SCHOOL OF COMPUTER SCIENCE AND IT

School of Computer Science & IT

Department of BCA

SOFTWARE ENGINEERING (22BCA3C01)

MODULE 1: Software Product and Process

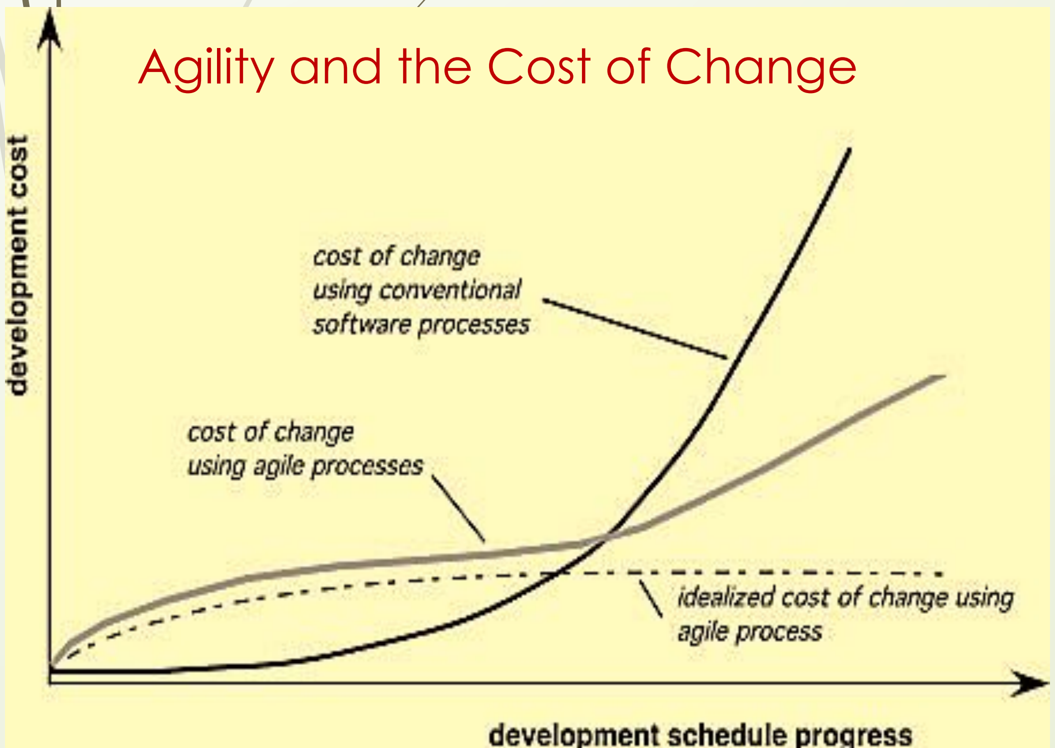
Session -7

- Process Models / Life Cycle Models
  - Agile Software Development

- Human Aspects of Software Engineering

Change and Agility in Software Development

- Change is very much inevitable in software development. Change can be due to
  - changes in the requirements of the software being built,
  - changes to the team members,
  - changes because of new technology,
- Changes of all kinds may have an impact on the software product or the project that creates the product. Changes escalates the cost quickly as the project progresses.
- So, support for changes should be embraced into the development process, because it is the heart and soul of software. Agility in software development process facilitates it.



