Software Process

Lecture 2

Assoc.prof/ Mohammed Safy

Introduction

- The success of your business depends on more than just a great idea - to make sure that you reach your business goal you have to find a strategic alignment between the idea and the right software development model.
- **A software development model is a structured framework or methodology used to plan, design, create, test, and deploy software applications.** It's essentially a blueprint that guides the development process from initial idea to delivery.

The software process

A structured set of activities required to develop a software system.

Many different software processes but all involve:

- **Specification** – defining what the system should do;
- **Design** – defining the organization of the system;
- **Implementation** – implementing the system;
- **Validation (Testing)** – checking that it does what the customer wants;
- **Evolution (Maintenance)** – changing the system in response to changing customer needs.

A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.

Factors in choosing a software process

When embarking on a software development project, one of the critical decisions that can greatly impact its outcome is selecting the appropriate methodology. The choice of methodology sets the framework for how the project will be planned, executed, and managed.

Factors in choosing a software process

Choosing the right software process model for your project can be difficult. If you know your requirements well, it will be easier to select a model that best matches your needs. You need to keep the following factors in mind when selecting your software process model:

Project size

Consider the size of the project you will be working on. Larger projects mean bigger teams, so you'll need more extensive and elaborate project management plans.

Project complexity

Complex projects may not have clear requirements. The requirements may change often, and the cost of delay is high. Ask yourself if the project requires constant monitoring or feedback from the client.

Factors in choosing a software process

Cost of delay

Is the project highly time-bound with a huge cost of delay, or are the timelines flexible?

Customer involvement

Do you need to consult the customers during the process? Does the user need to participate in all phases?

Familiarity with technology

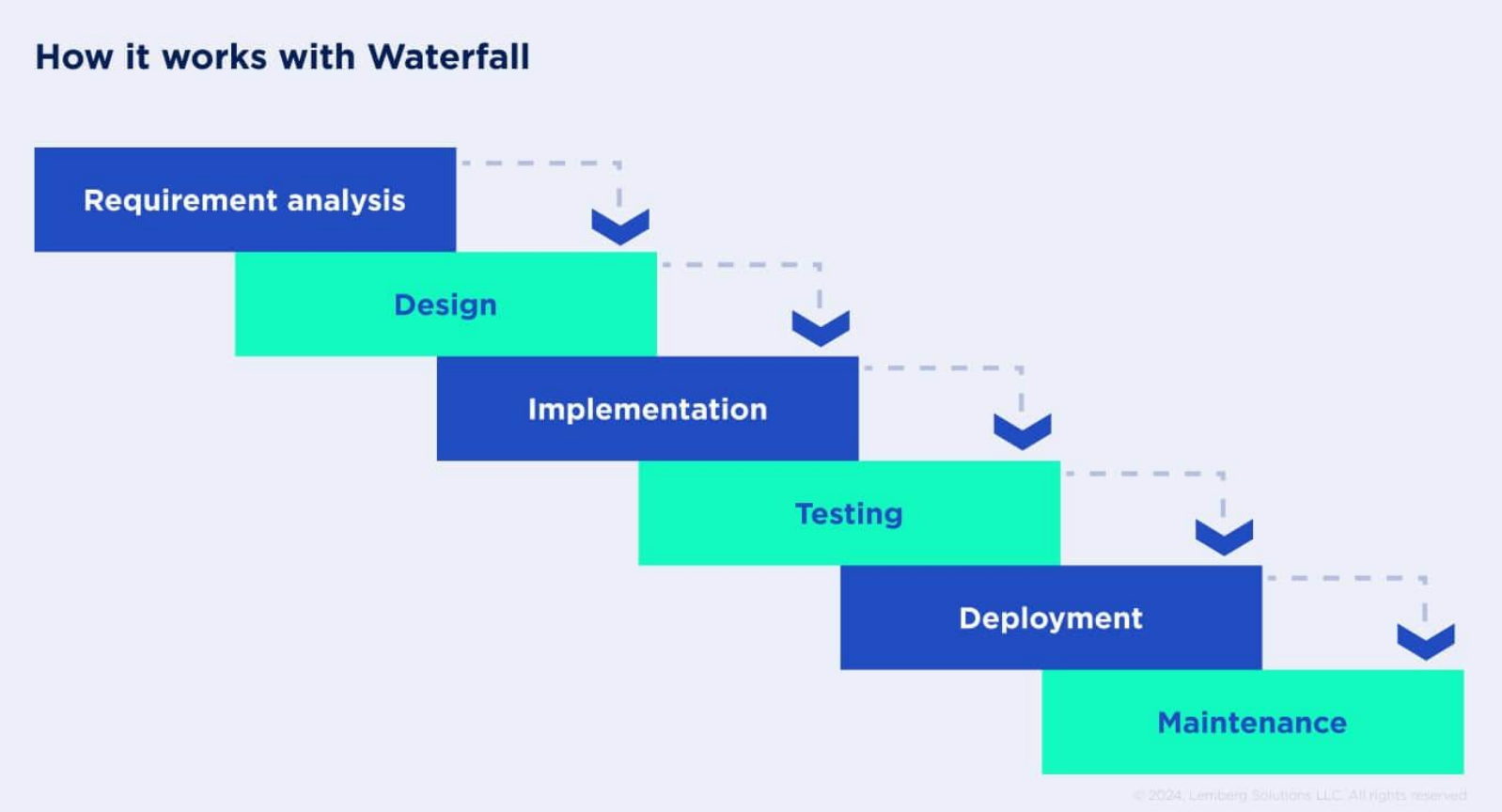
This involves the developers' knowledge and experience with the project domain, software tools, language, and methods needed for development.

