Agile Development

Chapter 3

Outline

- ❑ **Common fears for developers**
- ❑ **What is agility?**
- ❑ **Agile process**
- ❑ **Principles of agile methods**
- ❑ **Agile process models**
- ❑ **Understanding Scrum**
- ❑ **Extreme programming (XP)**

Quick Look on Agile Process Model

The meaning of Agile is **swift or versatile.**

"**Agile process model**" refers to a software development approach based on **iterative development.**

Agile methods **break tasks into smaller iterations**, or **parts do not directly involve long term planning.**

The project scope and requirements are **laid down at the beginning of the development process**.

Plans regarding the **number of iterations**, the **duration and the scope of each iteration are clearly defined in advance**.

Quick Look on Agile Process Model

Who does it?

Software engineers and other project stakeholders (managers, customers, end users) **work together** on an agile.

An agile team **care for communication and collaboration** among all who serve on it.

Quick Look on Agile Process Model

پوهنتون کاردان
KARDAN UNIVERSITY

Why is it important?

The modern business environment that use computer-based systems and software products is **fast-paced** and **ever-changing**.

Agile software engineering represents a **reasonable alternative to conventional software engineering** for certain **classes of software and certain types of software projects.**

It has been demonstrated to **deliver successful systems quickly**.

Quick Look on Agile Process Model

What are the steps?

Agile development might best be termed "**software engineering lite.**"

The basic framework activities—communication, planning, modeling, construction, and deployment— remain.

But they form into a minimal task set that pushes the project team

toward **construction and delivery**.

Quick Look on Agile Process Model

How do I ensure that I've done it right?

If the agile team agrees that the **process works**, and the team **produces deliverable** software

increments that **satisfy the customer**, you've done it right.

Quick Look on Agile Process Model

What is the work product?

Both the customer and the software engineer have the same view—the only really important work product is an operational **"software increment" that is delivered to the customer on the appropriate commitment date.**

Strategy of Agile

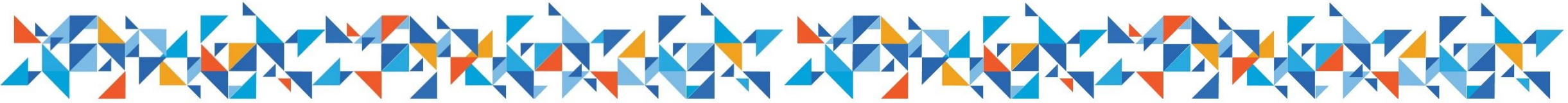
“We are uncovering better ways of developing software by doing it and helping others do it.

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

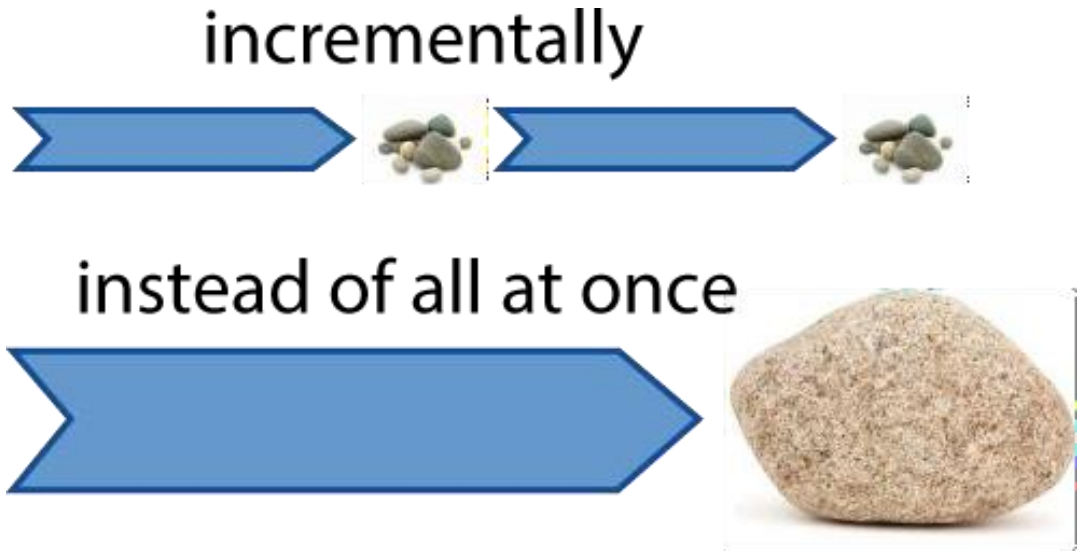
That is, while there is value in the items on the right, we value the items on the left more.”

