CSE4708: Software Project Management

Unit II : Project Evaluation & Estimation

Topic: Process Models

Name: Manka Sharma

Delivered on: 2nd September 2020

Agile Developmental Methodology

Extreme Programming (XP)

- It is based on four core values: communication, simplicity, feedback, courage and respect.
- First, the developers must provide rapid feedback to the end users on the continuous basis.
- Second, the developers must follow simplicity principle.
- The developers must make the incremental changes to grow the system.
- The quality-first mentality should be obeyed.

Extreme Programming (XP)

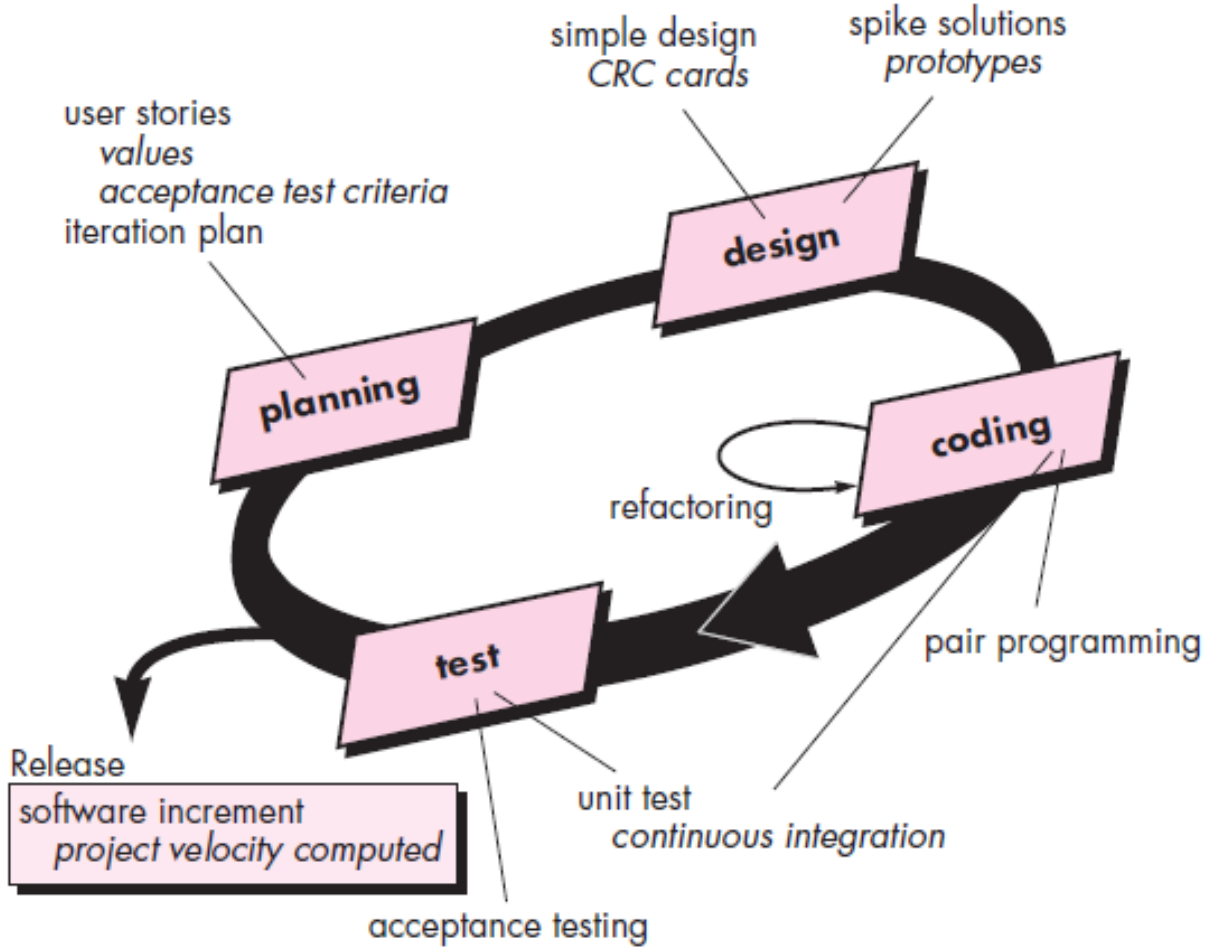
The XP Process

- Extreme Programming uses an object-oriented approach.
- its preferred development paradigm and a set of rules and practices that occur within the context of four framework activities:
  - planning,
  - design,
  - coding, and
  - testing.

Extreme Programming (XP)

FIGURE 3.2

The Extreme Programming process



Extreme Programming (XP)

Planning: The planning activity begins with

- listening—a requirements gathering activity that enables the technical members of the XP team to understand the business context for the software and to get a broad feel for required output and major features and functionality.
- Listening leads to the creation of a set of "stories", also called *user stories* that describe required output, features, and functionality for software to be built.
- Each *story* is written by the customer and is placed on an index card.

Extreme Programming (XP)

- The customer assigns a *value* (i.e., a priority) to the story based on the overall business value of the feature or function.
- Members of the XP team then assess each story and assign a *cost*—measured in development weeks—to it.
- If the story is estimated to require more than three development weeks, the customer is asked to split the story into smaller stories and the assignment of value and cost occurs again.
- It is important to note that new stories can be written at any time

Extreme Programming (XP)

- Customers and developers work together to decide how to group stories into the next release.
- Once a basic *commitment* i.e. agreement on stories to be included, delivery date, and other project matters is made for a release, the XP team orders the stories that will be developed in one of three ways:
  - all stories will be implemented immediately
  - the stories with highest value will be moved up in the schedule and implemented first or
  - the riskiest stories will be moved up in the schedule and implemented first.