Project resources

This involves the amount and availability of funds, staff, and other resources.


SOFTWARE DEVELOPMENT
PRODUCT OWNER
DEVELOPMENT TEAM
CUSTOMERS

Waterfall Development Models



The waterfall model

There are separate identified phases in the waterfall model:

- Requirements analysis and definition
- System and software design
- Implementation and unit testing
- Integration and system testing
- Operation and maintenance

The main drawback of the waterfall model is the difficulty of accommodating change after the process is underway. In principle, a phase has to be complete before moving onto the next phase.

The waterfall model Problems

Inflexible partitioning of the project into distinct stages makes it difficult to respond to changing customer requirements.

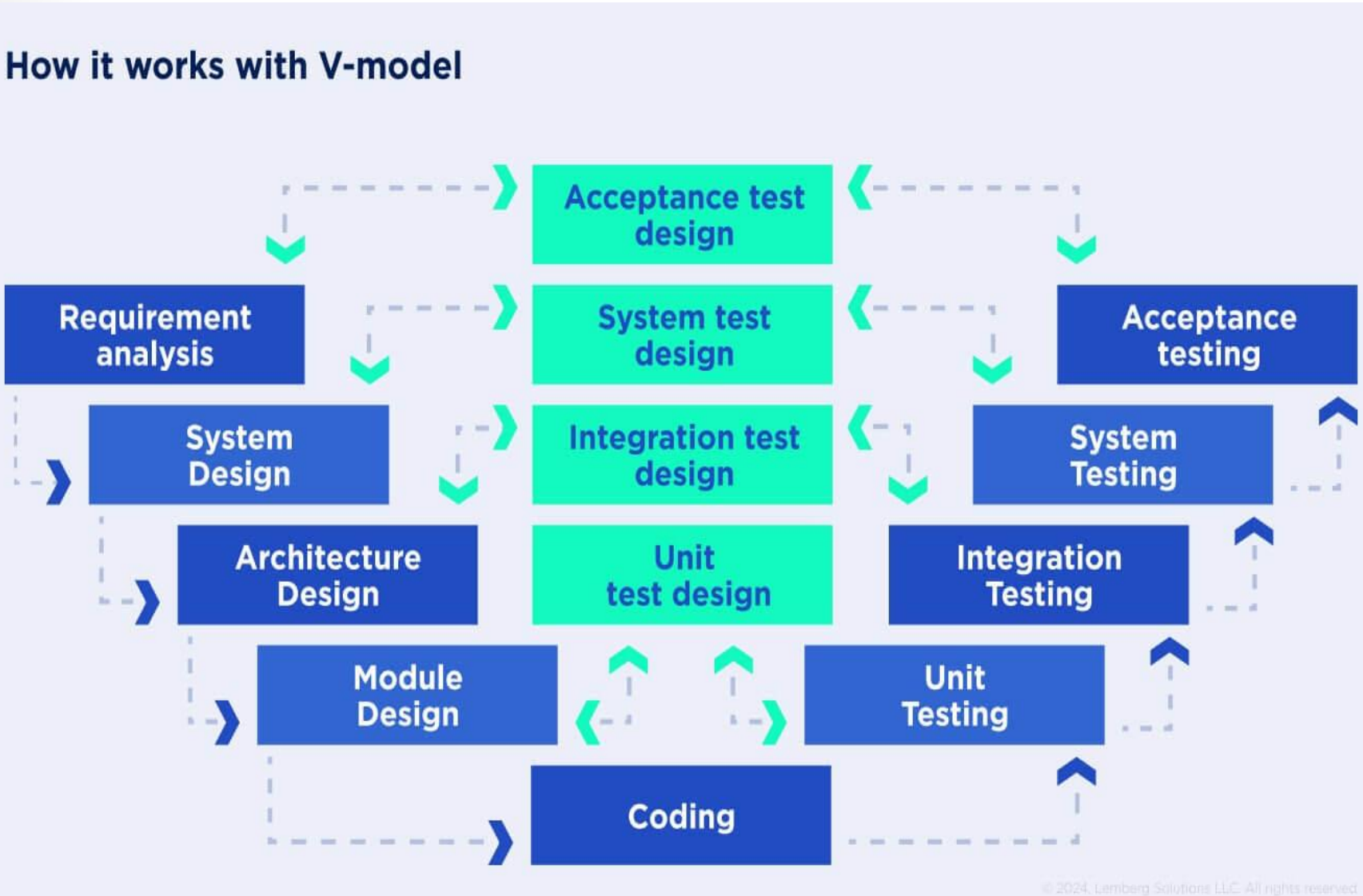
- Therefore, this model is only appropriate when the requirements are well-understood and changes will be fairly limited during the design process.
- Few business systems have stable requirements.