Kent Becketal



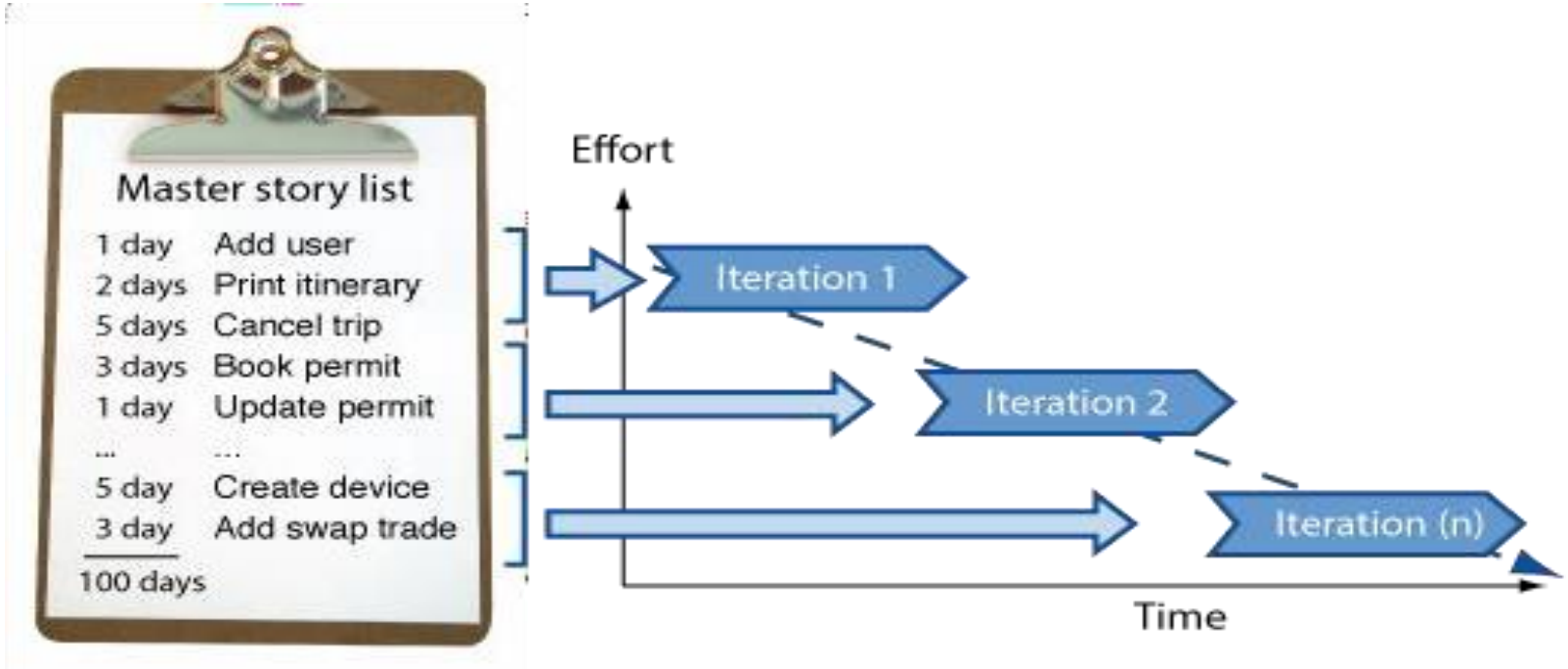
What is “Agile”?

Agile is a **time boxed**, iterative approach to software delivery that builds software incrementally from the start of the project, instead of trying to deliver it all at once near the end.



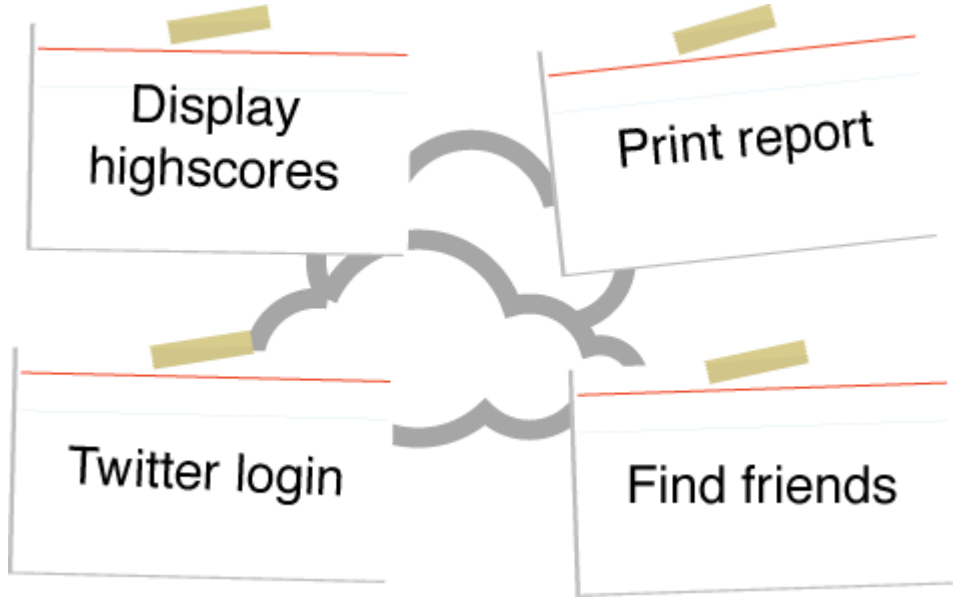
What is Agile?