Extreme Programming (XP)

- After the first project release, also called a software increment, has been delivered the XP team computes project velocity.
- Stated simply, *project velocity* is the number of customer stories implemented during the first release.
- Project velocity can then be used to
  - help estimate delivery dates and schedule for subsequent releases and
  - determine whether an over commitment has been made for all stories across the entire development project.
- If an over commitment occurs, the content of releases is modified or end delivery dates are changed.

Extreme Programming (XP)

- As development work proceeds, the customer can add stories, change the value of an existing story, split stories, or eliminate them.
- The XP team then reconsiders all remaining releases and modifies its plans accordingly.

Extreme Programming (XP)

Design:

- XP design rigorously follows the keep it simple principle.
- A simple design is always preferred over a more complex representation.
- In addition, the design provides implementation guidance for a story as it is written—nothing less, nothing more.
- The design of extra functionality is discouraged.
- XP encourages the use of CRC cards as an effective mechanism for thinking about the software in an object-oriented context.
- CRC (class-responsibility collaborator) cards identify and organize the object-oriented classes that are relevant to the current software increment.

Extreme Programming (XP)

- If a difficult design problem is encountered as part of the design of a story, XP recommends the immediate creation of an operational prototype of that portion of the design.
- Called a *spike solution*, the design prototype is implemented and evaluated.
- The intent is to lower risk when true implementation starts and to validate the original estimates for the story containing the design problem.

Extreme Programming (XP)

- XP encourages *refactoring*—a construction technique that is also a method for design optimization.
- Refactoring is the process of changing a software system in such a way that it does not alter the external behavior of the code yet improves the internal structure.
- It is a disciplined way to clean up code [and modify/simplify the internal design] that minimizes the chances of introducing bugs.
- In essence, refactoring is improving the design of the code after it has been written.

Extreme Programming (XP)

- XP design uses virtually no notation and produces few, if any, work products other than CRC cards and spike solutions, design is viewed as a transient artifact that can and should be continually modified as construction proceeds.
- The intent of refactoring is to control these modifications by suggesting small design changes that "can radically improve the design".
- It should be noted, however, that the effort required for refactoring can grow dramatically as the size of an application grows.

Extreme Programming (XP)

Coding

- After stories are developed and preliminary design work is done, the team does *not* move to code, but rather develops a series of unit tests that will exercise each of the stories that is to be included in the current release.
- Once the unit test has been created, the developer is better able to focus on what must be implemented to pass the test.
- Nothing extraneous is added. Once the code is complete, it can be unit-tested immediately, thereby providing instantaneous feedback to the developers.

Extreme Programming (XP)

- A key concept during the coding activity is *pair programming*.
- XP recommends that two people work together at one computer workstation to create code for a story.
- This provides a mechanism for real time problem solving and real-time quality assurance.
- It also keeps the developers focused on the problem at hand.
- In practice, each person takes on a slightly different role.
- For example, one person might think about the coding details of a particular portion of the design while the other ensures that coding standards are being followed or that the code for the story will satisfy the unit test that has been developed to validate the code against the story

Extreme Programming (XP)

- As pair programmers complete their work, the code they develop is integrated with the work of others.
- In some cases this is performed on a daily basis by an integration team.
- In other cases, the pair programmers have integration responsibility.
- This "continuous integration" strategy helps to avoid compatibility and interfacing problems and provides a "smoke testing" environment that helps to uncover errors early.

Extreme Programming (XP)

Testing:

- The creation of unit tests before coding commences is a key element of the XP approach.
- The unit tests that are created should be implemented using a framework that enables them to be automated and hence, they can be executed easily and repeatedly.
- This encourages a regression testing strategy whenever code is modified.

Extreme Programming (XP)

- As the individual unit tests are organized into a "universal testing suite", integration and validation testing of the system can occur on a daily basis.
- This provides the XP team with a continual indication of progress and also can raise warning flags early if things go awry.
- XP *acceptance tests*, also called *customer tests*, are specified by the customer and focus on overall system features and functionality that are visible and reviewable by the customer.
- Acceptance tests are derived from user stories that have been implemented as part of a software release.

Advantages and Disadvantages

- Advantages :
  - Decrease the time required to avail some system features.
  - Face to face communication and continuous inputs from customer representative leaves no space for guesswork.
  - The end result is the high-quality software in the least possible time duration and satisfied customer.
- Disadvantages:
  - The ability and collaboration of the customer to express user needs.
  - Documentation is done at later stages.
  - Reduce the usability of components.
  - Needs special skills for the team.

Note : Students must go through 'XP debate' in the shared document.

Recommended Reading

- Pressman, Roger S., “Software Engineering – A practitioner’s Approach”, “Chapter -3: Agile Development”, 7th edition, pp. 65-93.
- Students to explore the other methodologies of Agile Development.

References

- Pressman, Roger S., “Software Engineering – A practitioner’s Approach”, “Chapter -3: Agile Development”, 7th edition, pp. 65-93.