Software Engineering @ LEIC/LETI

Software Process - Agile Methods

Software Process

Agile Methods

Agile Methods

The 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

(http://agilemanifesto.org/)

Principle	Description
Customer involvement	Customers should be closely involved throughout the development process. Their role is provide and prioritize new system requirements and to evaluate the iterations of the system.
Incremental delivery	The software is developed in increments with the customer specifying the requirements to be included in each increment.
People not process	The skills of the development team should be recognized and exploited. Team members should be left to develop their own ways of working without prescriptive processes.
Embrace change	Expect the system requirements to change and so design the system to accommodate these changes.
Maintain simplicity	Focus on simplicity in both the software being developed and in the development process. Wherever possible, actively work to eliminate complexity from the system.

(Fig 3.2, Sommerville)

eXtreme Programming

(http://www.extremeprogramming.org/)

How frequent risks are dealt by XP

- Business changes – customer can change requirements that are not yet implemented

- Schedule slips – several small versions, short duration

- Project cancelled – customer chooses the smaller version that has the biggest business impact

- System goes sour – a set of test is executed whenever the system is changed

- Defect rate – testing from different perspectives:

  - Programmer

  - User

- Business misunderstood – customer is part of the team
- False feature rich – only the highest priority activities are performed
- Staff turnover – foster communication, programmers estimate their work duration

The Rules of Extreme Programming

Planning

- User stories are written.
- Release planning creates the release schedule.
- Make frequent small releases.
- The project is divided into iterations.
- Iteration planning starts each iteration.

Managing

- Give the team a dedicated open work space.
- Set a sustainable pace.
- A stand up meeting starts each day.
- The Project Velocity is measured.
- Move people around.
- Fix XP when it breaks.

Designing

- Simplicity.
- Choose a system metaphor.
- Use CRC cards for design sessions.
- Create spike solutions to reduce risk.
- No functionality is added early.
- Refactor whenever and wherever possible.

Full Size Extreme Programming Project

Coding

- The customer is always available.
- Code must be written to agreed standards.
- Code the unit test first.
- All production code is pair programmed.
- Only one pair integrates code at a time.
- Integrate often.
- Set up a dedicated integration computer.
- Use collective ownership.

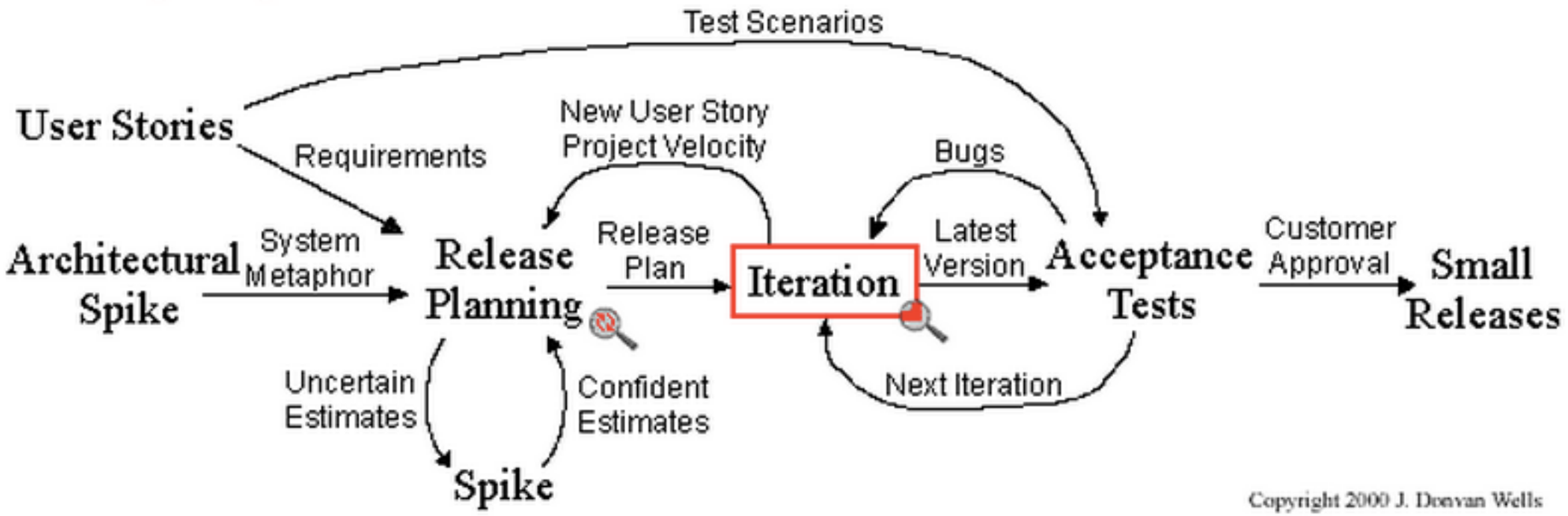
Testing

- All code must have unit tests.
- All code must pass all unit tests before it can be released.
- When a bug is found tests are created.
- Acceptance tests are run often and the score is published.

Let's review the values of Extreme Programming (XP) next.

ExtremeProgramming.org home | XP Map | XP Values | Test framework | About the Author

Extreme Programming Project



ExtremeProgramming.org home | Zoom in on Iteration. | Starting with XP | Email the webmaster

XPlorations

Wiki Wiki
The Portland
Pattern Repository

XProgramming.com

Scaling Agile Methods

- lack of contract
- deal with maintenance
- world distributed teams