It works by breaking projects down into little bits of user functionality called **User Stories**, prioritizing them, and then continuously delivering them in short two week cycles called **iterations**.



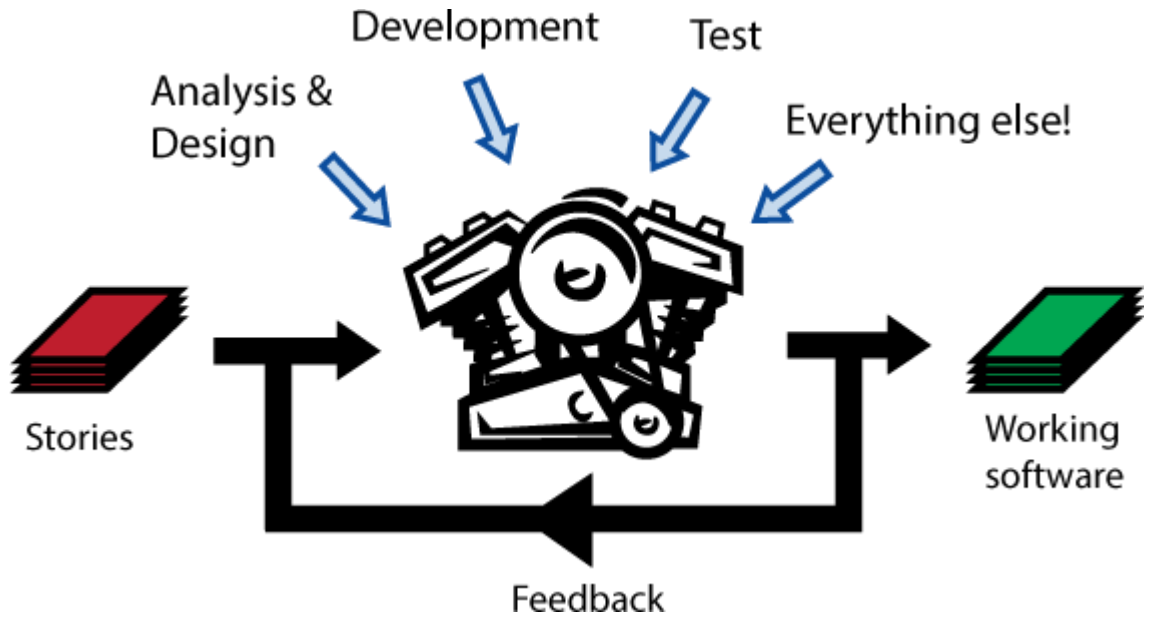
User Stories

- Because life's too short to write everything down
- User stories are features our customers might one day like to see in their software.



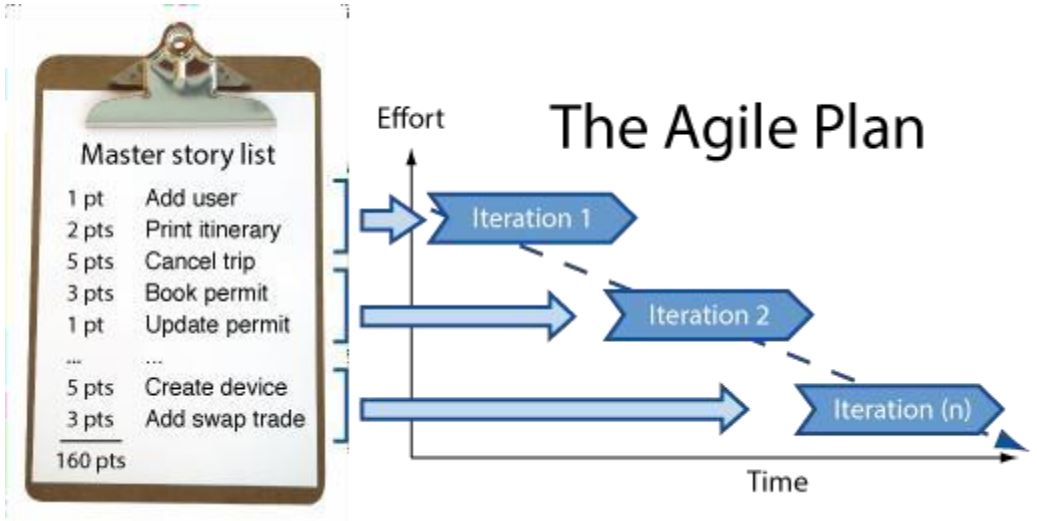
Iterations

- Agile's engine for getting things done
- An Agile iteration is a short one to two week period where a team takes a couple of their customers **most important user stories** and builds them completely as running-tested-software.



Iterations

- This means everything happens during an iteration**. Analysis, design, coding, testing**. It all happens here. The beauty of working this way, is every couple weeks the customer gets something of great value (working software).



What is “Agility”?

- Ability to move quickly and easily.
- Effective (rapid and adaptive) response to change
- Effective communication among all stakeholders
- Organizing a team so that it is in control of the work performed

Yielding …

- Rapid, incremental delivery of software

How Does it Work?