The waterfall model is mostly used for large systems engineering projects where a system is developed at several sites.

- In those circumstances, the plan-driven nature of the waterfall model helps coordinate the work.

https://tryqa.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/

V-model


How it works with V-model
Acceptance test design
Requirement analysis
System test design
Acceptance testing
System Design
Integration test design
System Testing
Architecture Design
Unit test design
Integration Testing
Module Design
Unit Testing
Coding

V-model

- The V-model helps to organize the project in a sequential manner, following the “V” shape of development stages. First, the team needs to do requirement elicitation and analysis. Second, system architecture design and software architecture design, along with specifications for each feature. In parallel, QA team adds acceptance criteria to stakeholder requirements and verification criteria to technical documentation. Both of them are integrated into test cases, ensuring traceability and consistency.
- V-model is also known as the Verification and Validation Model and is based on the Waterfall model. This model emphasizes bidirectional traceability from initial stakeholder requirements to test cases and implemented features, which ensures that the team implements precisely what the stakeholder requests. In this approach, each developmental stage incorporates product unit testing during the first phase, qualification testing of the stand-alone features, integration testing, and so on. The process entails completing a stage, testing the solution, and proceeding to the next stage, as well as updating an architecture design and adding test cases.

V-model

Pros:

- Frequent testing ensures a high chance of project success
- Includes clear requirements work to set a basis for next steps (like architecture)
- Includes integration phase and integration testing as a crucial part of SDLC not considered in other frameworks
- Provides the possibility for almost any SPICE inclusion, which makes it perfect for automotive or medical projects

Cons:

- Lack of flexibility
- No possibility of creating an early prototype
- Higher level of complexity of processes with the possibility of comprehensive configuration management

Use cases:

- Projects with strict requirements and deadlines (e.g., automotive, medical software)
- Complex and large projects where each requirement should be meticulously traced and executed
- Highly regulated industries which require very precise planning and execution in accordance with ISO standards or stakeholder’s requirements

Agile Development Models

The Agile model relies on small, cross-functional teams that develop usable software through rapid, incremental releases, necessitating expertise and adjustments to project management protocols. In essence, Agile is well-suited for projects with dynamic requirements that demand flexibility.