Agility and the Cost of Change

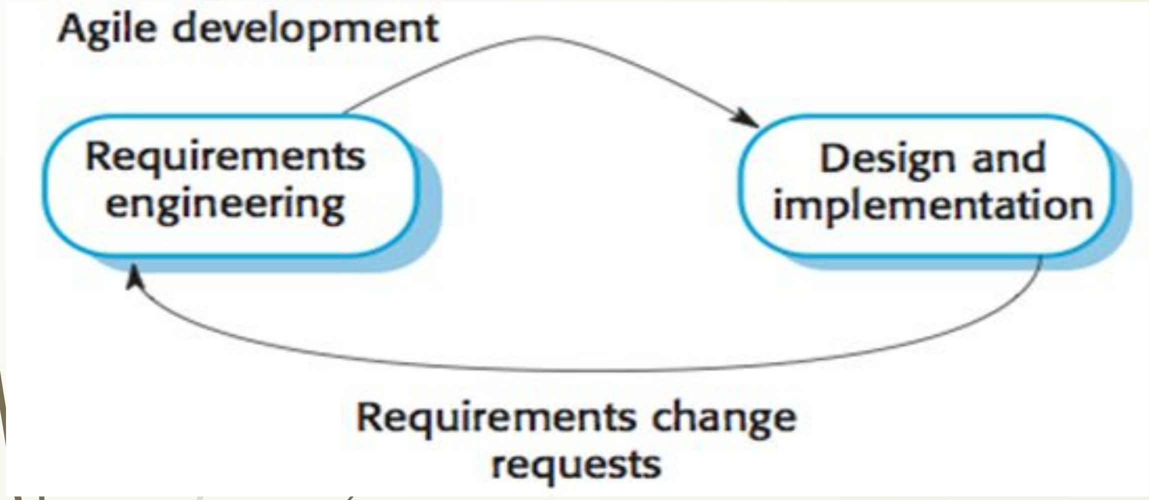
❑What is Agility?

- Effective (rapid and adaptive) response to change
- Effective communication among all stakeholders
- Drawing the customer onto the team
- Organizing a team so that it is in control of the work performed, no escalation of time and cost

Yielding …

- Rapid, incremental delivery of software

Agile Process / Agile Development



- ✔ **Agile software process** is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between development team and users.
- ✔ Dissatisfaction with the overheads involved in software design methods of the 1980s and 1990s led to the creation of agile methods.

❑ **Agile development methods** emerged in the early 2000 whose aim was to radically reduce the delivery time for working software systems:

- Are based on an **iterative approach** to software development.
- Are intended to deliver working software quickly and evolve this quickly to **meet changing requirements**.
- Program specification, design, and implementation are **interleaved**
- The system is developed as a series of frequent versions or **increments**
- **Stakeholders** involved in version specification and evaluation
- Extensive **tool support** (e.g. automated testing tools) used to support development
- **Minimal documentation** - focus on working code

The Values and Principles of the Agile Manifesto

In 2001, values and principles were introduced by Agile Alliance as part of Agile Manifesto that are considered as the guiding principles for Agile development.

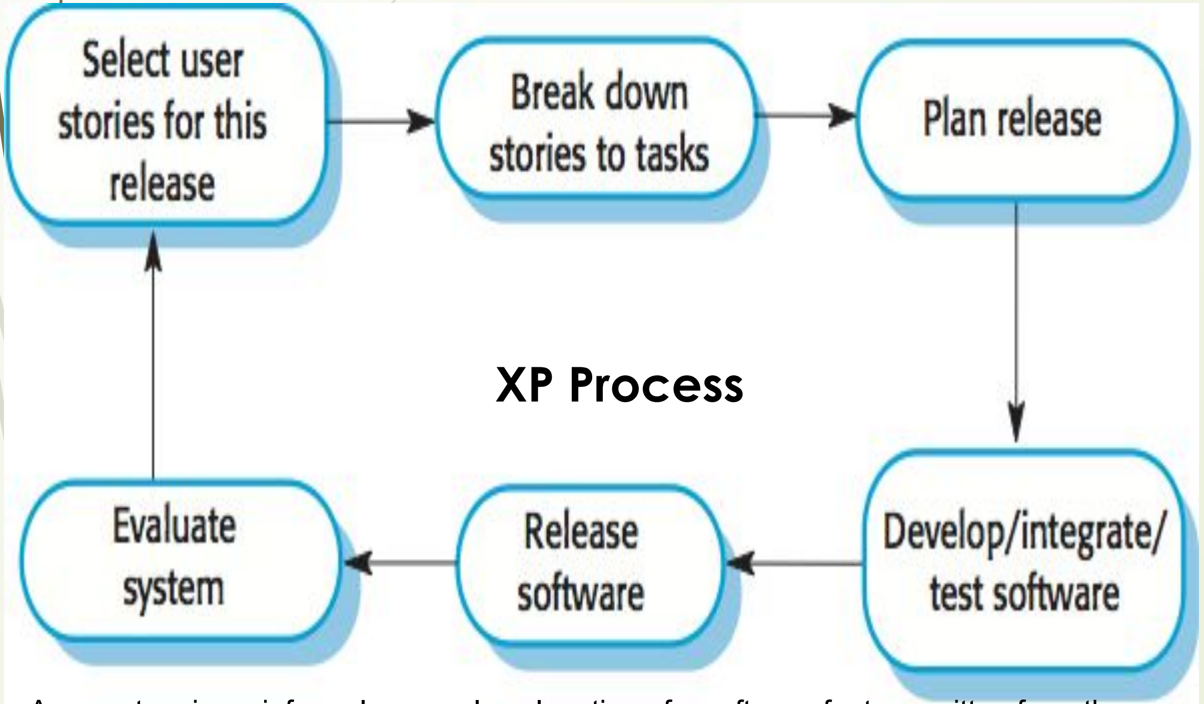
- **Values**:
  - ✓ Individuals and interactions over processes and tools
  - ✓ Working software over comprehensive documentation
  - ✓ Customer collaboration over contract negotiation
  - ✓ Responding to change over following a plan
- The **principles** of agile methods: There are 12 Principles summarized as:
  - **Customer involvement :** Customers should be closely involved throughout the development process. Their role is to 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.

