CALIFORNIA STATE UNIVERSITY
FULLERTON™

COLLEGE OF ENGINEERING
AND COMPUTER SCIENCE

Advanced Software Process

The Agile Process

XP

Dave Garcia-Gomez
Faculty / Lecturer
Department of Computer Science

Contents

- Extreme Programming (XP)
- Values
- Core Practices
- XP Lifecycle
- Work-products
- Roles
- Practices

Extreme Programming (XP)

Values

Extreme Programming (XP) is a well-known agile method.

- XP emphasizes:
  - Collaboration
  - Quick and early software creation
  - Skillful development practices
- XP is founded on four values:
  - Communication
  - Simplicity
  - Feedback
  - Courage

Extreme Programming (XP)

Core Practices

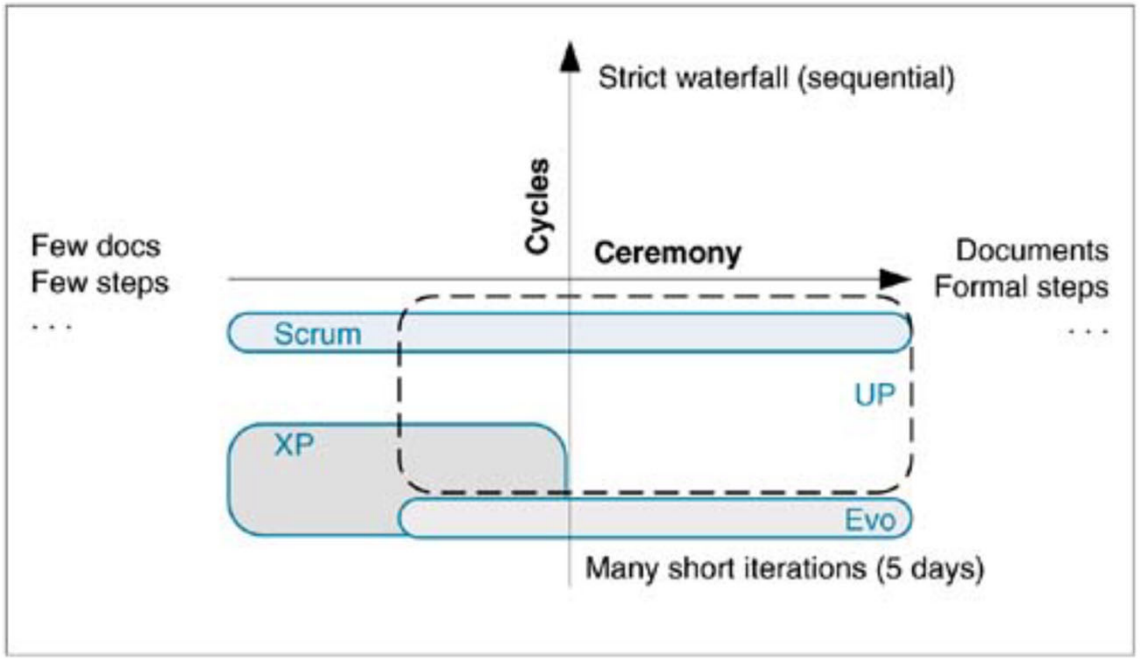
In addition to IID, XP recommends 12 core practices:

1. Planning game
2. Small, frequent releases
3. System metaphors
4. Simple design
5. Testing
6. Frequent factoring
7. Pair programming
8. Team code ownership
9. Continuous integration
10. Sustainable pace
11. Whole team together
12. Coding standards

Extreme Programming (XP)

Classification

XP on the cycle and ceremony scale.



XP is low on the ceremony scale; it has only a small set of a predefined, informal workproducts, such as paper index cards for summarizing feature requests, called story cards.

For average projects, the recommended length of a timeboxed iteration is between one and three weeks – somewhat shorter than for UP or Scrum.

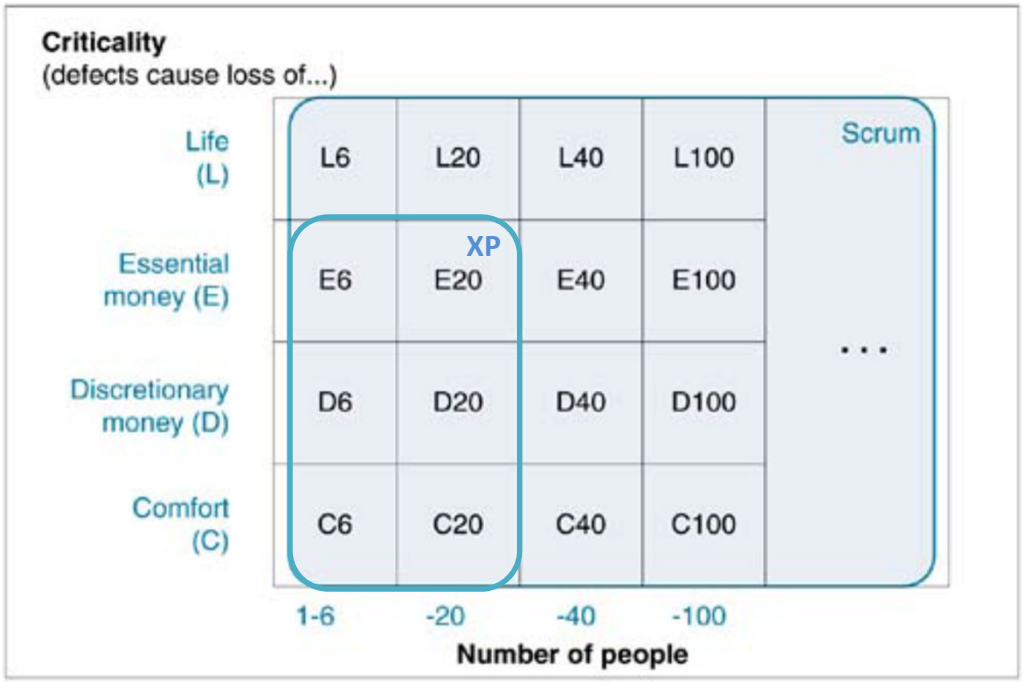
- Scrum: 30 days (4 w)
- XP: 1 – 3 weeks
- UP: 2 – 6 weeks
- Evo: 1 – 2 weeks

Extreme Programming (XP)

Classification

In terms of scope on the Cockburn scale, XP covers the cells shown in the following figure.