The main advantage and disadvantage of agile methodology are:

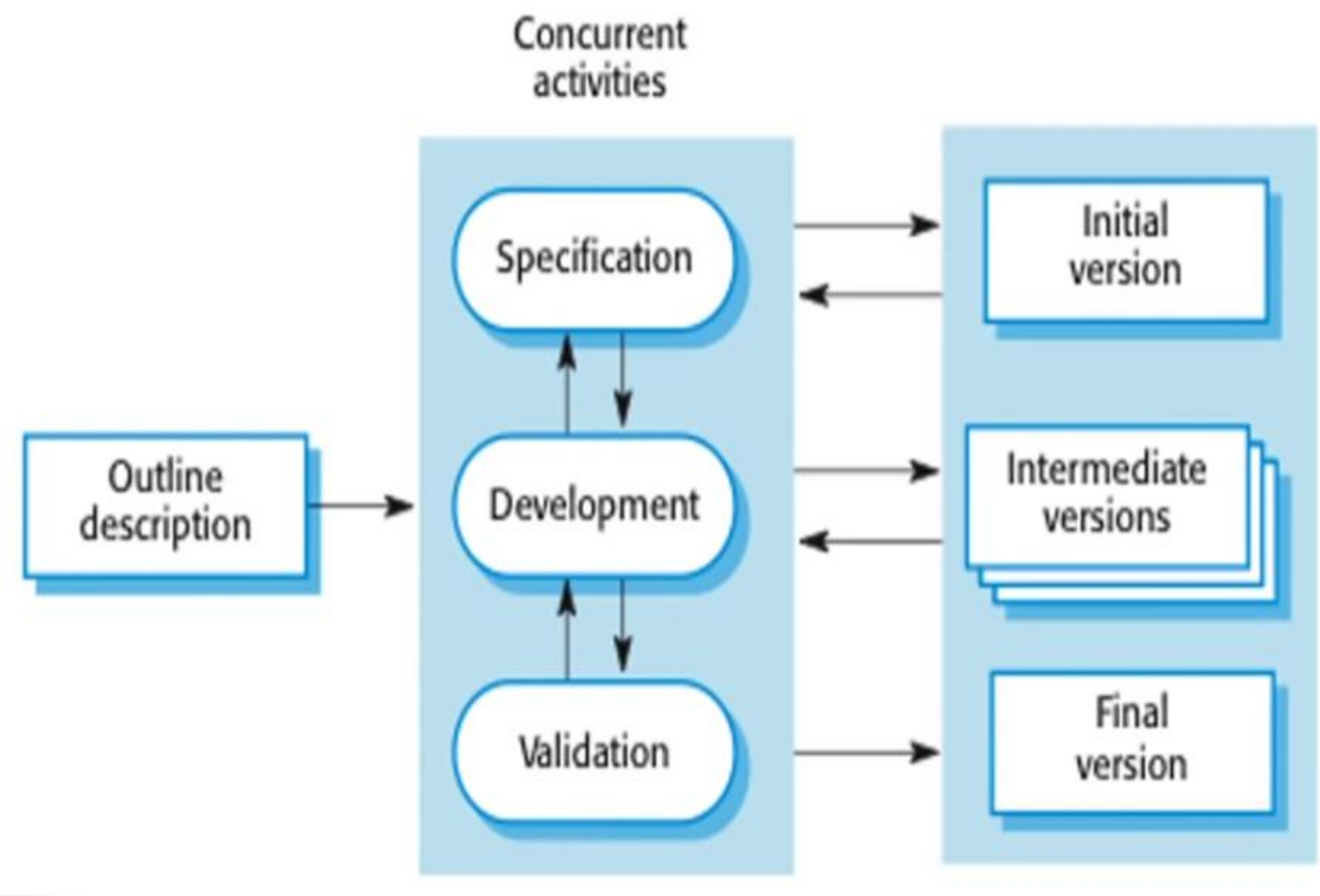
•**Advantage** : Agile methodologies allow for flexibility and adaptability in responding to changes. Teams can easily adjust their plans and priorities based on evolving requirements or feedback during the project.

•**Disadvantage**: The iterative and adaptive nature of agile can sometimes lead to uncertainty, especially in projects with unclear or rapidly changing requirements. This may pose challenges in estimating timelines and costs accurately.

