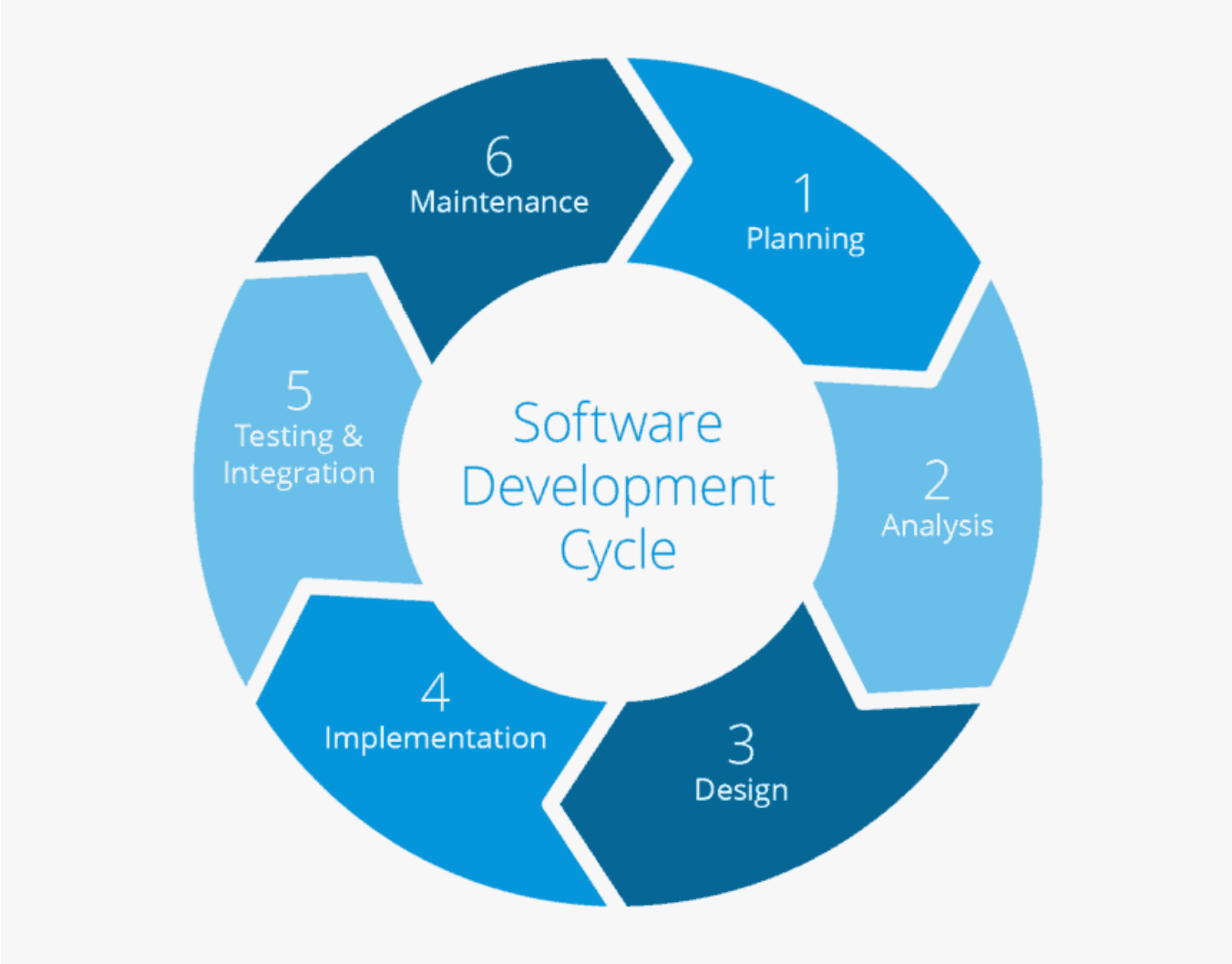
Software Development Process

A structure imposed on the development of a software product.
A framework that is used to structure, plan, and control the process of developing an information system.
There are several schemes of software development that have developed over time.

Software Development Life Cycle (SDLC)



The **SDLC** is a ***structured process*** that enables the production of high-quality low-cost software, in the shortest possible amount of time.
The SDLC defines and outlines a ***detailed plan*** with stages that each encompass a process of tasks.
The plan describes how to develop, maintain, replace, and enhance specific software.

- A **methodology** is defined for improving the quality of sotware and the overall development process.

Stages of the Life Cycle:

1. Planning Phase

An objective for each and every activity, where we want to discover things that belong to the project.

