Lecture Notes for Lecture 3 of CS 5500 (Foundations of Software Engineering) for the Spring 2021 session at the Northeastern University San Francisco Bay Area Campuses.

Focus on Agile Methodologies

Philip Gust,
Clinical Instructor
Department of Computer Science

http://www.ccis.northeastern.edu/home/pgust/classes/cs5500/2021/Spring/index.html

Focus on Agile Methodologies

Review of Lecture 2

- In lecture 2 we began to study specific types of software engineering process models that have been developed to guide product development.
- For each process model, we sought to
  - understand the underlying principles of the model
  - look at how it fits in to the generic software process framework
  - see how the process works
  - consider its relative strengths and weaknesses

Focus on Agile Methodologies

Review of Lecture 2

- We looked at the following historical process models.
  - Code and fix (1950s – 1960s)
    - no underlying engineering model
  - Waterfall model (1970s – 1980s)
    - original engineering-based model, proposed as “strawman”
  - Incremental models (1970s – 1980s)
    - multiple waterfall cycles, fixed requirements
  - Evolutionary models (1970s – 1980s)
    - multiple waterfall cycles, evolving requirements
  - Evolutionary models with prototyping (1980s – 1990s)
    - Cyclic model includes continuous refinement
  - Spiral framework (c. 1986)
    - Evolutionary model sought to minimize risk at each turn

Focus on Agile Methodologies

Introduction

- We have already seen several evolutionary software engineering models, where requirements as well as process can evolve. In this lecture, we will focus on another.
- In an *Agile* software engineering approach, requirements and solutions evolve through collaboration with self-organizing, cross-functional teams and their stakeholders and customers.
- The focus is on customer satisfaction, active and continuous communication, incremental delivery, informal methods, minimal work products and flexible response to change.
- The only really important work product is an operational ”software increment” delivered to the costumer on the appropriate commitment date.

Focus on Agile Methodologies

Introduction

- With Agile software development, software engineers, managers, customers, and stakeholders work closely together as an agile team whose members' interests are aligned.
- Agile software development is a reasonable alternative to traditional software engineering for certain types of software projects.
- While there is much anecdotal evidence that Agile practices improves agility of professionals, teams, and organizations, some empirical studies have also disputed that evidence.

Focus on Agile Methodologies

History

- Adaptive software engineering processes began to emerge in the 1970s. By the 1990s, several “lightweight” processes models evolved in reaction to past “heavyweight” models:
  - Rapid Application Development (RAD) – 1991
  - Unified Process (UP) – 1994
  - Scrum – 1995
  - Extreme Programming (XP) -- 1996

Focus on Agile Methodologies

History

- In 2001, seventeen software engineers who were responsible for creating these lightweight models met at a resort in Snowbird Utah to discuss lightweight development methods.
- As a result of this meeting, members of the group founded the *Agile Alliance* to promote lightweight methodologies, and published the *Manifesto for Agile Software Development*.

- Four years later, the group published an addendum of project management principles, the *Declaration of Interdependence* to guide project managers using agile development methods.

Focus on Agile Methodologies

History

- Signers of the Manifesto for Agile Software Development (2001).

Focus on Agile Methodologies

History

- "Manifesto for Agile Software Development" (2001)
  - We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
    - Individuals and interactions over processes and tools
    - Working software over comprehensive documentation
    - Customer collaboration over contract negotiation
    - Responding to change over following a plan
  - That is, while there is value in the items on the right, we value the items on the left more.

Focus on Agile Methodologies

Agile values

- The manifesto states preferences not alternatives and encourages focus on certain areas but not eliminating others.
  - Tools and processes are important, but it is more important to have competent people working together effectively.
  - Good documentation is useful to understand how the software is built and how to use it, but the main point of development is to create software, not documentation.
  - A contract with a customer is important but is no substitute for working closely with a customer to discover what they need.
  - A project plan is important, but it must accommodate changes in technology, environment, stakeholders' priorities, and people's understanding of the problem and its solution.

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

1. The highest priority is to satisfy the customer through early and continuous delivery of valuable software.
   - The goal of software development should be the development and delivery of software.
   - The evolutionary approach to development does not need to have everything defined and specified upfront.
   - Software construction is part of the evolving process of discovering, understanding and satisfying the customer's needs.

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

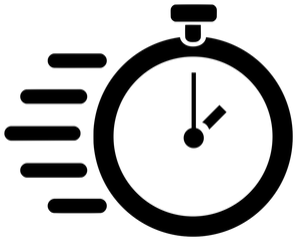
2. Welcome changing requirements, even late into development. Harness change for the customer's competitive advantage.
   - Requirements will change throughout a software project.
   - Traditional software developers often adopt change management processes which are designed to prevent/reduce scope creep
   - These are really change prevention processes, not change management processes.
   - Agile developers embrace change and instead follow an agile change management approach which treats requirements as a prioritized stack which is allowed to vary over time.

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

3. Deliver working software frequently, from weeks to a couple of months, with a preference for the shorter time scale.



   - Frequent delivery of working software provides stakeholders with concrete feedback.
   - This make the current status of your project transparent while at the providing an opportunity for stakeholders to provide improved direction for the development team.

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

4. Businesspeople and developers must work together daily throughout the project.

   - Your project is in serious trouble if you don't have regular access to customers and stakeholders.
   - Agile developers adopt practices such as on-site customer and active stakeholder participation.
   - They also adopt inclusive tools and techniques that enable stakeholders to be actively involved with software development.

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

5. Make projects on motivated individuals, give them the support and environment they need, and trust them to get the job done.

   - Hiring hordes of relatively unskilled people who have no stake in the outcome beyond pay is not the way to build quality software.
   - Agile organizations realize that teams are built on people who are willing to work together collaboratively and learn from each other.
   - They also have the humility to respect one another and realize that people are a primary success factor in software development.