Agile Development Models

- Kanban
- Scrum
- Lean
- Adaptive Project Framework (APF)
- Adaptive Software Development (ASD)
- Dynamic Systems Development Method (DSDM)
- Feature Driven Development (FDD)
- XP or Extreme Programming
- Crystal
- Scaled Agile Framework (SAFe)

Incremental development



Incremental development

- Iterative and incremental development is a process that combines the iterative design method with the incremental build model. It is used by software developers to help manage projects.
- To fully understand the incremental and iterative development process, you must first split it into its two parts:
- **Incremental:** An incremental approach breaks the software development process down into small, manageable portions known as increments. Each increment builds on the previous version so that improvements are made step by step.
- **Iterative:** An iterative model means software development activities are systematically repeated in cycles known as iterations. A new version of the software is produced after each iteration until the optimal product is achieved.

When we use the Incremental Model?

- When the requirements are superior.
- A project has a lengthy development schedule.
- When Software team are not very well skilled or trained.
- When the customer demands a quick release of the product.
- You can develop prioritized requirements first.

Advantage and disadvantages of Incremental Model

Advantage of Incremental Model

- Errors are easy to be recognized.
- Easier to test and debug
- More flexible.
- Simple to manage risk because it handled during its iteration.
- The Client gets important functionality early.

Disadvantage of Incremental Model

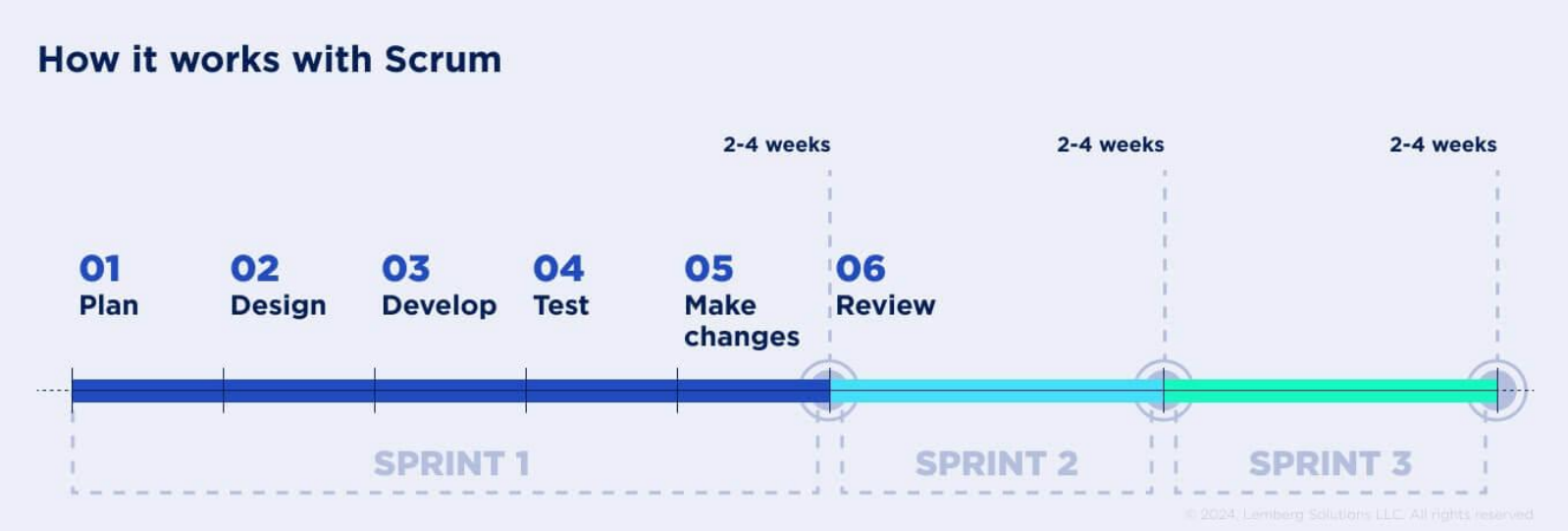
- Need for good planning
- Total Cost is high.
- Well defined module interfaces are needed.