XP on the Cockburn scale



A method selector = a function of (criticality, size, priority): how much formal process (ceremony, cycles) a software project requires: Cockburn scale (criticality, size)

- L: Loss of Life
- E: Loss of Essential Money
- D: Loss of Discretionary Money
- C: Loss of Comfort

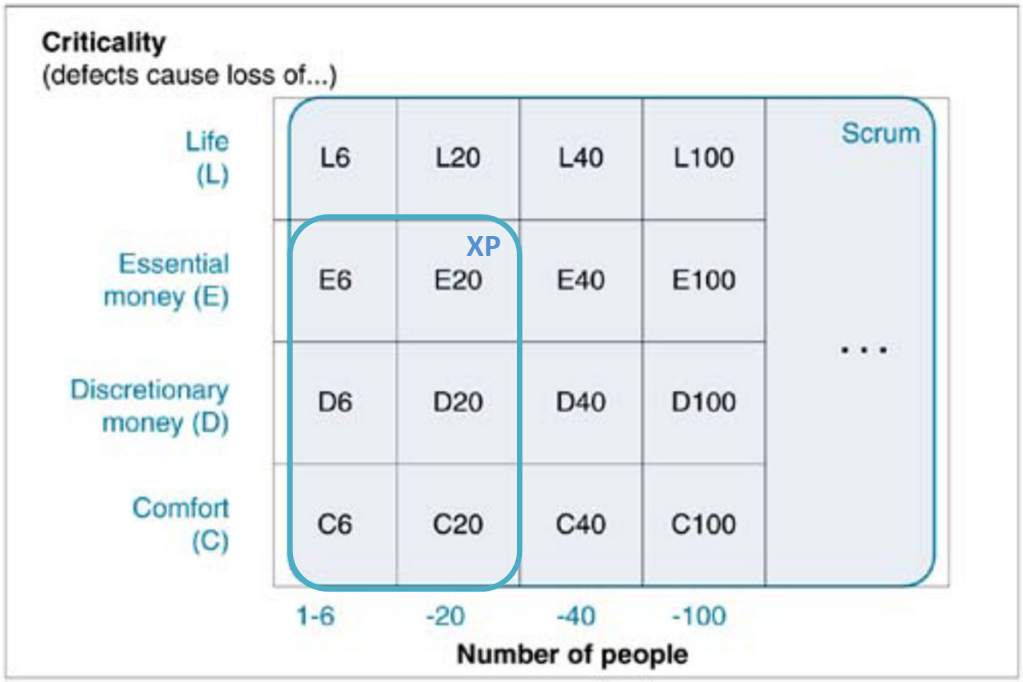
Reference:
http://alistair.cockburn.us/Cockburn+Scale

Extreme Programming (XP)

Classification

In terms of scope on the Cockburn scale, XP covers the cells shown in the following figure.

XP on the Cockburn scale



A refreshing quality of the original XP description was the statement of known applicability: It had been proven on projects involving roughly 10 developers or fewer, and not proven for safety-critical systems. Nevertheless, it has been more recently applied with larger teams.

Extreme Programming (XP)

- XP, created by Kent Beck [Beck 00], is an IID method that stresses customer satisfaction through:
  - rapid creation of high-value software
  - skillful and sustainable software development techniques
  - flexible response to change
- It is aimed at relatively small team projects, usually with delivery dates under one year.
- Iterations are short – usually one to three weeks.

Extreme Programming (XP)

As the word 'programming' suggests, it provides explicit methods for programmers, so they can more confidently respond to changing requirements, even late in the project, and still produce quality code.

These include test-driven development, refactoring, pair programming, and continuous integration, among others.

In contrast to most methods, some XP practices truly are adopted by developers - they sense its practical programmer-relevant techniques.

Extreme Programming (XP)

XP is very communication and team-oriented:

- Customers, developers, and managers form a team working in a common project room to quickly deliver software with high business value.

XP Lifecycle

XP Lifecycle

- Exploration Phase
- Planning Phase
- Iterations to Release Phase
- Productionizing Phase
- Maintenance Phase