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

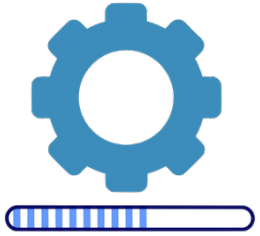
6. The most efficient, effective method to convey information to and within a development team is “face-to-face” conversation.
   - For a software development team to succeed its members must communicate and collaborate effectively.
   - There are many ways for people to communicate, but face-to-face communication at a shared drawing environment (such as paper or a whiteboard) is often the most effective way

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

7. Working software is the primary measure of progress.
   - The primary measure of software development is the delivery of working software that meets the changing needs of its stakeholders.
   - ”Earned value" measures based on the delivery of documentation or the holding of meetings provides no direct benefit to the customer.



Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

8. Agile processes promote sustainable development. Developers, stakeholders and end-users should be able to maintain a constant pace indefinitely.

   - You cannot successfully develop software by forcing people to work overtime for months at a time.
   - High-quality, intellectual work can be done for only 5-6 hours a day before burning out.
   - High-quality work for 12 hours a day can be done for a few days straight, but after that all you have is 12 hours of mediocre work.

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

9. Continuous attention to technical excellence and good design enhances agility.
   - It is much easier to understand, maintain, and evolve high-quality source code than it is to work with low-quality code.
   - Agile developers know that they need to start with good code, keep it good via refactoring, and take a test-driven approach so that they know at all times that their software works.
   - They also adopt and follow development guidelines, in particular coding guidelines and sometimes even modeling guidelines.

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

10. Simplicity – the art of maximizing the amount of work *not* done – is essential.
    - Agile developers focus on high value activities, strive to maximize stakeholder's return on investment, and either cut out or automate drudge work.

Focus on Agile Methodologies

Agile principles

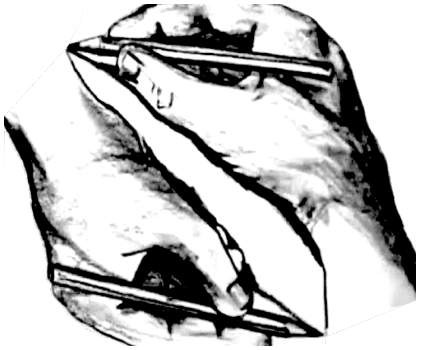
The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.

11. The best architectures, requirements, and designs emerge from self-organizing teams.
    - This is one of the most radical principles of the agile movement.
    - Agile Model Driven Development (AMDD) and Test-Driven Design (TDD) methods are the primary approaches for ensuring the emergence of effective architectures, requirements, and designs.

Focus on Agile Methodologies

Agile principles

The Agile Alliance refined the philosophies in their manifesto into a collection of twelve principles.



12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
    - Software process improvement (SPI) is a continual effort.
    - Techniques such as retrospectives should be adopted to enable improving your approach to software development.

Focus on Agile Methodologies

Agile principles

- Not every agile process model applies these twelve principles with equal weight, and some models choose to ignore (or downplay) the importance of one or more of the principles.
- However, these principles define an agile spirit that is maintained in each of the process models we will look at.

The Agile Manifesto

image: Restya.

Focus on Agile Methodologies

Agile results

- Agile methodology provides a philosophical foundation for software development based on what the manifesto sets out as a common-sense approach.
  - The goals are neither radical nor impossible for agile teams to achieve.
  - The focus is on customer value and incremental deliverables, rather than on process or intermediate work products.
  - It is a measured and responsive approach that is the way most teams would like projects to work but seldom do.