- Expectations are clearly defined; the team then determines what is desired but also what is ***NOT*** desired from the clients.
  Tangible results from this phase include: project plans, estimated costs, projected schedules, and procurement needs.

2. Analysis & Design

Analysis requirements and design of software is done throughout development.

- One design approach for the product is proposed and documented. This design is based on specified requirements.
- A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any).

3. Implementation

Implementation is the part of the process where software engineers write the actual code of the project.

- Developers follow coding guidelines.
- Programming language and tech stack is chosen with respect to the type of software being developed.

4. Testing & Integration

Software testing is the process to ensure that defects are recognized as soon as possible.

- Demonstrates that developed system conforms to requirements as specified in the Functional Requirements Document. Conducted by Quality Assurance staff and users. Products Test Analysis Reports.
  5. Maintenance
  Describes tasks to operate and maintain information systems in a production environment includes Post-Implementation and In-Process Reviews.

- Team fixes bugs, resolves customer issues, and manages software changes. In addition, the team monitors overall system performance, security, and user experience to identify new ways to improve the existing software.

SDLC Models

A software development lifecycle (SDLC) model conceptually presents SDLC in an organized fashion to help organizations to implement it. Different models arrange the SDLC phases in varying ways while adhering to the chronological ordering.

There are two classes of SDLC models:

1. ***Prescriptive Models*** (Traditional)
   - Prescriptive process models advocate an orderly appraoch to software engineering.
2. ***Agile Models*** (Modern)
   - Agile methods intend to develop systems **more quickly** with limited time spent on analysis and design.
   - Characterized by: ****Rapid and Adaptive response to change***, effective communication, drawing the customer onto the team, organizing a team that's in control of the work performed, rapid incremental delivery of software.

Prescriptive Development Models

Waterfall Method

The Waterfall Method is a **linear framework**.
The phases are strictly followed in order, each phase is dependent on the previous step.
The next phase doesn't start until the previous step is complete.

The Flaws of the Waterfall Model

- Real projects are rarely sequential.
- Most project requirements aren't clear at the begininning.
- Requirements can't be changed in the mdidle of the project.
- Not flexible.
- Development can take much longer and does not yield a working version of the system until late into the process.

Evolutionary Development

Combination of iterative and incremental models of SDLC.
All work is done during the development phase. Work is divided into small chunjs called modules.

Incremental Development

Development and delivery is brown down into increments with each increment delivering part of the required functionality.
Requirements are prioritized and highest priority requirements are included in early increments.
Once development of an increment is started the requirements are frozen through requirements for kater increments can continue to evolve.

The Flaws of the Incremental Model

- Lack of process visibility.
- Systems are often poorly structured.
- Not ideal for large systems.

Spiral Method

- Risk management at regular stages in development cycle.
- Combines key aspects of waterfall model and rapid prototyping
- Good for complex systems.

Rapid Application Development (RAD)

- Requires minimal planning
- Faster development
- Easier to change requriements
- Starts with Data Models and business process modeling
- Requirements are verified by prototyping eventually to refine the data and process models.

Agile Development Modles

More on the Agile Process

- Agile is driven by custoemr descriptions of what is required (scenarios).
  Recognizes that plans are short-lived.

- Develops software iteratively with a heavy emphasis on construction activities delivers multiple software increments
- Adapts as changes occur.
  Methods are considered Lightweight, ***people-based rather than plan-based***

Extreme Programming (XP)

Extreme Programming is used to improve software quality and responsiveness to customer requirement. **XP is built upon values, principles, and practices**, and its goal is to allow small to mid-sized teams to produce high-quality software and **adapt** to evolving requirements.

- **XP emphasizes the technical aspects of software development.**
- XP is precises on *how* engineers work.
  - Following engineering practices allows teams ot deliver high-quality code at a sustainable pace.

Why is it called "Extreme Programming"?