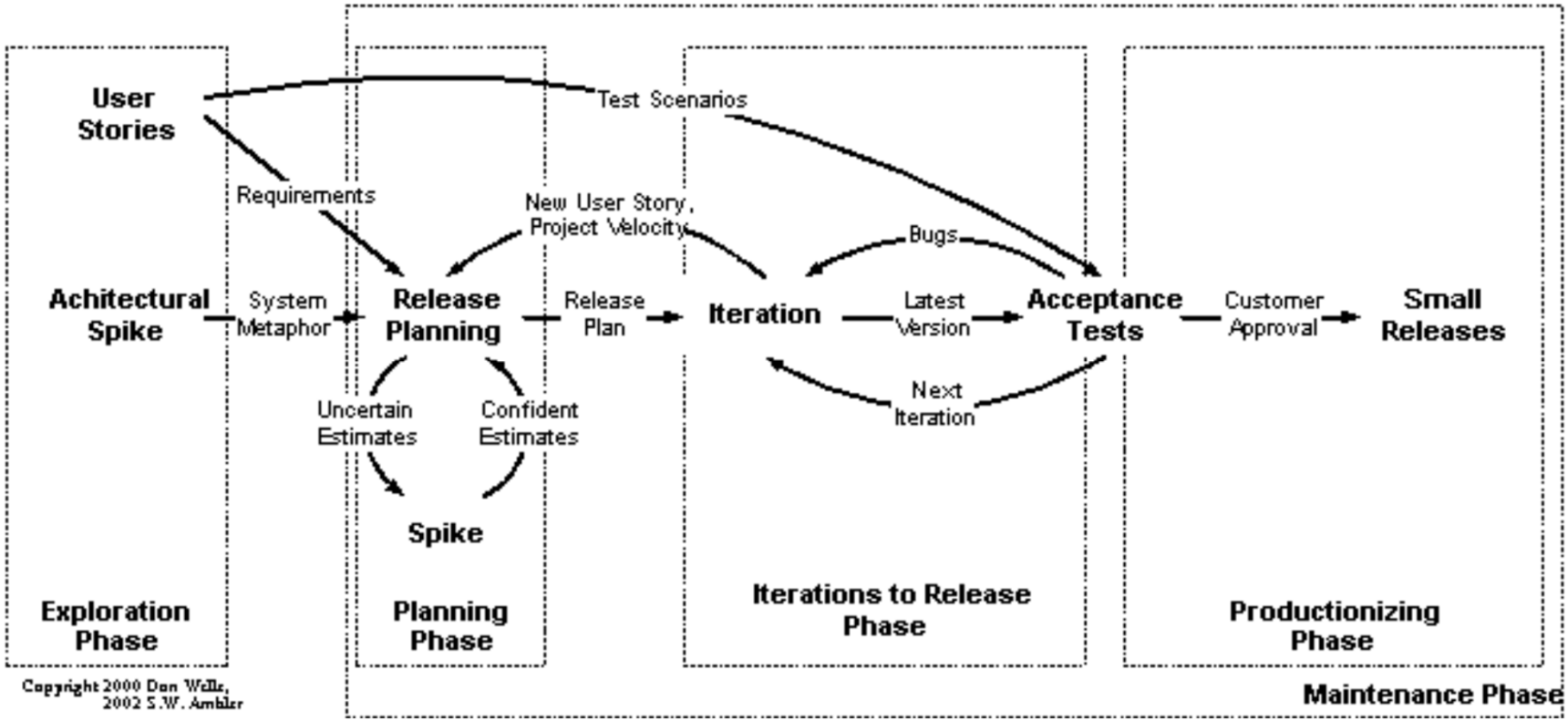
XP Lifecycle

XP Lifecycle

EXPLORATION	PLANNING	ITERATIONS TO FIRST RELEASE	PRODUCTIONIZING	MAINTENANCE
Purpose: - Enough well-estimated story cards for first release. - Feasibility ensured.	**Purpose**: - Agree on date and stories for first release.	**Purpose**: - Implement a tested system ready for release.	**Purpose**: - Operational deployment	**Purpose**: - Enhance, fix. - Build major releases
Activities: - prototypes - exploratory proof of technology programming - story card writing and estimating	**Activities**: - Release Planning Game - story card writing and estimating	**Activities**: - testing and programming - Iteration Planning Game - task writing and estimating	**Activities**: - documentation - training - marketing - . . .	**Activities**: - May include these phases again, for incremental releases.

XP Lifecycle

XP Project Lifecycle



XP Lifecycle

XP Project Lifecycle Definitions

Metaphor: a conceptual framework; a logical architecture of the system; a shared story or description of how the system works.; a description of how you intend to build your system. The metaphor is defined during an architectural spike early in the project (during the 1st iteration or pre-iteration).

Architectural spike: intends to identify areas of high risk, to get started with estimating them correctly.

Spike: end-to-end, but very thin; depth-first, from the top to the bottom, then closed-loop.

Project velocity: a measure of project progress.

XP Lifecycle

1. Like many projects, XP can start with exploration. Some story cards (features) may be written, with rough estimates.

2. In the Release Planning Game, the customers and developers complete the story cards and rough estimates, and then decide what to do for the next release.

XP Lifecycle

3. For the next iteration, in the Iteration Planning Game, customers pick stories to implement. They choose stories – and thus steer the project – based on current status, and their latest priorities for the release.

   – Developers then break the stories into many short, estimated tasks.

XP Lifecycle

- Finally, a review of the total estimated task-level effort may lead to readjustment of the chosen stories, as XP does not allow overworking the developers with more than they can do based on "family-friendly" work days, such as an eight-hour day.

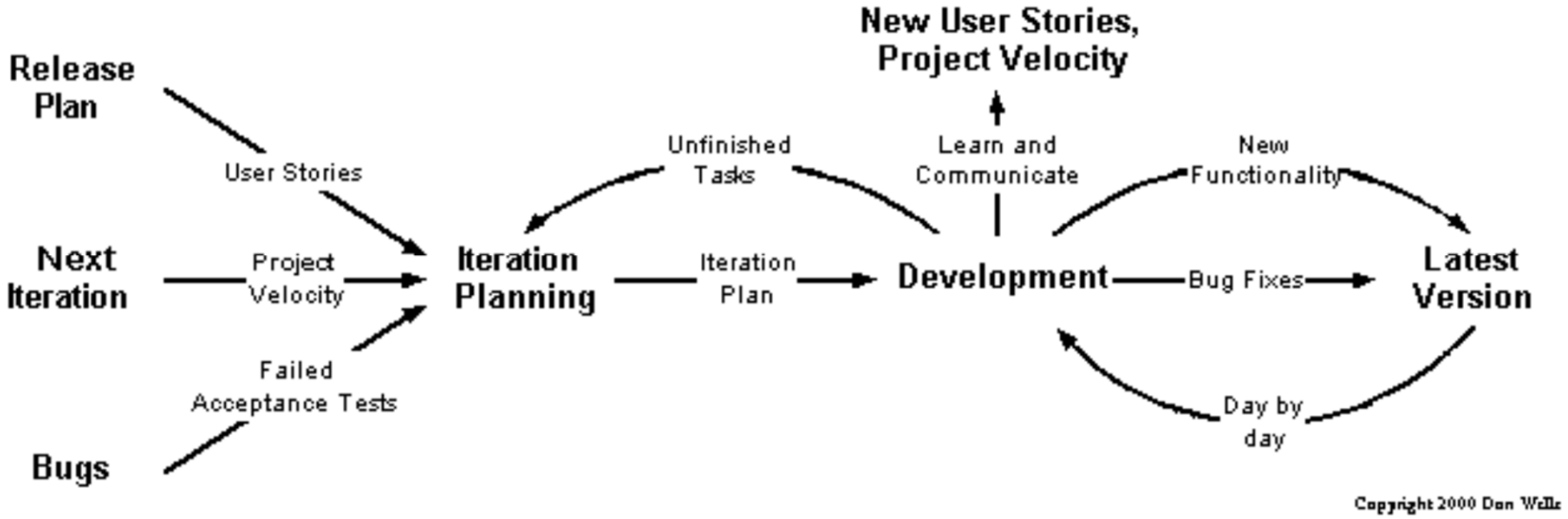
- Overtime is seriously discouraged in XP; it is viewed as a sign of a dysfunctional project, increasingly unhappy people, and dropping productivity and quality.