At its core, Agile does the same thing you and I do when faced with too much to do and not enough time.

You make a list

Sitting down with your customer you make a list of features they would like to see in their software. We call these things **User stories** and they become the To Do list for your project.

You size things up

Then, using Agile estimation techniques, you size your stories relatively to each other, coming up with a guess as to how long you think each user story will take.

How Does it Work?

You set some priorities

Like most lists, there always seems to be more to do than time allows. So you ask your customer to prioritize their list so you get the most important stuff done first, and save the least important for last.

You start executing

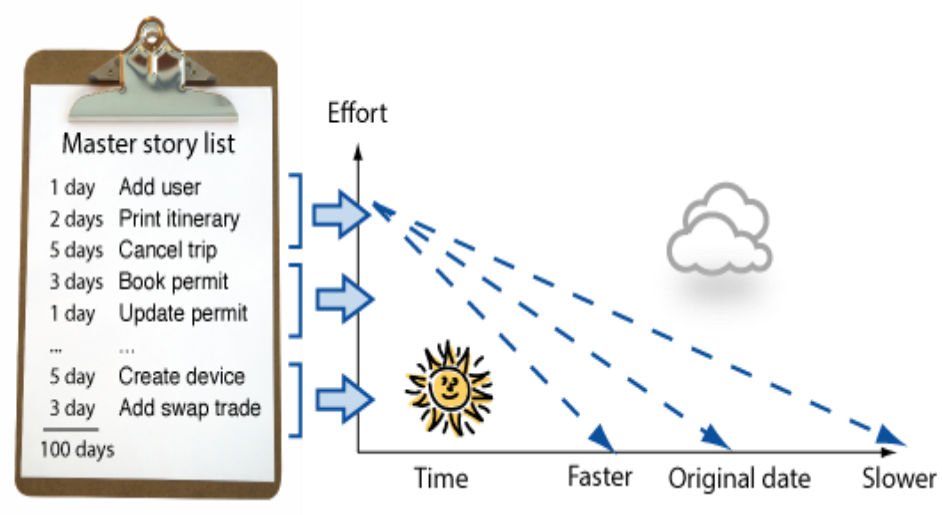
Then you start delivering some value. You start at the top. Work your way to the bottom. Building, iterating, and getting feedback from your customer as you go.

How Does it Work?

You update the plan as you go.
Then, as you and your customer starting delivering, one of two things is going to happen. You'll discover:

a. **You're going fast enough. All is good**. Or,

b. **You have too much to do and not enough time.**



At this point you have two choices.

a) do less and cut scope (recommended).

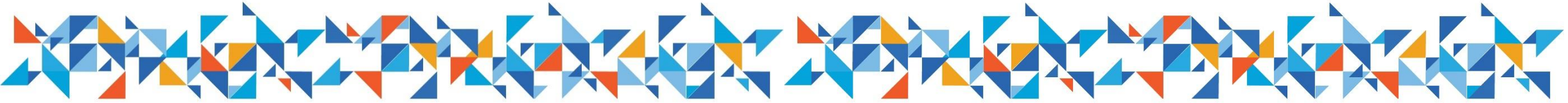
b) push out the date and ask for more money.

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 bind 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.

Agility Principles Cont..

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 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.



Agility Principles Cont..

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.

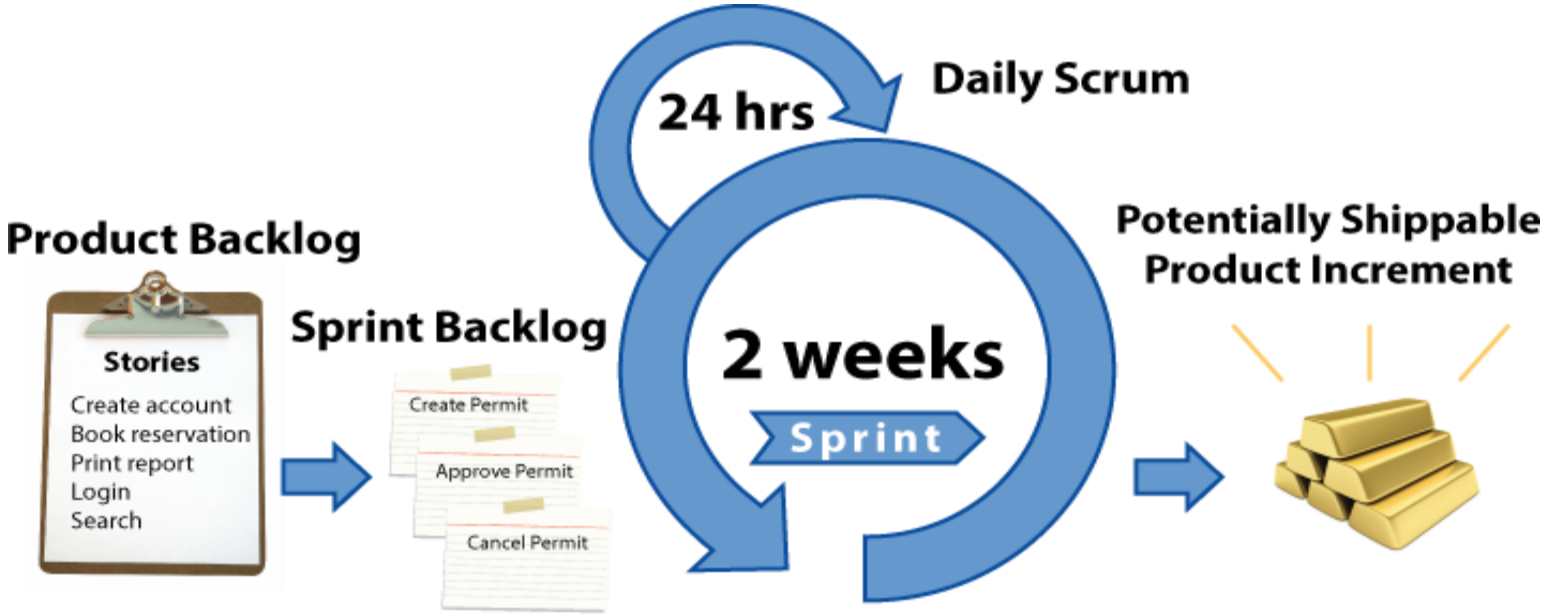
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile Methods (agile process models)