Scrum

Scrum is Everywhere

90% - Estimated Agile Teams use Scrum
12M+ Estimated Using Scrum Daily
Practiced Everywhere

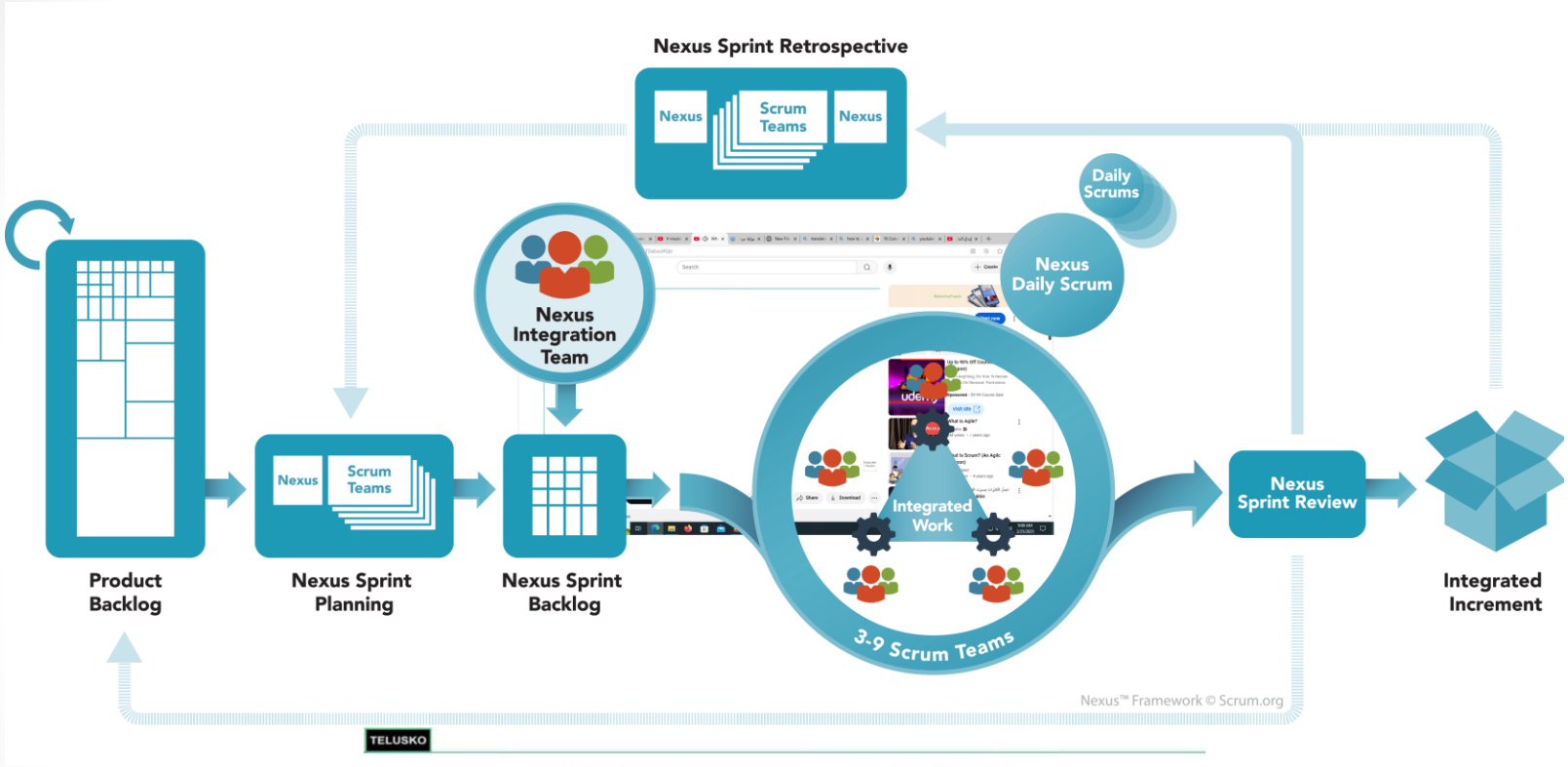
One Scrum Guide

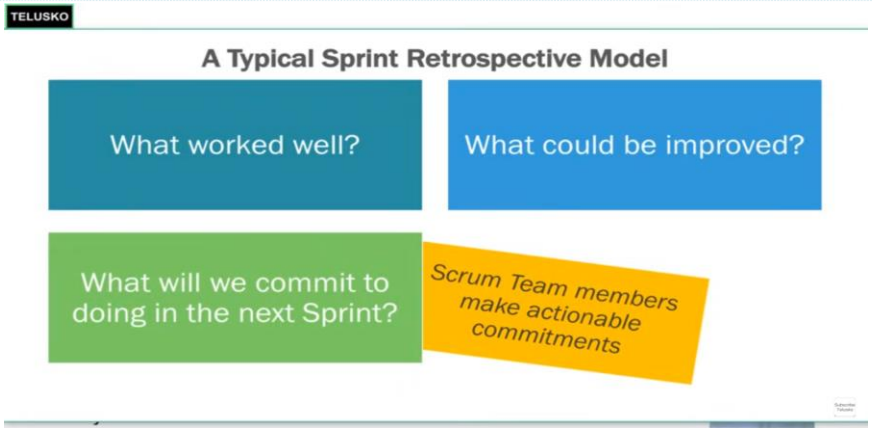


Agile admirers usually choose Scrum as a main project management approach. The team thoroughly plans and assesses each sprint. Usually, sprint duration takes 2 to 4 weeks. However, using Scrum, the team cannot change requirements for the sprint. The teams are usually self-managed with specific roles assigned.

Scrum is best employed when immediate results are needed, amidst ambiguous situations with undefined tasks, or when a client requests a highly tailored development approach for a specific product.