XP Lifecycle

4. Developers implement the stories within the agreed timeboxed period, continually collaborating with customers (in the common project room) on tests and requirement details.
5. If not finished for release, return to step 3 for the next iteration.

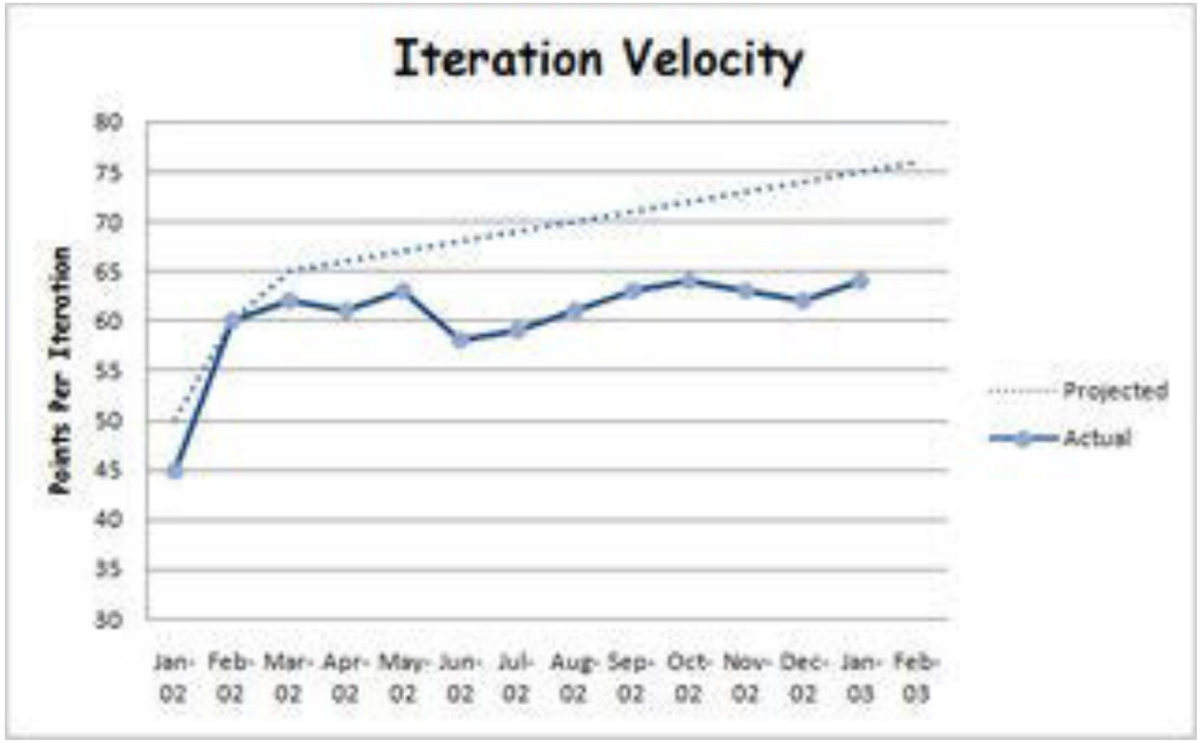
XP Lifecycle

The XP Iteration Lifecycle



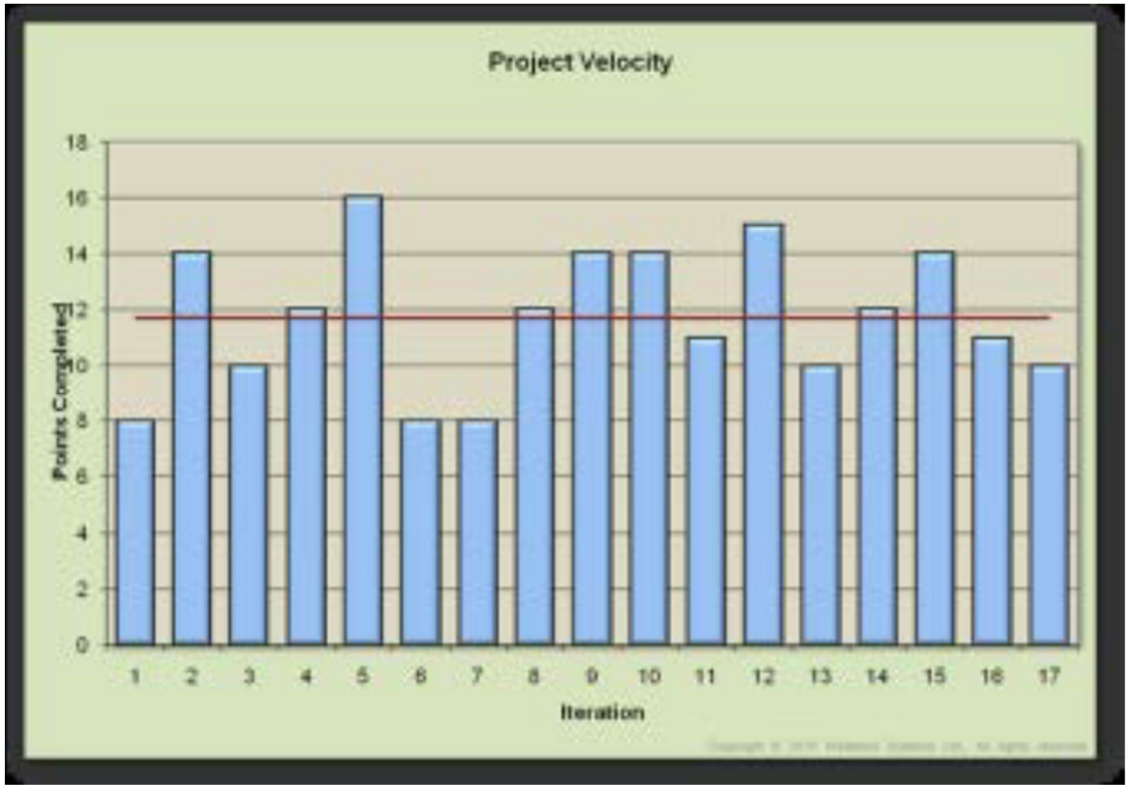
XP Lifecycle

The XP Iteration Lifecycle



XP Lifecycle

The XP Iteration Lifecycle



Project velocity: a measure showing how much move fast, and to which direction the project is going; not simply measure how much code is written, but measure the value from the customer's view; how much user stories are completed during an iteration and estimation; the ratio between estimated time and real calendar time.

