CMSC 128 Laboratory Handout 3

Software Development Cycle and Agile Development Model

Stages of Software Development Phase

- **Analysis** – specifies what services the proposed system is to provide and to identify any conditions (time constraint, security) on those services
- **Design**
  - concentrates on how the system will accomplish the goals
  - establishes the structure of the software system
  - creates user interface
- **Implementation** – involves actual writing of programs, creation of datafiles
- **Testing** – occurs in two forms
  - **Validation Testing** – confirming that the software system as implemented conforms to the requirements and specifications identified during the original analysis
  - **Defect Testing** – identifying and correcting errors

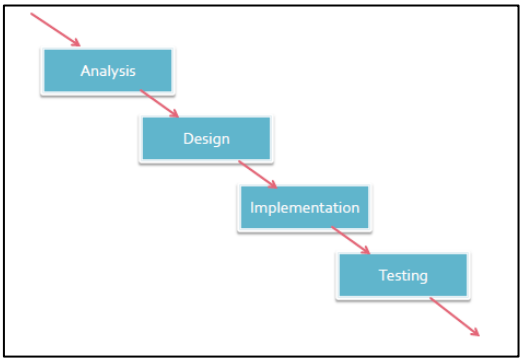

Figure 1. Software Development Phase

Some Software Development Process Models

1. Waterfall Method

- sometimes called the *classic life cycle*
- suggests a systematic sequential approach to software development that begins with customer specification of requirements and progress through planning, modeling, construction, and deployment, culminating in ongoing support of the completed software

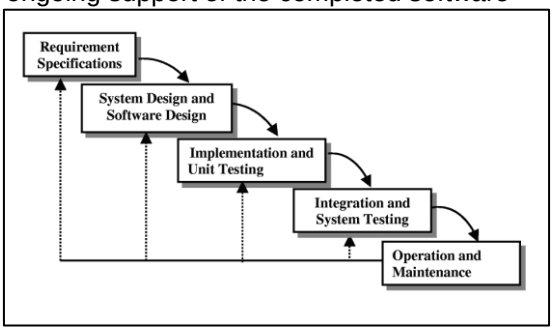

Figure 2. The Waterfall Model

2. Incremental Process Models

- applies linear sequences in a staggered fashion as calendar time progresses
- each linear sequence produces deliverable "increments" of the software in a manner that is similar to increments produced by an evolutionary flow

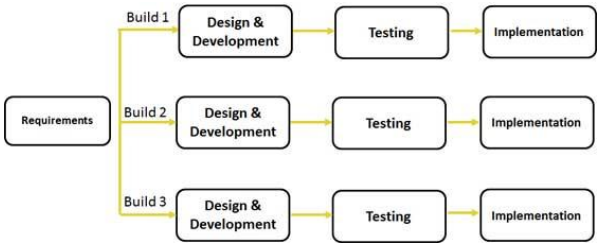

Figure 3. The Iterative Model

3. Evolutionary Process Models

- are iterative
- characterized in a manner that enables development of increasingly more complete versions of the software

a. *Prototyping* – can be used as a stand-alone process model

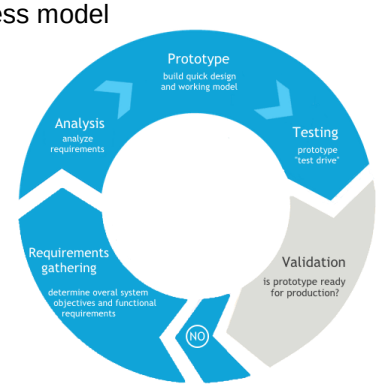

Figure 4. Prototyping

b. *Spiral Model* – risk-driven process model generator that is used to guide multi-stakeholder concurrent engineering of software intensive systems

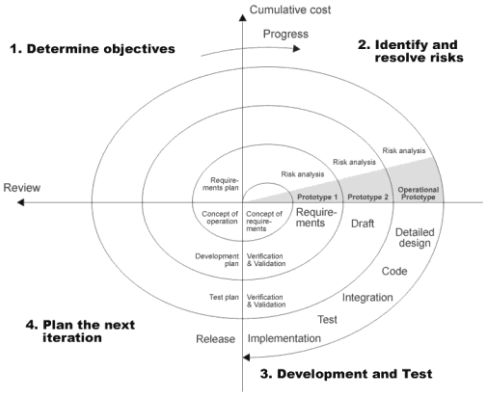

Figure 5. Spiral Modeling

Agile Software Development

- characterized in a manner that addresses a number of key assumptions about the majority of software process:
  1) It is difficult to predict in advance which software requirements will persist and which will change. It is equally difficult to predict how customer priorities will change as the project proceeds.
  2) For many types of software, design and construction are *interleaved*. It is difficult to predict how much design is necessary before construction is used to prove the design.