Focus on Agile Methodologies

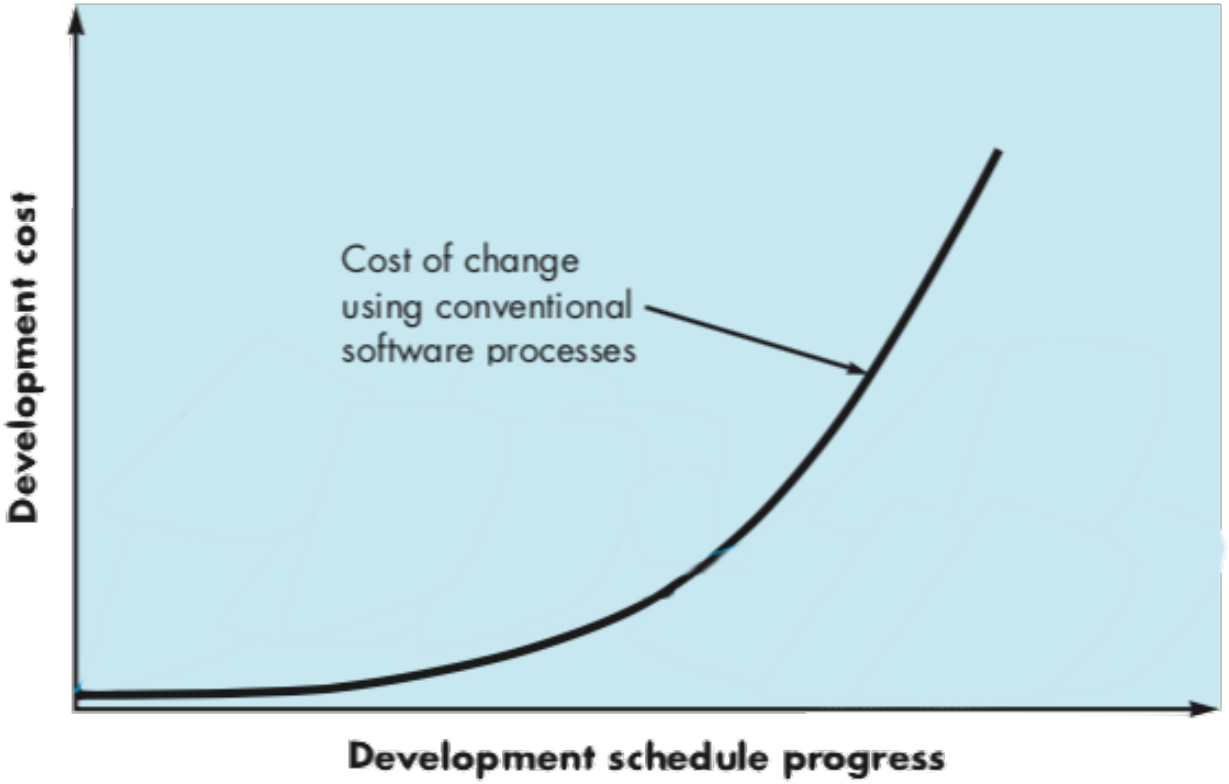
Agility and cost of change

- A common problem with traditional software methodologies is that the cost of a change in requirements increases non-linearly as the project progresses
- It is relatively easy to accommodate a change early in the process, when the software team is gathering requirements.
- What about when a stakeholder requests a major functional change relatively late in the project, requiring a change to the architectural design, plus new implementation and testing?
- Costs escalate, and the time and cost required to ensure there are no side-effects is non-trivial.

Focus on Agile Methodologies

Agility and cost of change

- Cost of change for conventional software process



Focus on Agile Methodologies

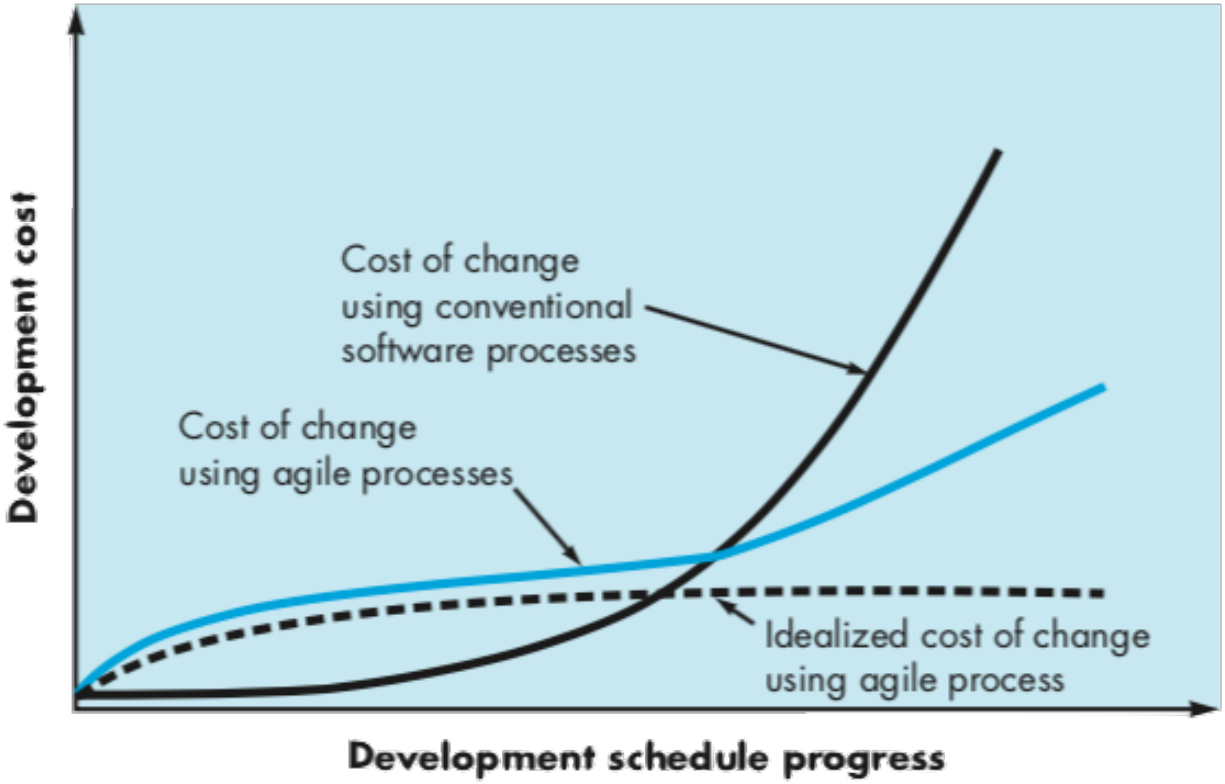
Agility and cost of change

- Agile processes are said to “flatten” the cost of change curve, allowing software teams to accommodate changes late in the process without dramatic cost impact.
- When incremental delivery is coupled with agile practices like continuous unit testing, the cost of changes is attenuated.
- Although there is debate about the degree to which the curve flattens, evidence suggests that significant cost reductions are possible.

Focus on Agile Methodologies

Agility and cost of change

- Cost of change for agile software process (ideal and actual)



Focus on Agile Methodologies

Agility and risk

- Agile processes rely on continual adaptation of product requirements as conditions change.
- To accomplish this, an agile team requires frequent and accurate costumer and stakeholder feedback. Otherwise, continual adaptation does not result in forward progress.
- If customers and stakeholders are not fully engaged as members of the product team, the risk is that feedback will not take place often enough to be useful.

Focus on Agile Methodologies

Agility and risk

- Resolving this risk means keeping customers and stakeholders engaged using an incremental development strategy.
- *Software increments* must be delivered in short time periods so that adaption stays in sync with changing external conditions and customer requirements.
- This iterative approach encourages customers to stay engaged as they evaluate software increments regularly, provide feedback, and influence process adaptations.

