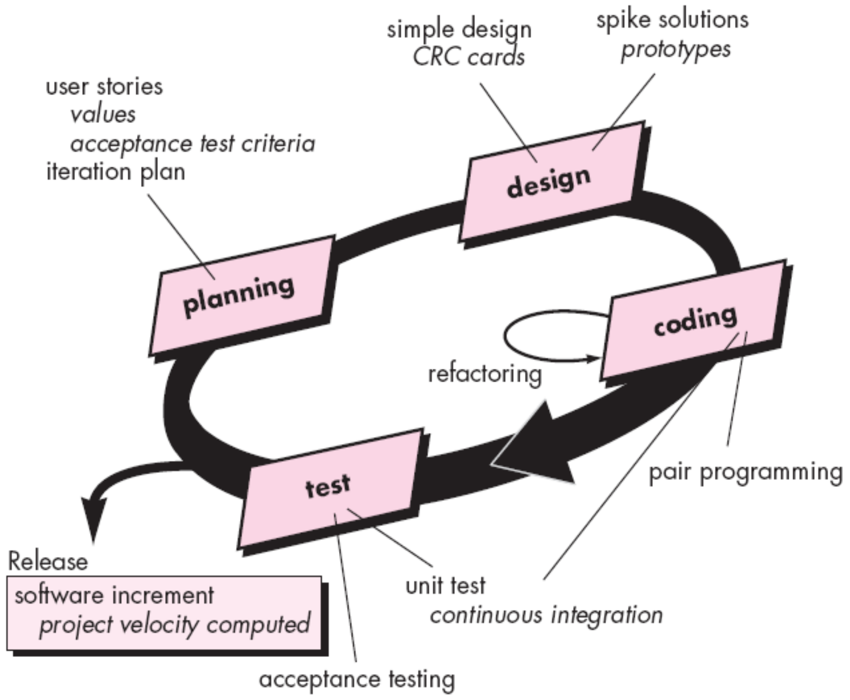
EXTREME PROGRAMMING:

In order to illustrate an agile process in a bit more detail, we'll provide you an overview of Extreme Programming (XP), the most widely used approach to agile software development.

Figure below illustrates the XP process and notes some of the key ideas and tasks that are associated with each framework activity. XP activities are summarized in the paragraphs that follow.

Planning. The planning activity (also called the planning game) 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. The customer assigns a value (i.e., a priority) to the story. 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.

The XP team orders the stories that will be developed in one of three ways:

(1) all stories will be implemented immediately (within a few weeks),

(2) the stories with highest value will be moved up in the schedule and implemented first, or

(3) the riskiest stories will be moved up in the schedule and implemented first.

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:

(1) help estimate delivery dates and schedule for subsequent releases and

(2) 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.

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.

Design. XP design follows the KIS (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.

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.

In the preceding section, we noted that 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 way to clean up code [and modify/simplify the internal design] that minimizes the chances of introducing bugs. In essence, when you refactor, you are improving the design of the code after it has been written.

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 (software increment).

Once the unit test has been created, the developer is better able to focus on what must be implemented to pass the test. Once the code is complete, it can be unit-tested immediately.

A key concept during the coding activity (and one of the most talked about aspects of XP) 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 (two heads are often better than one) and real-time quality assurance (the code is reviewed as it is created). 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.

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.

Testing. You have already noted that 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.