Agility Principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face–to–face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity – the art of maximizing the amount of work not done – is essential.
11. The best architectures, requirements, and designs emerge from self–organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile Method: Extreme Programming

- One well-known agile method, Extreme Programming (XP) takes an 'extreme' approach to iterative development:
  - New versions may be built several times per day;
  - Increments are delivered to customers every 2 weeks;
  - All tests must be run for every build and the build is only accepted if tests run successfully.


Select user stories for this release
Break down stories to tasks
Plan release
XP Process
Develop/integrate/ test software
Release software
Evaluate system

A user story is an informal, general explanation of a software feature written from the perspective of the end user or customer.

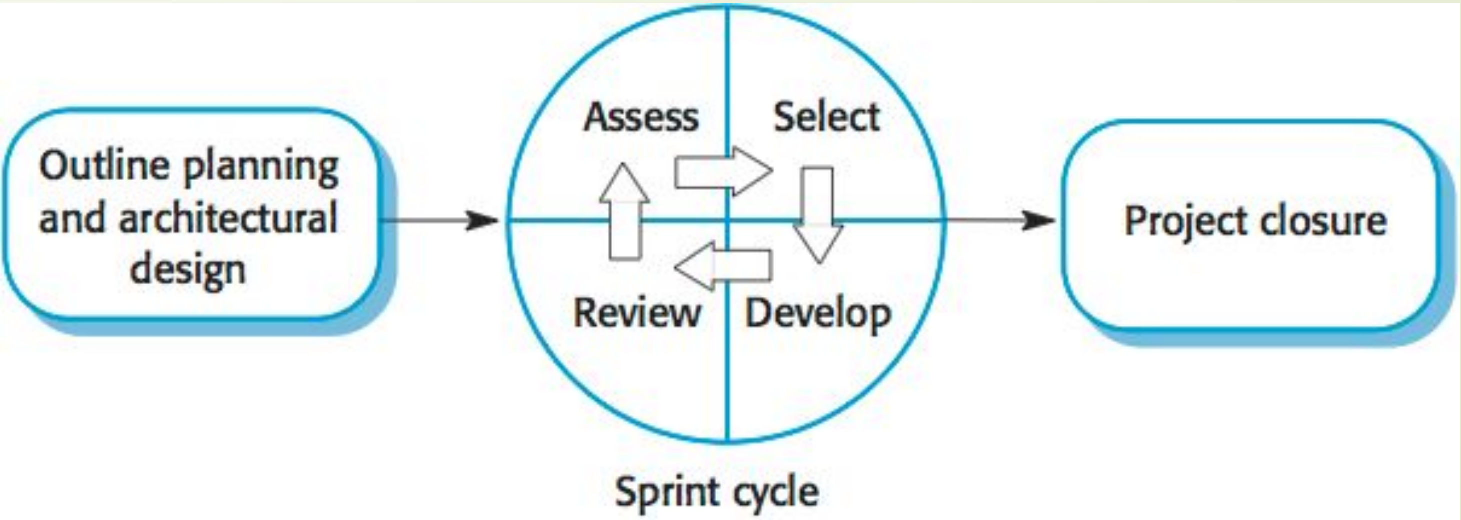
- This is how XP supports **agile principles**:
  - Customer involvement means **full-time customer engagement** with the team.
  - Incremental development is supported through **small, frequent system releases**.
  - People not process through **pair programming** (one programmer write code, the other reviews; both switch roles regularly), **collective ownership**, and a process that avoids long working hours.
  - Change supported through **regular system releases**.
  - Maintaining simplicity through **constant refactoring** of code.