- Several methods that are often cited to be agile, e.g.,
  - Extreme Programming
  - SCRUM
  - Lean Software Development
  - Crystal Methodology
  - Kanban

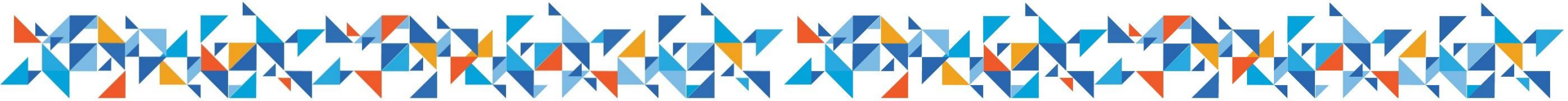
Scrum

Perhaps the most popular of Agile methods today, Scrum is an project management framework that encourages teams to self organize and deliver functionality iteratively in two week time boxes called **sprints**.



Scrum Cont..

Agile is a foundational philosophy and mindset,
while scrum is a **framework** that materialize or bring that philosophy into life.



SCRUM Three Pillars

1. **Transparency**

   Make the most significant aspects of our work visible to those responsible for the outcomes.

2. **Inspection**

   Conducting timely checks towards the outcomes of a sprint goal to detect undesirable variances.

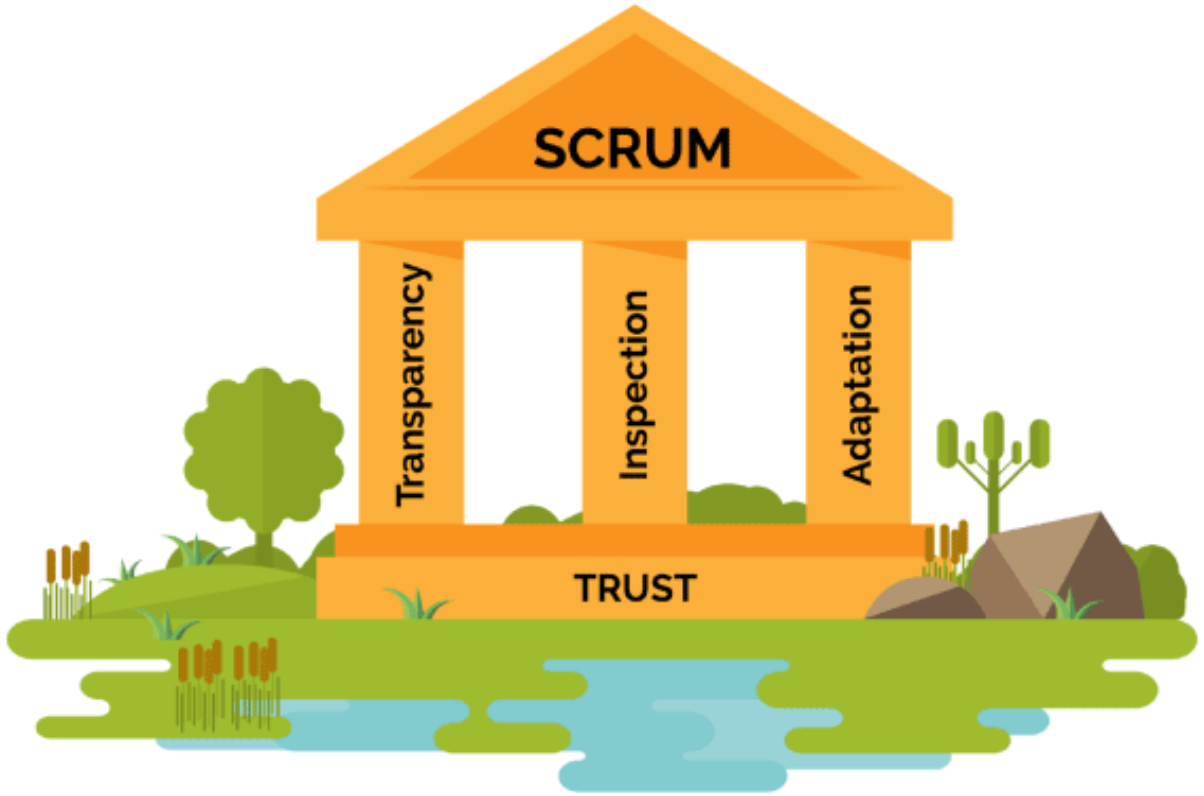
3. **Adaptation**

   Proactive adjustment to **reduce risk** or **keep the project aligned with goals.**.

SCRUM Five Values

1. **Commitment:**
   Personally committing to achieving the goals of the scrum team.
2. **Courage:**
   The scrum team members must have courage to do the right things and work on tough problems.
3. **Focus:**
   Everyone focusing on the work of the sprint and the overall goals of the scrum team.
4. **Openness:**
   The scrum team and its stakeholders agree to be open about all the work and challenges with performing the work.
5. **Respect:**
   Team members should respect the opinions , skills, and independence of their teammates.

Scrum Values and Pillars



COURAGE
Scrum Team members have courage to do the right thing and work on tough problems

FOCUS
Everyone focuses on the work of the Sprint and the goals of the Scrum Team

COMMITMENT
People personally commit to achieving the goals of the Scrum Team

RESPECT