XP Lifecycle

The XP Iteration Lifecycle

Story	Estimate (Points)	Completed?
Customer Searches for Book by Author	2	Yes
Customer Searches for Book by Category	3	Yes
Customer Searches for Book by Title	2	Yes
Customer Searches for Book by ISBN	1	Yes
Preferred Customer receives Discount	2	No

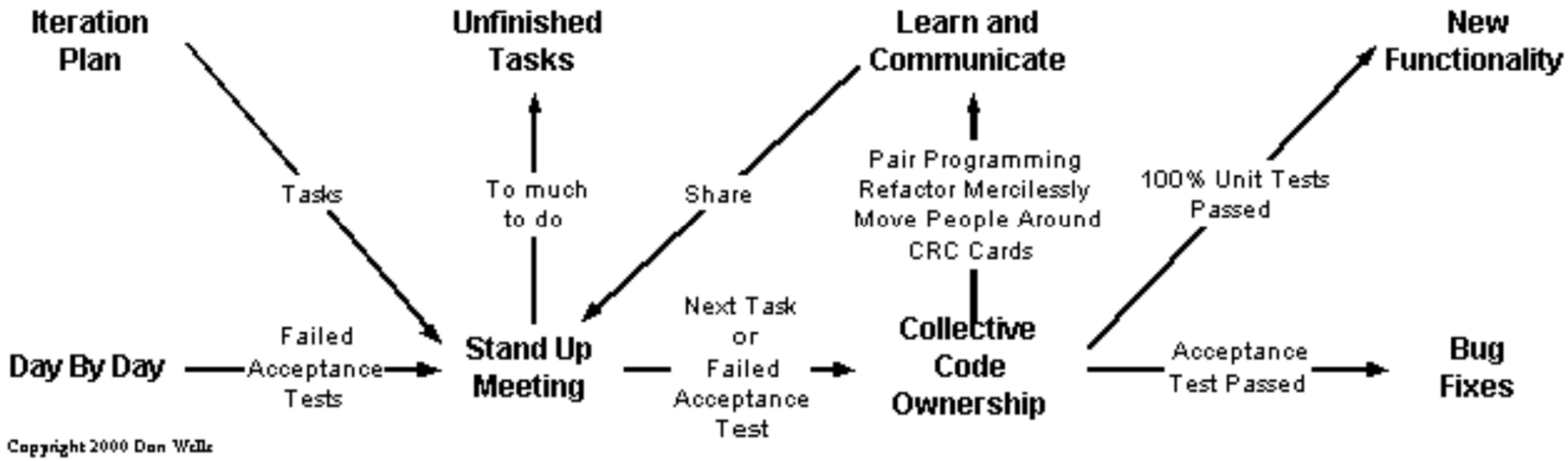
XP Lifecycle

The XP Iteration Lifecycle

Project velocity: a measure showing how much move fast, and to which direction the project is going; not simply measure how much code is written, but measure the value from the customer's view; how much user stories are completed during an iteration and estimation; the ratio between estimated time and real calendar time.