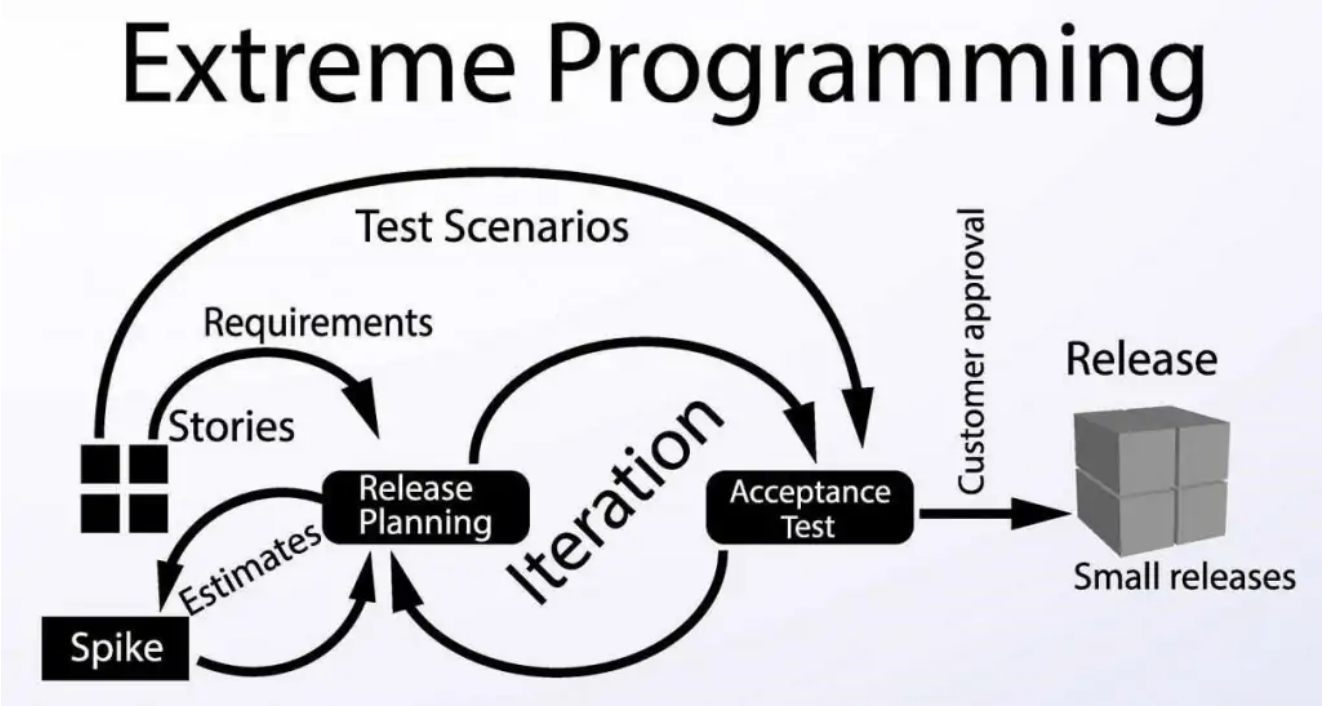
Well that's because XP is all about ***performing good practices in the extreme.***

- Maximizing on concepts like pair-programming and testing early to produce high-quality results.

Phases are carried out in extremely small and continuous releases. Carried out by the following process:

1. First write automated tests as concrete goal for development.
2. Coding is completed only if **ALL** tests pass.
3. Design and architecture emerge out of **refactoring**
4. The Incomplete but functional system is deployed or demonstrated.

5. Move to next part of the system.



XP Values:

- Communication
  - Lack of communication prevents knowledge from flowing inside a team. It may prevent a team from learning about a problem or a potential solution to said problem. So the problem ends up being solved twice and thus wasting time and resources.
- Simplicitiy
  - Simplificity is contextual, what's simple for one team is complex for another depending on a team's skills. Thus, strive for the simplest thing that works. THis will pay off when debugging and testing.
- Feedback
  - Embrace change, and constant communication with your client for feedback. XP follows feedback loops at different levels so that the project continues in the correct direction at all times.
- Courage
  - Effective action in the face of fear. It takes courage to speak the truth. Giving and receiving feedback also takes courage.
- Respect
  - Everyone cares about their work. No amount of technical excellence can save a project if there's no care and respect.

Principles:

- incremental changes, embracing change, assumed simplicity, quality work, rapid feedback.