Focus on Agile Methodologies

Agility at scale

- Agile software development is widely seen as suited to certain types of environments, including small teams of experts working on “greenfield” projects.
- A “greenfield” project is one that lacks constraints imposed by prior work.
- The challenges and limitations in the adoption of agile software development methods in a large organization with legacy infrastructure are well documented and understood.

Focus on Agile Methodologies

Agility at scale

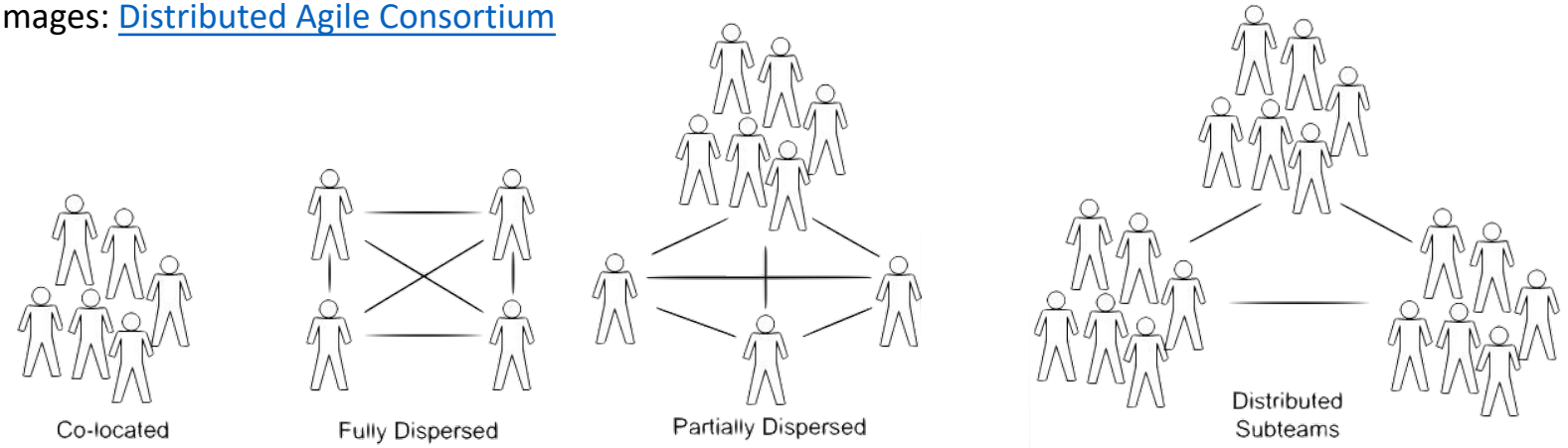
- A range of strategies and patterns have evolved for overcoming challenges with large-scale development efforts (>20 developers) or distributed (non-colocated) teams.
- There are now several recognized agile methodologies that seek to mitigate or avoid the challenges. Here are just a few:
  - Scaled agile framework (SAFe)
  - Disciplined agile delivery (DAD)
  - Large-scale scrum (LeSS)
  - Nexus (scaled professional Scrum)
  - Scrum at Scale
  - Enterprise Scrum
  - Setchu (Scrum-based lightweight framework)
  - Industrial XP (IXP)

Focus on Agile Methodologies

Agility at scale

- There are conflicting viewpoints on whether these are effective or fit the definition of agile development, and this remains an active and ongoing area of research.
- Agile software development applied in a distributed setting (with teams dispersed across multiple business locations), is commonly referred to as *distributed agile development*.

images: Distributed Agile Consortium



Focus on Agile Methodologies

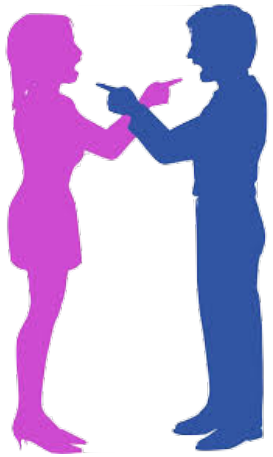
Agility at scale

- Distributed agile development allows software organizations to build software by setting up teams in different parts of the globe, virtually building software round-the-clock.
- This is more commonly referred to as *follow-the-sun* model.
- The application of agile development to distributed teams provides increased transparency, continuous feedback and more flexibility when responding to changes.

Focus on Agile Methodologies

Politics of Agile Development

- There continues to be wide debate about the benefits and applicability of agile software development as opposed to more traditional software processes.
- Would traditional methodologists rather produce flawless documentation than meet real business needs? Are agilists glorified hackers who distain sound engineering practices?
- Though these are extremes, the debate can easily degenerate into religious wars rather than rational discussions.



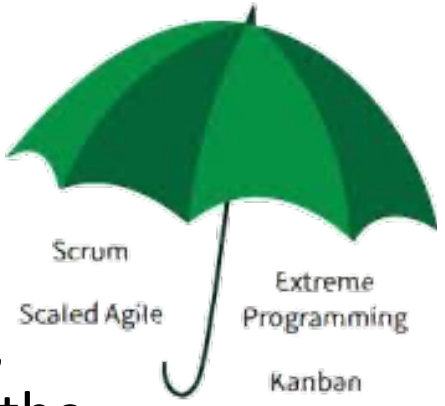
Focus on Agile Methodologies

Politics of Agile Development

- No one is against agility. The question is: what is the best way to achieve it?
- How to build software that meets customers' needs today with quality characteristics that will enable it to be extended and scaled to meet customers' needs in the long term?
- Even within the Agile school, process models take different approaches that depart from traditional methodologies, but they still adopt good software engineering practices.
- There are no absolute answers, but there is much to gain by considering the best of both the traditional and Agile schools.