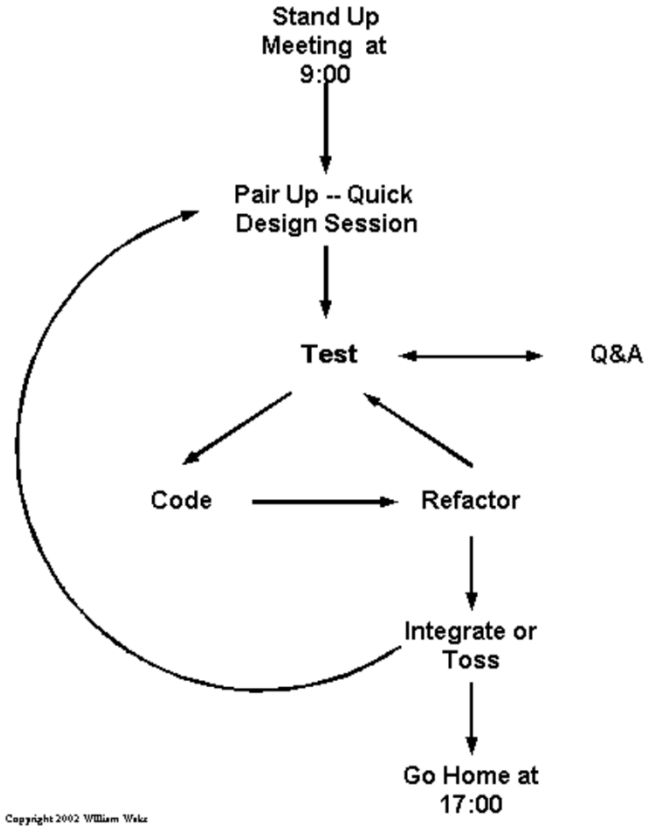
XP Lifecycle

The XP Development Lifecycle



XP Lifecycle

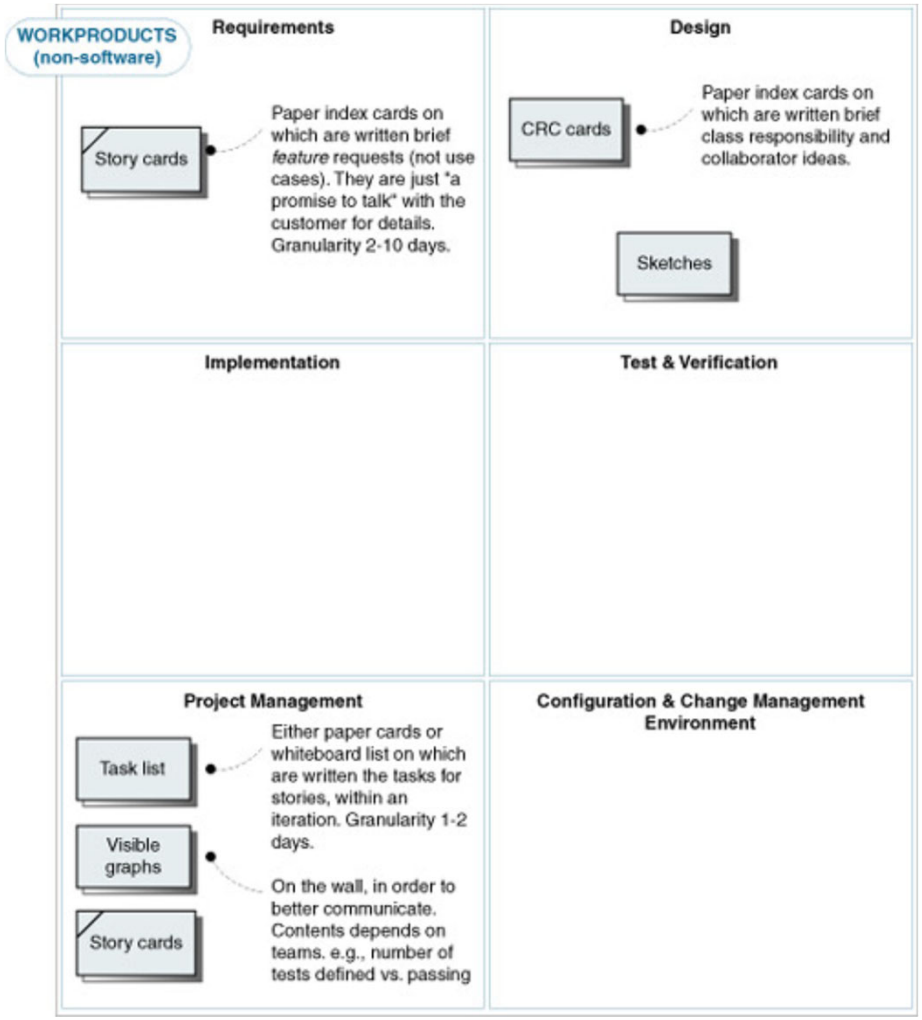
Typical Day of an XP Developer



Pair programming is a programming technique in which two programmers work together at one computer. One programmer (driver) types in code, while the other programmer (observer, navigator, reviewer) reviews the code.

