will be about four times as productive as a 15th-percentile team." [Boehm 1981]
- You should hire the smartest people you can find

Principles of Agile Development

- **The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.**
- Communication is like air conditioning
- Cockburn defines *modalities* of communications and their effectiveness (*Agile Software Development*, 2002)

Cockburn's Modalities and Effectiveness



Communication

- Face to face communication about specific topics is most efficient
- Agile also takes advantage of "knowledge radiators"
  - Simple, low-tech information stores
  - Poster board with index cards or Post-it notes
  - Project status beside the coffee pot
  - Automated build report via web browser (I'll talk about continuous integration later)

Principles of Agile Development

- **The best architectures, requirements, and designs emerge from self-organizing teams.**
- This means either that architectures, requirements, and designs emerge
  - in small steps over time
  - as a logical consequence of human-centric rules the team uses
- All 3 must be allowed to adjust over time
- An architecture that grows in steps can follow the changing knowledge of the team and the changing wishes of the users

Principles of Agile Development

- **Continuous attention to technical excellence and good design enhances agility.**
- The design must evolve
  - You must create good designs initially
  - You must review and improve designs regularly
  - Design cleanup as compared to debt [Cunningham 2001]
- Technical excellence applies to all aspects of the project
  - Build management
  - Coding standards
  - Collaboration documentation

Principles of Agile Development

- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Social aspect
  - An alert, engaged staff is more effect than a tired, plodding staff
  - Long hours are a symptom that something has gone wrong
  - Only work 8 hours a day; never put in overtime 2 weeks in a row
- Technical aspect
  - Software development can be viewed as a long strategic game
  - Each move sets up the next move

Principles of Agile Development

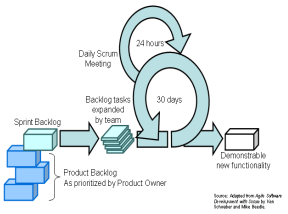
- **Simplicity--the art of maximizing the amount of work not done--is essential.**
- "This letter is longer than I wish, for I had not the time to make it shorter" *Blaise Pascal*
- It is sometimes more difficult to make things simpler
- A cumbersome model is easier to produce than a simple one
- DRY principle (Don't Repeat Yourself)
- "Design for simplicity; add complexity only where you must." *Art of Unix Programming*, Raymond 2003
- "Code smell"

Principles of Agile Development

- **At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.**
- Constant awareness is key
- Never get lazy
- XP encourages this every step of the way
- Questions:
  - How light is too light?
  - How simple is too simple?
  - Can we make this simpler

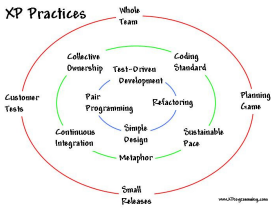
Agile Methodologies
Scrum

- Project management focus
- Empirical process
- Can be wrapped around engineering practices



Agile Methodologies
eXtreme Programming (XP)

- Developer Focus
- Based on Values:
  - Simplicity
  - Communication
  - Feedback
  - Courage
- 12 Core Practices



Agile Myths

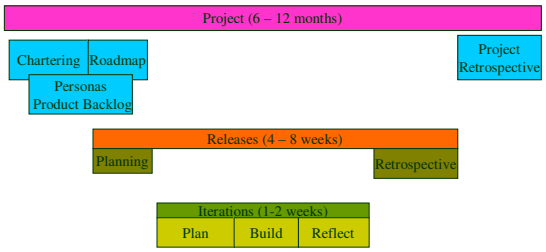
- Agile will work for you.
- Agile is simple.
- There is no planning in agile.
- There are no documents created in agile.
- Agile promotes “cowboy coding”.

Not a silver bullet

Agile is simple…

…but it isn’t easy

No planning in agile?

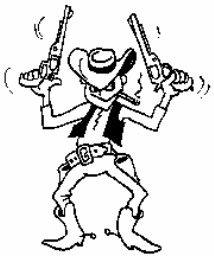


There are no documents created in an agile process.

There are no unnecessary documents created in an agile process.

Cowboy Coders

- Practices such as "Pair Programming".
- Daily stand-up meetings.
- Co-located team rooms.
- Customer demos.
- Teamwork vs. Mework



The "Evil" Side of Agile

- People have to work together
- Increased visibility into the organization.
- Loss of titles/roles
- Change from "Command and Control" to "Servant Leadership"
- Less up front design
- Sustainable pace
- Projects fail fast/increased accountability
- Cost to physically reorganize workstations

History of Agile

- 1950's W. Edwards Deming revitalizes Japan manufacturing $Quality = \frac{results\ of\ work\ efforts}{all\ costs}$
- 1986 Takeuchi, H. and I. Nonaka write "The New New Product Development Game" in the Harvard Business Review
- 1993 Jeff Sutherland creates Scrum based upon Takeuchi and Nonaka's ideas
- 1995 Ken Schwaber formalizes Scrum
- 1996 Kent Beck creates eXtreme Programming
- 2001 Snowbird conference produces the Agile Manifesto

Who is using Agile?

- Locally
  - Iowa Student Loan
  - ITA Group
- Worldwide
  - Coca-Cola Web team
  - British Telecom
  - Yahoo!

Discussion

Retrospectives