XP Process: Begin with “user stories”. Group stories for a deliverable increment, Make commitment on delivery date. Design, develop, test and deliver the software with acceptance testing.

Agile Method: Scrum

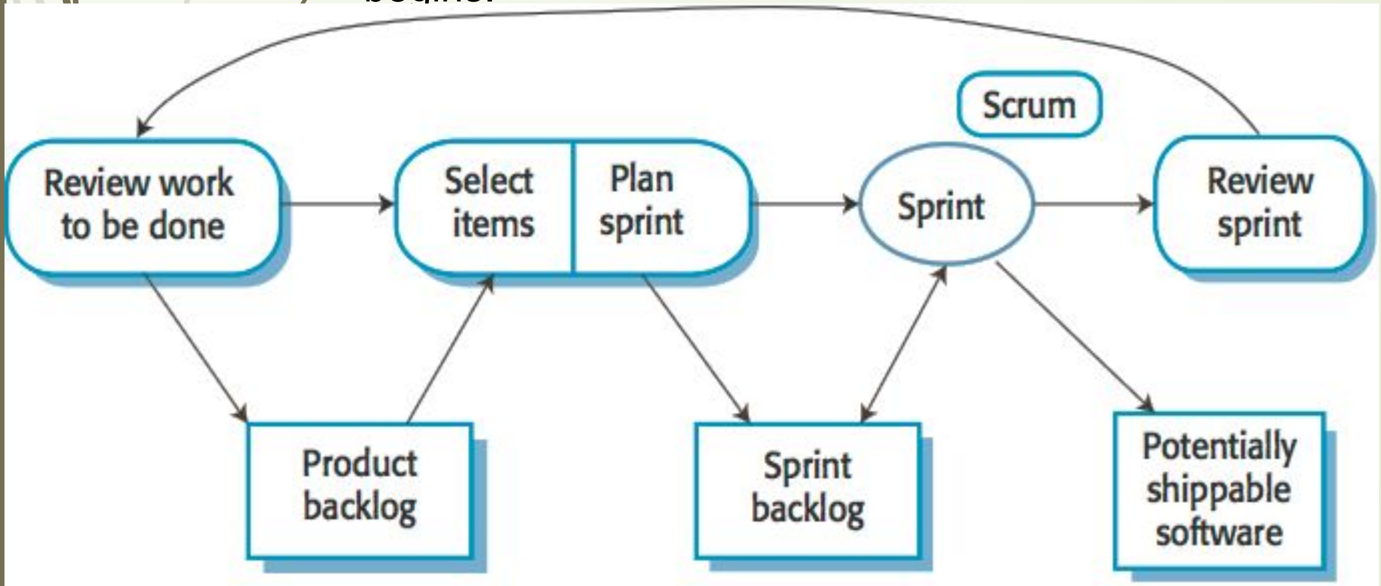
- The Scrum approach is a general agile method but its focus is on managing iterative development rather than specific agile practices. There are three phases in Scrum:

1. The initial phase is an outline planning phase where you establish the general objectives for the project and design the software architecture.
2. This is followed by a series of **sprint** cycles, where each cycle develops an increment of the system.
3. The project closure phase wraps up the project, completes required documentation such as system help menu and user manuals and assesses the lessons learned from the project.