Focus on Agile Methodologies

Examples of Agile Software Development Processes



- In this next part of the lecture, we will look in more detail at two lightweight software development processes that fall under the Agile umbrella.
- They were developed at about the same time, several years prior to the Agile manifesto and the coining of the term, and they both share the same agile philosophy and principles.
  - **Scrum** (1995) focuses on managing the workflow
  - **Extreme Programming – XP** (1996) focuses on the practices

Focus on Agile Methodologies

Scrum framework

- Scrum is an agile process *framework* for managing complex knowledge work, with initial emphasis on software development.
- Ken Schwaber and Jeff Sutherland co-developed the Scrum framework in the early 1990s, and both were signatories to the Agile manifest in 2001.
- Schwaber went on to found the *Agile Alliance* in 2005, and the *Scrum Alliance* in 2009.

Focus on Agile Methodologies

Scrum framework

- Scrum principles guide development activities within a process that incorporates the following framework activities:
  - requirements
  - analysis
  - design
  - evolution
  - delivery
- For each framework activity, work tasks occur within a process pattern called a *sprint*. Each sprint is no longer than one month and most commonly two weeks.
- Work is performed within a sprint by a *scrum team*. Progress is tracked and re-planning occurs in daily 15-minute time-boxed meetings called *daily scrums*.

Focus on Agile Methodologies

Scrum framework

- The term “scrum” comes from rugby football, and involves players packing closely together with their heads down. trying (sometimes violently) to move the ball down-field.

Photo: Wikipedia

Focus on Agile Methodologies

Scrum framework

- Scrum process flow

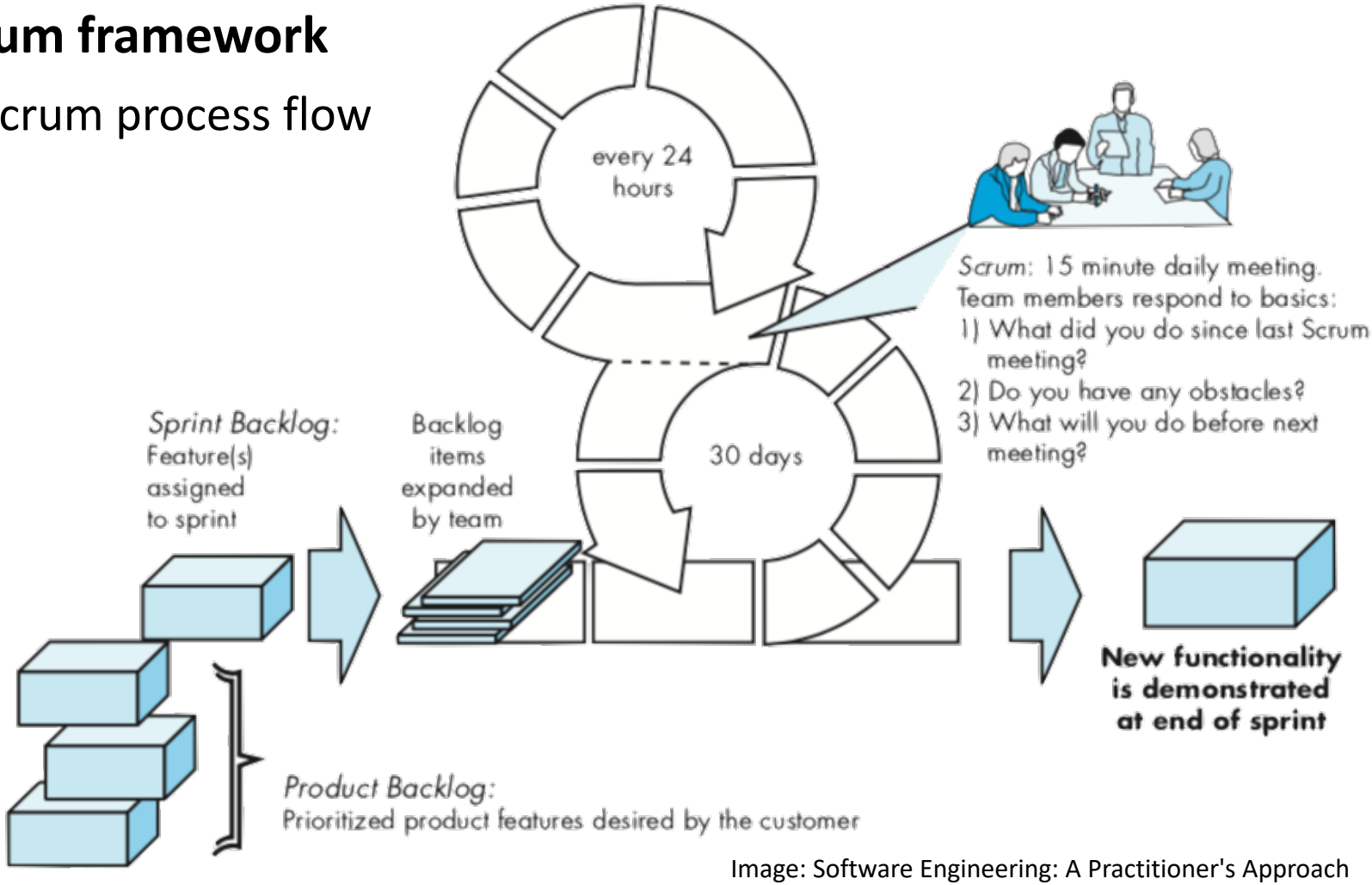


Image: Software Engineering: A Practitioner's Approach

Focus on Agile Methodologies

Scrum framework

- There are three roles in the Scrum framework. These are ideally co-located to ensure optimal communication among team members.
- While many organizations have other roles involved with defining and delivering the product, Scrum defines only these three:
  - Product owner
  - Development team
  - Scrum master

Focus on Agile Methodologies

Scrum framework

- **Product owner** -- represents the product's stakeholders and the voice of the customer (or may represent the desires of a committee)
- The product owner is responsible for delivering good business results. Hence, the product owner is accountable for the product backlog and for maximizing the value that the team delivers.
- The product owner defines the product in customer-centric terms (typically user stories), adds them to the *product backlog*, and prioritizes them based on importance and dependencies.
- A scrum team should have only one product owner (although a product owner could support more than one team). This role should not be combined with that of the *scrum master*.