Scrum Team members respect each other to be capable, independent people

OPENNESS
The Scrum Team and its stakeholders agree to be open about all the work and the challenges with performing the work

Credit: ABN AMRO Bank N.V.

Scrum Team

The fundamental unit of Scrum is a small team of people, a Scrum Team. The Scrum Team consists of one Scrum Master, one Product Owner, and Developers.

1. Developers

- Developers are the people in the Scrum Team that are committed to creating any aspect of a usable Increment.

2. Product Owner

- The Product Owner is accountable for maximizing the value of the product.

3. Scrum Master

- The Scrum Master is accountable for establishing Scrum as defined in the Scrum Guide.

Scrum Events

- **Sprint**
  - Sprints are the heartbeat of Scrum, where ideas are turned into value.
  - They are fixed length events of one month or less to create consistency
- **Daily Scrum**
  - The Daily Scrum is a 15-minute event for the Developers of the Scrum Team.
- **Sprint Review**
  - Scrum Team and stakeholders review what was accomplished in the Sprint and what has changed in their environment.

Scrum Events

- Product Backlog
  - Product Backlog items that can be Done by the Scrum Team within one Sprint are deemed ready for selection.
    - Often include such attributes as *description, value, order, estimate, and size*.
- Sprint Backlog
  - Real-time picture of the work that the Developers plan to accomplish during the Sprint in order to achieve the Sprint Goal

User-stories

- **Epic**: broad picture from software requirements, which is collection of user-stories.

- **User-stories**: gathering requirements of software in the form story.

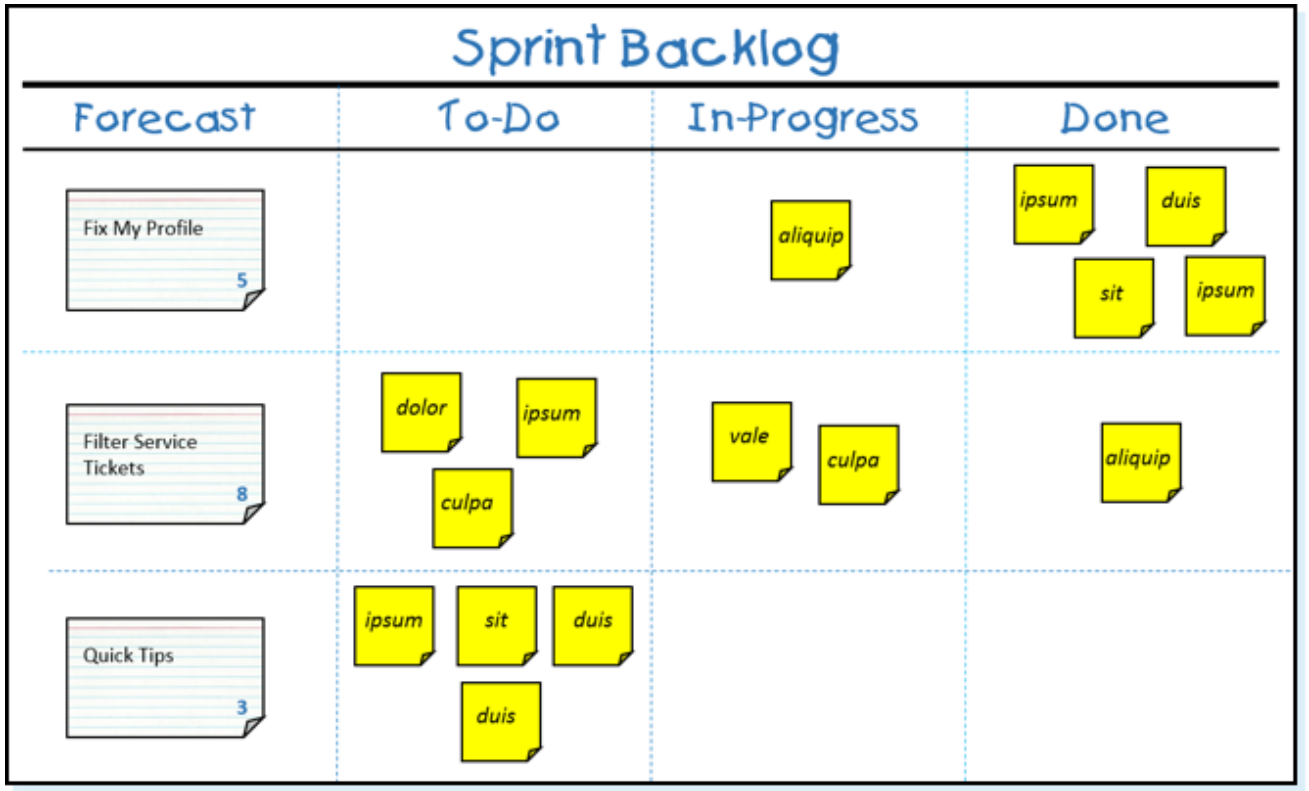
  User-story template:
  - As a ***<user-role>,*** I want this ***<action>,*** to get this ***<value>***