Sprint of Scrum

- Sprints are fixed length, normally 2-4 weeks. The starting point for planning is the **product backlog**, which is the list of work to be done on the project. The selection phase involves all of the project team who work with the customer (**product owner**) to select the features and functionality to be developed during the sprint.
- Once these are agreed, the team organize themselves to develop the software. During this stage the team is relatively isolated from the product owner and the organization, with all communications channeled through the **ScrumMaster**. The role of the ScrumMaster is to arrange daily meetings (daily scrums) tracks the backlog of work to be done, records decisions, measures progress against the backlog and communicates with the product owner and management outside of the team.
- At the end of the sprint, the work done is reviewed and presented to stakeholders (including the product owner). **Velocity** (i.e. amount of work) is calculated during the **sprint review**; it provides an estimate of how much product backlog the team can cover in a single sprint. Understanding the team's velocity helps them estimate what can be covered in a sprint and provides a basis for measuring and improving performance. The next sprint cycle then begins.



Advantages:

- The product is broken down into a set of **manageable and understandable chunks**.
- Unstable requirements do not hold up **progress**.
- The whole team have visibility of everything and consequently **team communication** is improved.
- Customers see **on-time delivery** of increments and gain feedback on how the product works.
- **Trust** between customers and developers is established and a positive culture is created in which everyone expects the project to succeed.

Characteristics of a Software Engineer

So you want to be a software engineer? Obviously, you have to master the technical stuff, learn and apply the skills required to understand the problem, design an effective solution, build the software, and test it in an effort to develop the highest quality possible. You have to manage change, communicate with stakeholders, and use appropriate tools in the appropriate situations.

- But there are other things that are equally important—the human aspects that will make you an effective software engineer.
- There are seven traits that can make an individual software engineer exhibits “super professional” behavior.
- **Sense of individual responsibility:**
  - This implies a drive to deliver on his promises to peers, stakeholders, and his management. It implies that he will do what needs to be done, when it needs to be done in an overriding effort to achieve a successful outcome.
- **Acutely aware of the needs of team members and stakeholders:**
  - It enables him to work in groups and adapt his behavior when needed.
- **Brutally honest about design flaws and offers constructive criticism**
  - Should not distort facts on schedule, feature, performance, or other characteristics of the product, project. Should be trustful.
- **Resilient under pressure**
  - Pressure comes in many forms—changes in requirements and priorities, demanding stakeholders or peers, an unrealistic or overbearing manager. He should be able to manage the pressure so that his performance does not suffer.
- **Heightened sense of fairness**
  - Gladly share credit with colleagues. Try to avoid conflicts of interest and never acts to sabotage the work of others.
- **Attention to detail**
  - carefully considers (or understand) the technical decisions (or works) fully in detail and achieve perfection.
- **Pragmatic**
  - software engineering is not a religion in which dogmatic rules must be followed, but rather a discipline that can be adapted based on the circumstances at hand.

Effective Software Team Attributes

- A jelled team is a group of people so strongly knit that the whole is greater than the sum of the parts . . . .
- Once a team begins to jell, the probability of success goes way up. The team can become unstoppable, a juggernaut for success . . . . They don't need to be managed in the traditional way, and they certainly don't need to be motivated. They've got momentum.
- There is no foolproof method for creating a jelled team. But there are attributes that are normally found in effective software teams.
- Sense of purpose
  - All team members should strongly agree with the goal of the team.
- Sense of involvement
  - Every member should feel that his skill set and contributions are valued, and contribute fully.
- Sense of trust
  - Software engineers on the team should trust the skills and competence of their peers and their managers.
- Sense of improvement
  - Team members should periodically improve its approach to software engineering and looking for ways to improve their work
- Diversity of team member skill sets
  - The most effective software teams are diverse in skill sets.

Quiz Questions

1) What is agile software development?
2) What is agile manifesto?
3) What are principles of agile development?
4) What is extreme programming?
5) What is scrum?
6) What is a sprint in scrum?
7) What are the advantages of agile development?

Next class: Flip Class:
Software Development Life Cycle (SDLC) 1 hr 34m
https://www.linkedin.com/learning/software-development-life-cycle-sdlc/processes-for-software-projects?u=92695330

THANK YOU

Any questions…?