XP Lifecycle

Workproducts

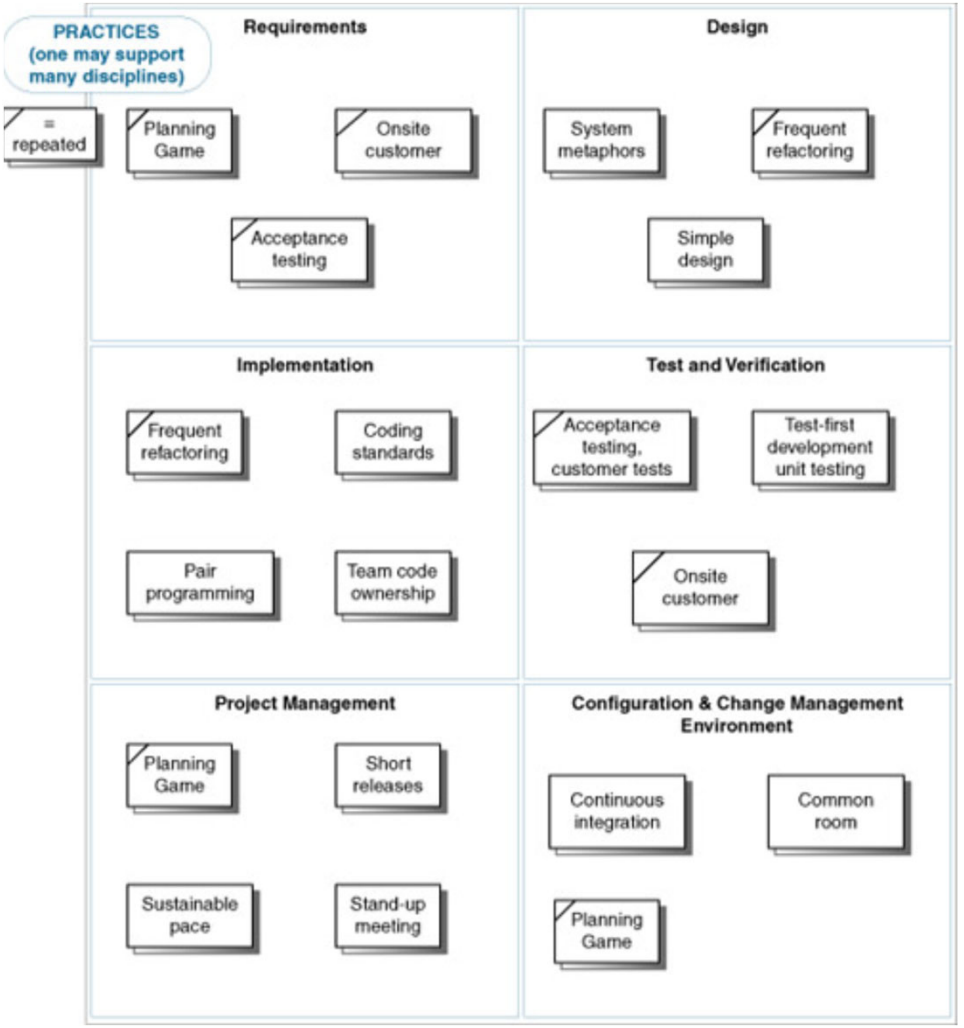


XP Lifecycle

Roles

XP Lifecycle

Practices



Story Cards

- A handwritten note on a paper index card.
- During the Planning Game, many of these are written.
- Sample story card
  - Find available classes, upload a course material
- The story cards record user stories: features, fixes, or nonfunctional requirements that the user wants.
- Stories are usually in the one-day to three-week range of estimated duration.

Stories, Use Cases & Features

- Contrary to some misunderstanding, XP stories are not use cases or scenarios.
- They usually represent features.
- Note that XP prefers a feature-driven approach to describing requirements rather than the use-case-driven approach that UP promotes.

Stories, Use Cases & Features

- A feature is a service that the system provides to fulfill customers' needs.
- A use case describes a series of interactions between users and the system to accomplish their goals. A use case describes how users and the system interact to realize the identified feature.

Stories and Communication

In XP, oral communication is preferred, and the story card purpose is not to detail the user story, but to jot a summary, make references to other documents, and in general, to view the card as "a promise to talk" (in Alistair Cockburn's words) with the customer who wrote it, by the developers implementing it.

- Since whole team together is an XP practice, the card donor should be readily available.

Stories and Communication

XP coaches vary on their advice regarding granularity for estimation.

- Some say stories can be in the two-day to two-week range of effort.
- Others recommend stories be estimated in units of one, two, or three weeks, but not in finer person-day units.

Stories for Estimation

XP coaches vary on their advice regarding granularity for estimation.

- Some say stories can be in the two-day to two-week range of effort.
- Others recommend stories be estimated in units of one, two, or three weeks, but not in finer person-day units.

Embrace Change

- Onsite customer proxies
- Customer on call
- Embrace change
  - The overarching attitude that XP promotes is to embrace rather than fight change, in the requirements, design, and code, and be able to move quickly in response to change.

Task List

- During an Iteration Planning Game, the team convenes around a whiteboard, and generates a list of tasks for all the stories chosen for the iteration.
- Another popular alternative is to generate individual task cards.
- Once a task is chosen by a volunteer, they enter an effort estimate (in ideal engineering hours) – tasks should be in the 1–2 day range.

Volunteering

- Only by volunteering (accepted responsibility)
- Tasks are not assigned to people.
- Rather, during the Iteration Planning Game, people choose or volunteer for tasks.
- This leads to a higher degree of commitment and satisfaction in the self-accepted responsibility.

Light Modeling

Very light modeling

- XP encourages programming very early and does take to "extreme" the avoidance of up-front design work.
- Any more than 10 or 20 minutes of design thinking (e.g., at the whiteboard with sketches or notes) before programming is considered excessive.
- Contrast this with Scrum or the UP, for example, where a half-day of design thought near the start of an iteration is acceptable.

Minimal or "Just Enough" Documentation

- With the goal of getting to code fast, XP discourages writing unnecessary requirements, design, or management documents.
- The use of small paper index cards is preferred for jotting brief descriptions, as are verbal communication and elaboration.
- Note that the practice of "avoiding documentation" is compensated by the presence of an onsite customer.
- XP is not anti-documentation, but notes it has a cost, perhaps better spent on programming.

Measurement Visible Graphs - Communication

- Metrics
  - XP recommends daily measurement of progress and quality. (Measure at least one thing.)
  - It does not mandate the exact metrics, but to "use the simplest ones that could work."
    - Examples: numbers of completed tasks and stories, success rate of tests
- Visible wall graphs
  - The collected metrics are daily updated on wall graphs for all to easily see - easily communicate.

Tracking

Tracking and Daily Tracker

- The regular collection of task and story progress metrics is the responsibility of a tracker.
- This is done with a walk-about to all the programmers, rather than email.
  - Ron Jeffries (one of the XP founders) said, "*XP is about people, not computers*."
- Test metrics can be automatically collected by software.

Common Project Room

Common project room

- XP projects are run in a common project room rather than separate offices.
  - Pair programming tables are in the center of the room, and the walls are clear for whiteboard and poster work.
  - Of course, people may have a private space for private time, but production software development is a team sport in XP.

Daily stand-up meeting

- As in Scrum, there is a daily short stand-up (to keep it short) meeting of status.

Agile Project Room

The project room walls are exposed (no furniture against them) and covered in whiteboards and static-cling-sheet whiteboard material.

Ideal Engineering Hours (IEH)

Ideal Engineering Hours (IEH)

- Task estimates – and possibly story estimates – are done in terms of IEH, or uninterrupted, dedicated, focused time to complete a task.

Story estimates

- To estimate larger stories, some XP practitioners recommend using only coarse values of one, two, or three week durations rather than IEH or day-level estimates.