Focus on Agile Methodologies

Scrum framework

- **Development team** -- has from three to nine members who carry out all tasks required to build increments of valuable output every sprint.
- Team members include researchers, architects, designers, data specialists, statisticians, analysts, programmers, and testers among others.
- The team is self-organizing. No work should come to the team except through the product owner. The scrum master is expected to protect the team from too much distraction,
- However, the team should still be encouraged to interact directly with customers and/or stakeholders to gain maximum understanding and immediacy of feedback.

Focus on Agile Methodologies

Scrum framework

- **Scrum master** -- is accountable for removing any impediments to the ability of the team to deliver the product goals and deliverables.
- The scrum master is not a traditional team lead or project manager but acts as a buffer between the team and any distracting influences.
- The scrum master ensures that the scrum framework is followed and helps to ensure the team follows the agreed processes in the Scrum framework.
- The scrum master often facilitates key sessions and encourages the team to improve. The role has also been referred to as a team facilitator or servant-leader to reinforce these dual perspectives.

Focus on Agile Methodologies

Scrum framework

- Scrum emphasizes the use of a set of software process patterns that have proven effective for projects with tight timelines, changing requirements, and business criticality.
- Each of these process patterns defines a set of development activities:
  - sprints
  - backlogs
  - scrum meetings
  - demos

Focus on Agile Methodologies

Scrum framework

- **Sprints**—work units required to achieve a requirement defined in the backlog; must fit into a predefined time-box (typically 30 days).
- At the beginning of a sprint, team holds a sprint planning event to:
  - Discuss and agree on the scope of work during that sprint.
  - Select product backlog items that can be completed in one sprint.
  - Prepare sprint backlog with work for selected product backlog items.
  - Agree the sprint goal, short description of delivery at the end.
  - Recommended duration is 4 hours for a two-week sprint.
    - During first half, select product backlog items they think could be completed.
    - During second half, identify detailed work (tasks) required to complete items.
    - Some product backlog items may be split or put back if cannot finish in sprint.
  - Development team votes which tasks will be delivered within the sprint.

Focus on Agile Methodologies

Scrum framework

- **Backlogs** -- prioritized lists of project requirements or features that provide business value for the customer. These are often in the form of customer-centered “stories” that describe features.
- The product owner can add Items to the *product backlog* at any time (this is how changes are introduced). The product owner assesses the product backlog and updates priorities as required.
- During planning for a sprint, features from the product backlog that can be completed in one sprint are selected and added to the *sprint backlog.*
- Features are not added to the sprint backlog during the sprint. Hence, team members can work in a short-term, but stable environment.

Focus on Agile Methodologies

Scrum framework

- **Scrum meetings** — are short (typically 15-minute) meetings held daily by the Scrum team. Three key questions are asked and answered by all team members:
  - What did you do since the last team meeting?
  - What obstacles are you encountering?
  - What do you plan to accomplish by the next team meeting?
- The Scrum master, leads the meeting and assesses the responses from each person.
- The Scrum meeting helps the team to uncover potential problems as early as possible. Also, these daily meetings lead to “knowledge socialization” and so promote a self-organizing team structure.

Focus on Agile Methodologies

Scrum framework

- **Demos** — deliver the software increment to the customer so that functionality that has been implemented can be demonstrated and evaluated by the customer.
- It is important to note that the demo may not contain all planned functionality, but rather those functions that can be delivered within the sprint time-box that was established.

Focus on Agile Methodologies

Extreme Programming (XP)

- Extreme Programming (XP) is an agile process for managing complex knowledge work, with an initial emphasis on software development.
- Kent Beck and Martin Fowler co-developed the XP methodology in the early 1990s, and both were signatories to the Agile manifest in 2001. A variant, Industrial XP (IXP) targets large organizations.
- XP is one of the most widely-used Agile frameworks.

Focus on Agile Methodologies

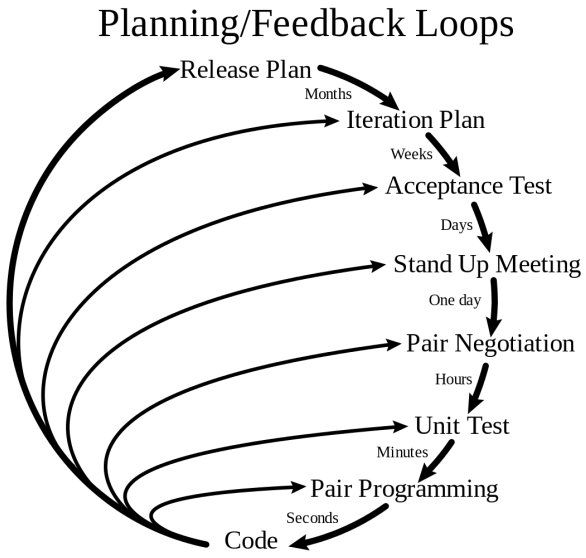
Extreme Programming (XP)

- Extreme Programming (XP) takes its name from the idea that the beneficial elements of traditional software engineering practices are taken to "extreme" levels.
- As an example, code reviews are considered a beneficial practice; taken to the extreme, code can be reviewed continuously, i.e. the practice of *pair programming*.