Scrum

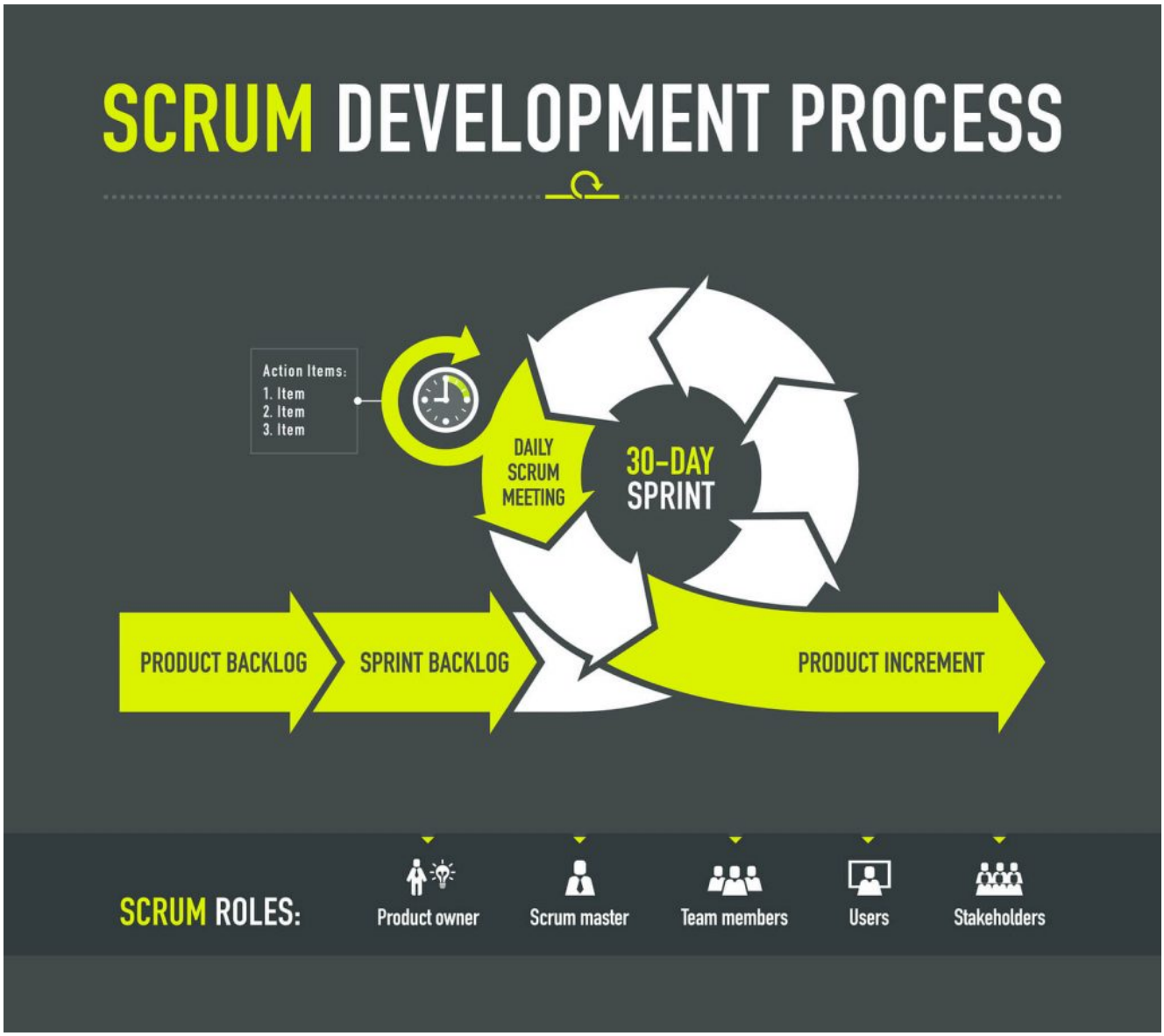
Scrum is adaptable, fast, flexible and effective agile framework that is designed to delvier value to the customer throughout the development of the project.

The primary objective of Scrum is to satify the customer's need through transparency.

Scrum can be described as:

- Enables the creation of self-organizing teams by encouraging co-location of all team members.
- Testing and documentation are on-going as the product is constructed.
- Work occurs in "sprints" and is derived from a "backlog" of existing requirements.
  - ***Sprints*** are timeboxed iterations of a continuous development cycle.
  - ***Backlogs*** are lists of all the things that need to be done within a sprint.
- Meetings are very short and sometimes conducted without chairs.

- Demos are delivered to the customer with the time-box allocated.



Test Driven Development

Test-Driven Development (TDD) is a process relying on software requirements being converted to test cases before software is fully developed.

- Test cases for each functionality are created and tested first and if the test fails then the new code is written in order to pass the test.
- TDD consists of designing and developing every small functionality of an application.
- The simple concept of TDD is to write and correct failed tests before writing new code.
- Only once the test passes, may the programmer refactor the design to make improvements.

TDD takes a functionality-first design.

There is a great focus on the programmer interactions at the unit level.

- Creating concrete examples as tests in collaboration between the staff and customer ***before*** code is created, and then running the tests after the code is created to demonstrate the code is implemented correctly.
- TDD requires the developers to accurately anticipate how the application and its features will be used in the real world.
  In short, TDD is based on the test first programming concept of XP. Initially failing tests are written before providing improvements. Programmers write the minimum amount of code to pass the test. Finally, code is re-factoredto acceptable standards.