Sample XP Projects

The following projects had significant XP influence:

- Large – Atlas leasing system
  - Three years, 60+ people, Java technologies, an E100 project [Schuh01]
- Medium – Orca security incident-response
  - One year, 25 people, a D40 project [Morales02]
- Small – C3 payroll
  - One year, 10+ people, an E20 project [C3Team98]

Adoption Strategies

XP recommends adoption like this:

1. Pick the worst project or problem.
2. Apply XP until solved.
3. Repeat.

Adoption Strategies

If all the XP practices cannot be swallowed at once, Beck recommends starting with:

- whole team together in a common project room
- test-first development
- acceptance tests written/owned by customers
- Planning Game
- pair programming

History

- In the mid-1980s, Kent Beck and Ward Cunningham worked together in a research group at Tektronix.
  - They founded the idea of CRC cards and (the seminal contribution of) design patterns, while building many Smalltalk systems.
  - The roots of XP come from this collaboration.

Class:	
Responsibilities:	Collaborators:

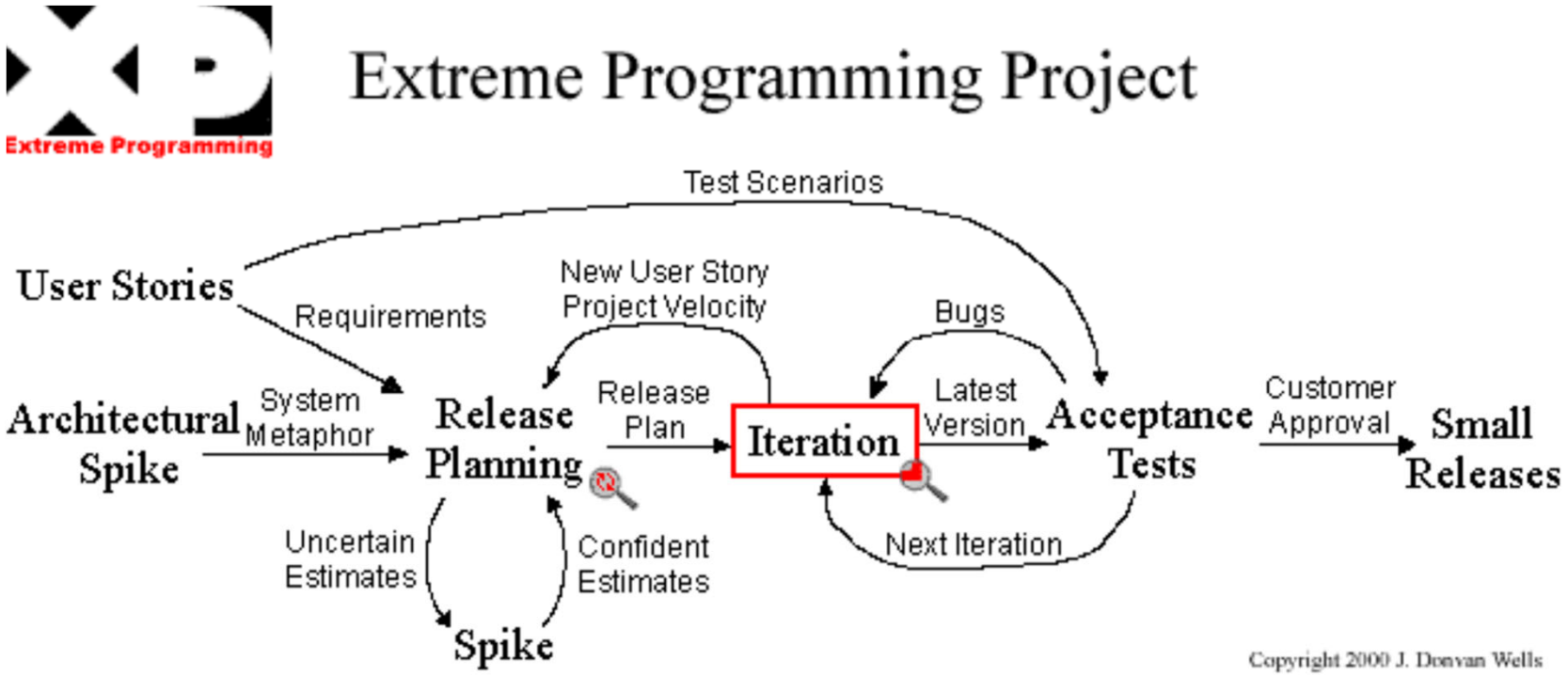
History

- In the C3 Project of Chrysler, Beck introduced the majority of practices that became XP, and brought in Ron Jeffries to daily lead and coach the team.
- Martin Fowler was also invited for some consulting.
- Primarily led by the vision of Beck, the XP practices coalesced on this project.
- Beck says that at its heart, XP is expressing what he learned with and from Cunningham.

Sample XP Practices

Don Wells: XP Rules

http://www.extremeprogramming.org/rules.html



Sample XP Practices

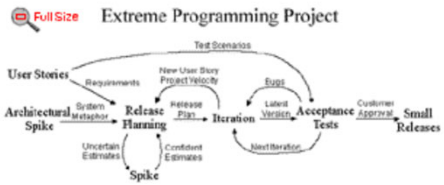
Don Wells: XP Rules

Extreme Programming

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.

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

References

- Craig Larman. Agile & Iterative Development: A Manager's Guide. Addison-Wesley, Pearson Education, 2004. (11th Printing, Aug. 2009) (ISBN-10: 0-13-111155-8, ISBN-13: 978-0-13-111155-4)
- Agile Alliance: http://www.agilealliance.org/
  - http://www.agilemanifesto.org/
  - http://www.agilemanifesto.org/principles.html
  - http://guide.agilealliance.org/subway.html
- Agile Modeling and RUP – Scott Ambler: http://www.agilemodeling.com/
- Agile, XP Method – Martin Fowler: http://www.martinfowler.com/
- Agile, XP Method – Robert Martin: http://www.objectmentor.com/
- Extreme Programming – Don Well: http://www.extremeprogramming.org/
- Extreme Programming – Ron Jeffries: http://xprogramming.com/index.php
- Agile Development – Artern Marchenko : http://agilesoftwaredevelopment.com/