Focus on Agile Methodologies

Extreme Programming (XP)

- Elements of XP include:
  - programming in pairs or doing extensive code review
  - unit testing of all code, avoiding programming of features until they are actually needed
  - a flat management structure
  - code simplicity and clarity
  - expecting changes in the customer's requirements as time passes and the problem is better understood
  - frequent communication with the customer and among programmers



Focus on Agile Methodologies

Extreme Programming (XP)

- Extreme Programming uses an object-oriented approach as its preferred development paradigm and encompasses rules and practices that occur in the context of four framework activities:
  - planning
  - design
  - coding
  - testing

Focus on Agile Methodologies

Extreme Programming (XP)

- XP process flow

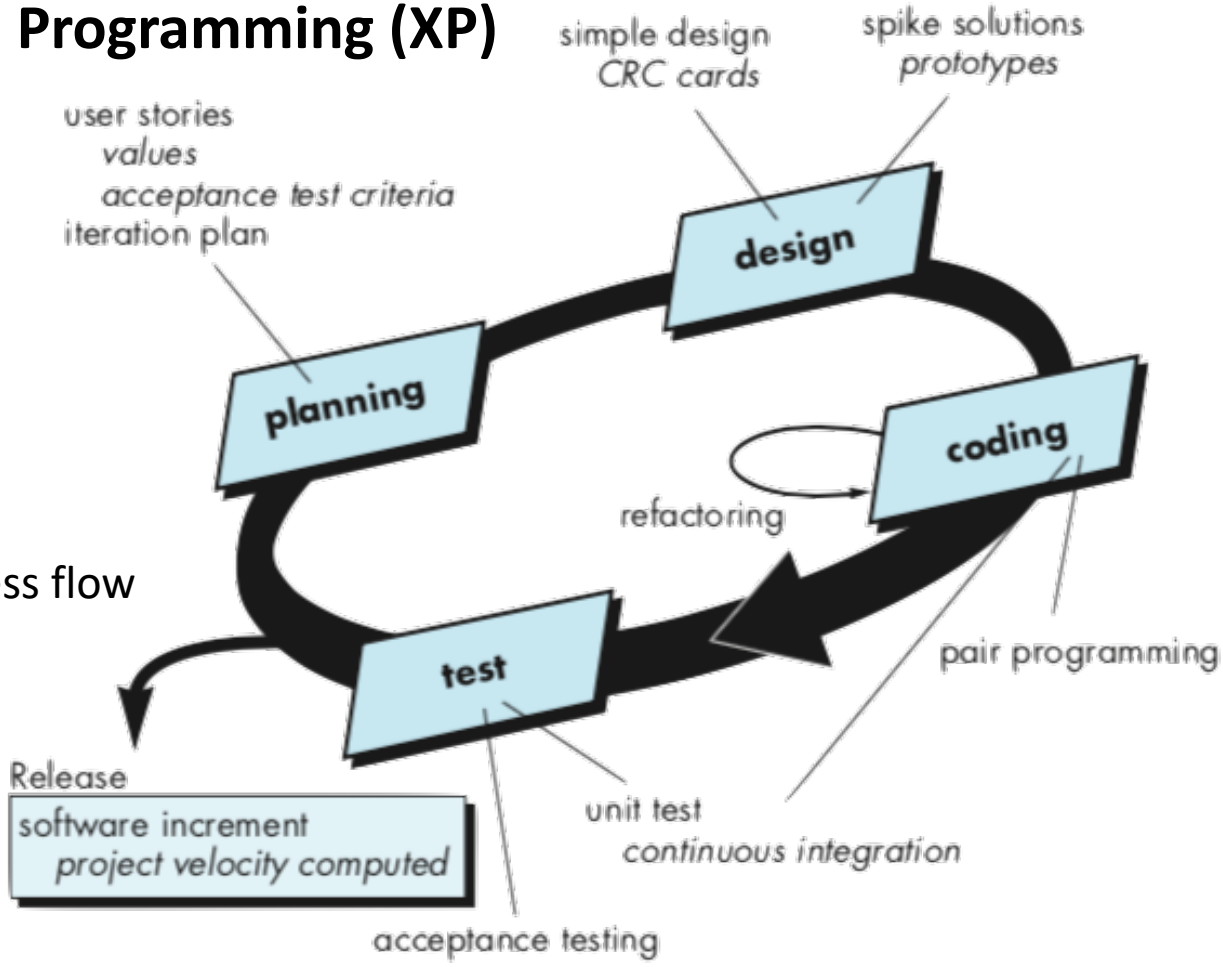


Image: Software Engineering: A Practitioner's Approach

Focus on Agile Methodologies

Extreme Programming (XP)

- **Planning** -- (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
- 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 (similar to a use case) is written by the customer and is placed on an index card. The customer assigns a value (i.e., a priority) to the story based on the overall business value of the feature or function.

Focus on Agile Methodologies

Extreme Programming (XP)

- **Planning (cont'd)**
- 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.

Focus on Agile Methodologies

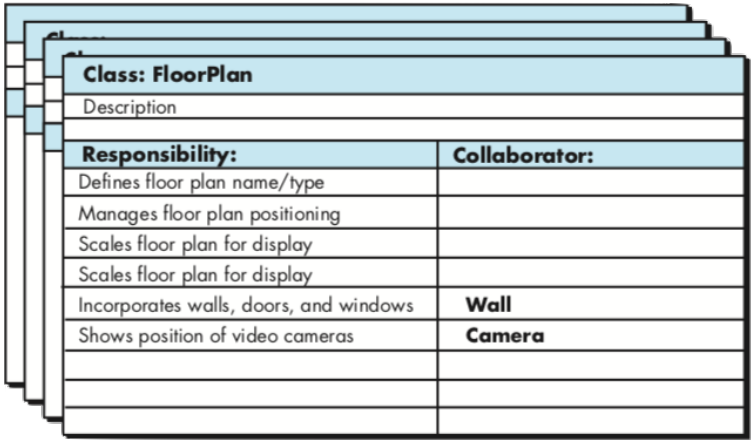
Extreme Programming (XP)

- **Planning (cont'd)**
- After the first project release (also called a software increment) has been delivered, the XP team computes *project velocity*.
- 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
  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.