3) Analysis, design, construction and testing are not as predictable (from a planning point of view) as we might like.

Refactoring – allows a software engineer to improve the external structure of a design (or source code) without changing its external functionality or behavior

Some Approaches in Agile Software Development

- **Extreme Programming (XP)**
  - Five XP Values: communication, simplicity, feedback, courage and respect

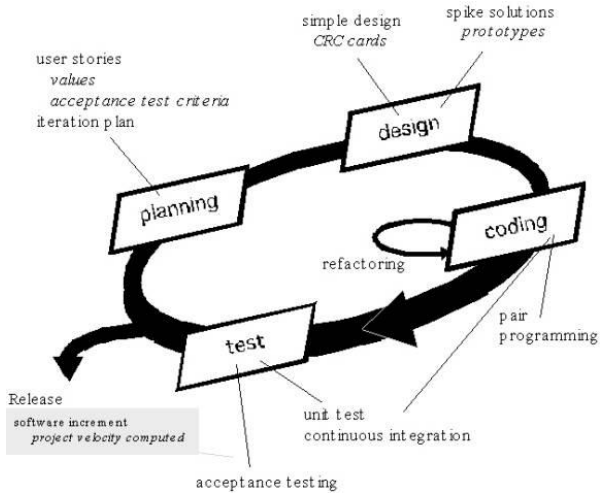

Figure 6. The XP Model

- **Scrum**
  - its principles are consistent with the agile manifesto and are used to guide development activities within a process that incorporates the following framework activities: requirements, analysis, design, evolution and delivery

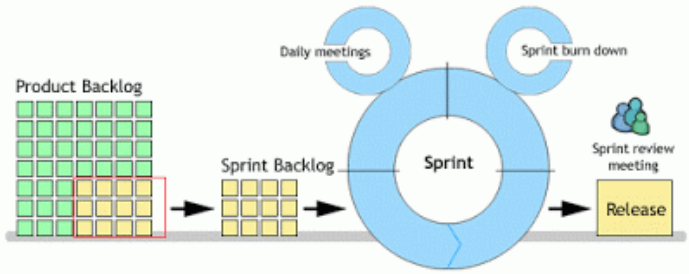

Figure 7. Scrum Process Flow

Some Important Scrum Terminologies

- Product Backlog
  - a prioritized list of project requirements (user stories) or features that provide business value for the customer
  - items can be added to the backlog at any time
  - project manager assesses the backlog and updates priorities as required
- Release Backlog
  - user stories for a certain release
  - prioritize, then estimate time for each feature (larger user stories can be broken-down into smaller chunks)
- Sprint Backlog – divided release backlogs
- Sprints
  - consist of work units that are required to achieve a requirement defined in the backlog that must be fit into a predefined timebox
  - *changes are not introduced* during the sprint
- Burndown chart – provides a day-by-day measure of the amount of work that remains in a given sprint or release
- Scrum meetings
  - are short meetings held daily by the Scrum team
  - all team members answers three key questions:
    1) What did you do since the last meeting?
    2) What obstacles are you encountering?
    3) What do you plan to accomplish by the next meeting?
- Product Owner – checks if the right features are in the product
- Scrum master
  - the team leader
  - leads meetings and assesses the responses from each person
- Demos – deliver the software increment to the customer so that functionality that has been implemented can be demonstrated and evaluated by the customer

Images from:

- http://adamjonescomputingscience.blogspot.com/2014/05/software-development-life-cycle.html
- http://www.broodware.com/software-prototyping/
- http://csis.pace.edu/~marchese/cs615sp/L2New/CS615l2n.htm
- http://www.testertroubles.com/2009/03/agile-scrum-process-model.html
- http://www.netspecglobal.com/approach.html

References

- Axosoft. (2012). NEW Intro to Agile Scrum in Under 10 Minutes - What is Scrum [Video file]. Retrieved from https://www.youtube.com/watch?v=XU0llRltyFM
- De Robles, M.Y.B. *CMSC 11 Slides (Software Engineering, Databases & Networks)*. 1st Semester, 2013-2014.
- Hunt, J. (2006). *Agile Software Construction*. 1st Edition. London, UK: Springer-Verlag.
- Pressman, R. S. (2010). *Software Engineering a Practitioner's Approach (Alternate Edition)*. 7th Edition. New York, NY: McGraw-Hill.

Figure 8. More detailed Scrum Process Flow