Bansal Tree

Product backlog: Example

Epic	User Story Title	Story	Acceptance Criteria
Bonsai Trees	Bonsai Selection	As a potential customer, I want to be able to select from a variety of different Bonsai trees so that I can pick the exact type I want.	1) Ability to select from multiple tree species 2) Options to sort by species, size, and care requirements
Bonsai Trees	Bonsai Shipping	As a customer, I want to know that my tree will be shipped securely so that it arrives in good condition.	1) Can review graphic explaining packing and shipping process during checkout 2) Signature required when a Bonsai shipment is received
Bonsai Trees	Bonsai Styles	As a new Bonsai tree owner, I want to learn about different Bonsai styles so I can decide which is right for my tree.	1) Customers can visit a guide to different styles (formal & informal upright, broom, cascade, etc.) on website 2) Customers can take a quiz to match their' trees to suitable styles
Bonsai Trees	Bonsai Tools	As a Bonsai tree owner, I want to have the right tools to care for my tree so I can shape and style it properly.	1) Option to purchase Bonsai starter kits (shears, wire, mesh, fertilizer, etc.) available for purchase 2) Customers can choose from a selection of premium tools (grafting knives, carving tools, root hooks, etc.)

Sprint backlog

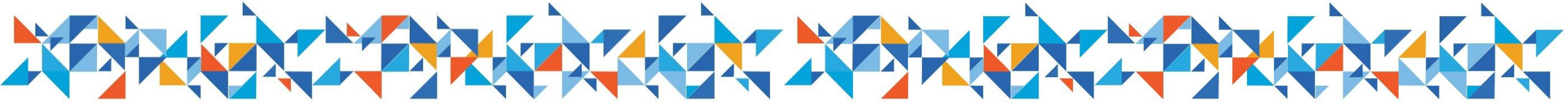


Extreme Programming (XP)

Extreme Programming (XP)

Extreme Programming (XP) takes an ‘extreme’ approach to iterative development.

- A software development methodology that aims to improve software quality and responsiveness to changing customer requirements.
- Created by Kent Beck in the late 1990s.
- XP emphasizes frequent releases, close collaboration, and continuous feedback.



Key Principles of XP

- **Communication:** Close collaboration between developers, clients, and team members.
- **Simplicity:** Focus on the simplest possible solution.
- **Feedback:** Continuous feedback from tests and customers.
- **Courage:** Ability to make changes and adapt as needed.
- **Respect:** Valuing the input of all team members.

XP vs. Traditional Development

- Waterfall vs. XP:
  - **Waterfall**: Linear, sequential approach.
  - **XP**: Iterative, frequent feedback cycles, continuous integration.

Waterfall	XP
Requirements fixed	Requirements evolve
Phases are distinct	Phases overlap
Long release cycles	Short, frequent iterations

XP Core Practices

1) **Pair Programming**: Two developers work together at one workstation.

2) **Test-Driven Development (TDD):** Write tests before the actual code.

3) **Continuous Integration:** Integrate code into a shared repository several times a day.

4) **Simple Design:** Design as simply as possible at all times.

Pair Programming

- What is Pair Programming?

- One programmer writes code, while the other reviews and provides feedback.
  - Benefits:
    - Higher code quality.
    - Shared knowledge.
  - Challenges:
    - Requires good communication and collaboration.