Focus on Agile Methodologies

Extreme Programming (XP)

- **Design** -- XP rigorously follows the KIS (keep it simple) principle. The design provides implementation guidance for a story as it is written. The design of extra functionality is discouraged.
- XP encourages the use of CRC (class-responsibility-collaborator) cards as an effective mechanism for thinking about OO software
- CRC cards identify and organize the object-oriented classes relevant to the current software increment. The XP team conducts the design exercise.
- The CRC cards are the only design work product produced as part of the XP process.



CRC Model Index card.
Image: Software Engineering: A Practitioner's Approach

Focus on Agile Methodologies

Extreme Programming (XP)

- **Design (cont'd)**
- 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.
- The design prototype, called a *spike solution*, is implemented and evaluated. The intent is to lower risk when true implementation starts and validate the original estimates for the story containing.

Focus on Agile Methodologies

Extreme Programming (XP)

- **Design (cont'd)**
- XP encourages *refactoring*—a construction technique that is also a design technique—to improve the internal design without altering the external behavior.
- Design is a transient artifact that can and should be continually modified as construction proceeds.
- A central notion in XP is that design occurs both before and after coding commences. Refactoring means that design occurs continuously as the system is constructed.
- The construction activity itself will provide the XP team with guidance on how to improve the design.

Focus on Agile Methodologies

Extreme Programming (XP)

- **Coding** -- After stories are developed and preliminary design work is done, the team does not move to code.
- The first develop a series of unit tests to exercise each story that is to be included in the current release (software increment) .
- Once a unit test is created, the developer is can 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.

Focus on Agile Methodologies

Extreme Programming (XP)

- **Coding (cont'd)**
- XP recommends pair-programming -- two people working 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 (code is reviewed as it is created). It also keeps the developers focused.

image: DeveloperExperience.io

Focus on Agile Methodologies

Extreme Programming (XP)

- **Coding (cont'd)**
- As pair programmers complete their work, their code is integrated with the work of others. In some cases, this done daily by an integration team. In others the programmers are responsible.
- This "continuous integration" strategy helps to avoid compatibility and interfacing problems and provides a "smoke test" environment that helps uncover obvious errors early.

Focus on Agile Methodologies

Extreme Programming (XP)

- **Testing** -- tests should be implemented using a test automation framework so they can be executed easily and repeatedly. Encourages regression testing strategy whenever code is modified.
- Organize unit tests into a “universal testing suite” so integration and validation testing can occur daily. This demonstrates progress and 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.

Focus on Agile Methodologies

Scrum vs. XP

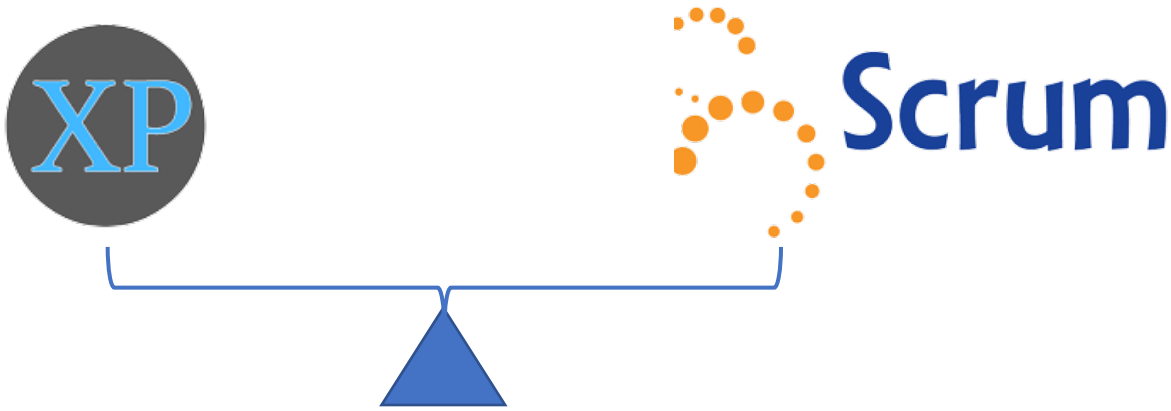
- We can summarize difference between Scrum and XP as follows:

Aspects	Practices	XP	Scrum
Iteration Length	Whether to allow modification of requirements	1-2 weeks	2-4 weeks
Handle Changes with an Iteration	Whether the demand is strictly in accordance with the priority	It can be replaced with other requirements when a need is not implemented, but the implementation time is equal.	Scrum is not allowed to do this. Once the iteration is completed, no changes are allowed, and Scrum Master is strictly checked.
Priority of Features	Whether the demand is strictly in accordance with the priority	Yes	No need to
Engineering Practices	Whether to adopt strict engineering methods to ensure progress or quality	Very strict	Require developers to be conscious

Focus on Agile Methodologies

Scrum vs. XP

- XP's approach brings Agile into a confusing paradox, because XP, combined with agile says, "you are fully self-managed, but have to use Test-Driven Development (TDD), pair programming, etc."
- It is not difficult to find that the distinctions are obvious:
  - Scrum emphasizes self-organization.
  - XP emphasizes strong engineering practice constraints.



Focus on Agile Methodologies

Scrum vs. XP

- Scrum is a framework for product development, which is a container where you can add other practices. XP is one of those practices that you can do within Scrum framework.
- There is no reason why a team needs to choose between Scrum and XP. XP rules and practices are not easy, and the majority of XP rules are non-negotiable.
- Adding XP into Scrum could be a natural path for teams starting out with Scrum and striving to be a professional Scrum Team.