Test-Driven Development (TDD)

- What is TDD?
  - Write automated tests before writing the code.
  - Cycle: **Red** (Fail) -> **Green** (Pass) -> Refactor.

- Benefits of TDD:
  - Encourages modular design.
  - Ensures that code meets requirements from the start.

Continuous Integration

- What is Continuous Integration (CI)?
  - Developers frequently commit code to a central repository.
  - Automated builds and tests run after each commit.

- Benefits of CI:
  - Reduces integration issues.
  - Catches bugs early in the development process.

Refactoring

- What is Refactoring?
  - Improving the design and structure of the existing code without changing its functionality.

- Why Refactor?
  - Maintain code simplicity.
  - Improve performance and scalability.
  - Reduce technical debt.

Customer Involvement in XP

- On-site Customer:
  - A customer or customer representative is always available to provide feedback.

- User Stories:
  - Simple, concise descriptions of features from the customer’s perspective.
  - Examples: "As a user, I want to log in securely."

Short Release Cycles

- XP emphasizes short release cycles (1-2 weeks).
  - Regular, frequent releases to customers.
  - Enables feedback and adjustment to requirements quickly.

- Advantages:
  - Faster response to changes.
  - Early detection of potential issues.

Collective Code Ownership

- What is Collective Ownership?
  - Everyone on the team is responsible for the entire codebase.

- Benefits:
  - Anyone can improve or fix any part of the code.
  - Encourages responsibility and reduces bottlenecks.

Coding Standards

- Why Coding Standards?
  - Consistent style makes code easier to read, understand, and maintain.

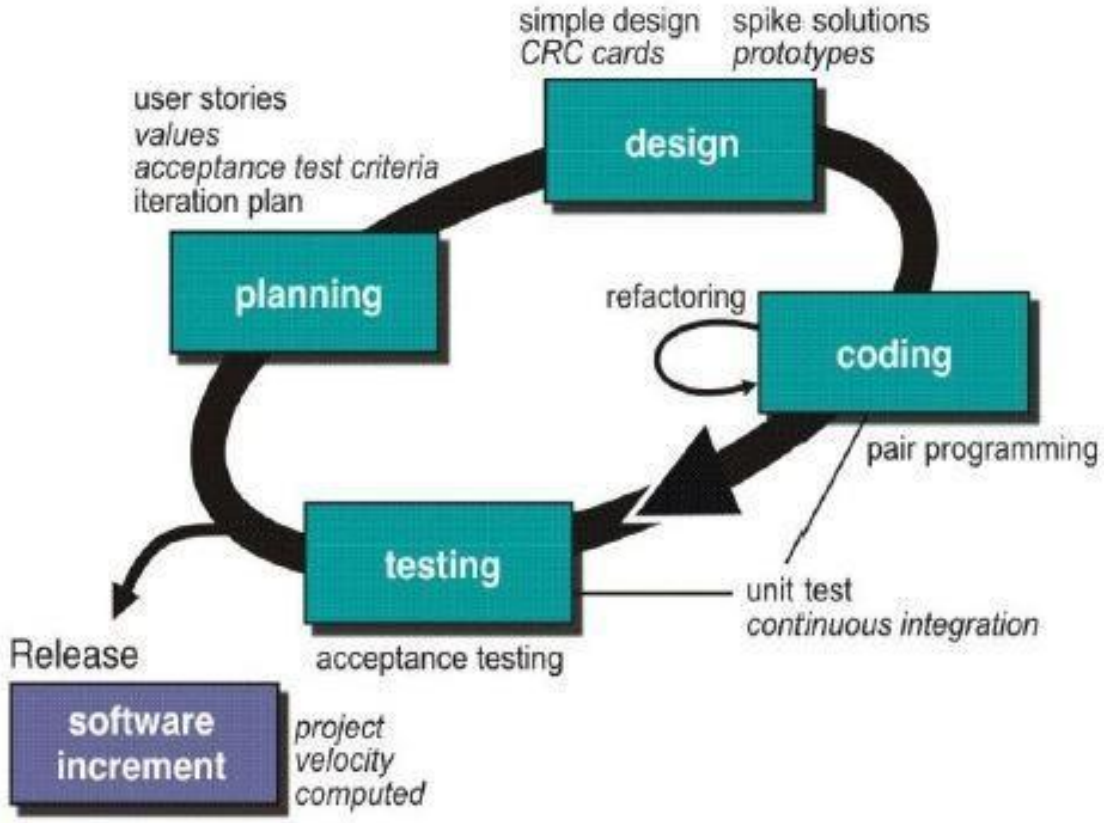
- XP Approach:
  - Use consistent naming, structure, and commenting across the team.
  - Tools such as linters can help enforce these standards.

XP Advantages and Challenges

- Advantages of XP:
  - Fast feedback loops.
  - High-quality code through testing and refactoring.
  - Frequent releases.

- Challenges of XP:
  - Requires a high level of discipline.
  - Difficult to implement in large, non-collaborative teams.
  - May require significant customer involvement.

Extreme Programming (XP)



ANY QUESTIONS ?