Scrum framework diagram





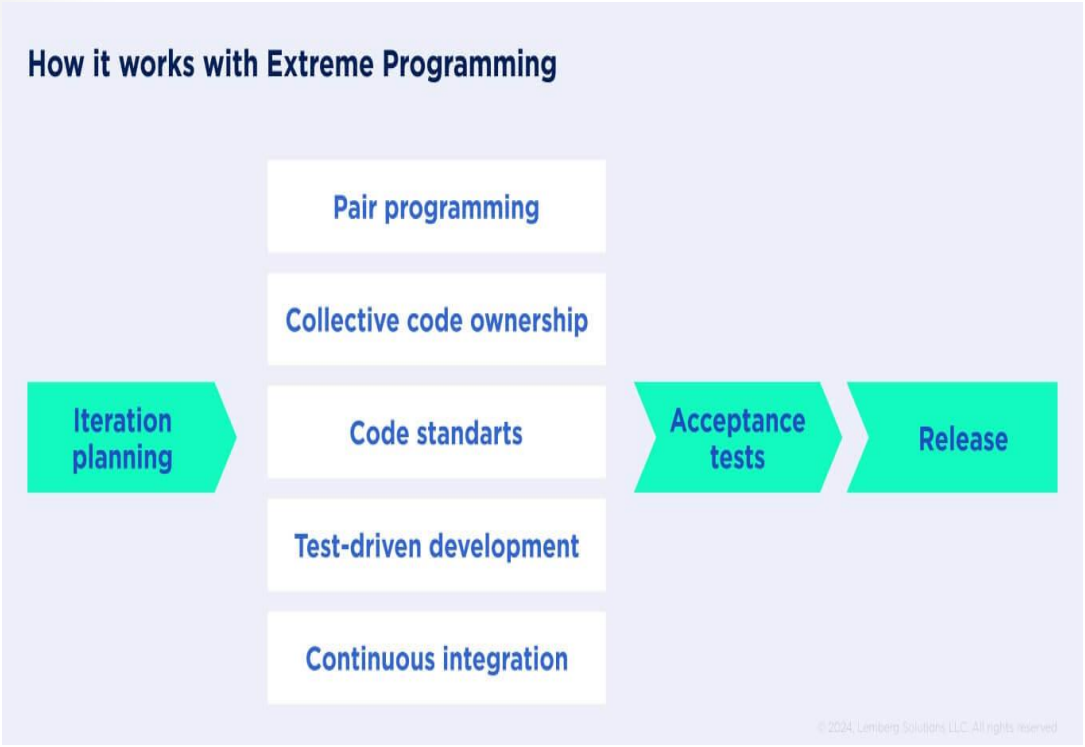
Scrum

Pros:
•Large projects are split into easily controlled sprints
•Ongoing deliverables are tested after each sprint
•Suitable for projects with tight deadlines
•Ensures development transparency for the team
•Allows for adjustment from stakeholders and clients

Cons:
•A project’s scope frequently increases due to the absence of a specific completion date
•Doesn’t work well for large teams
•Requires senior experience from the team members

Use cases:
•Flexible projects with evolving requirements
•Projects that depend on a collaborative team

Extreme Programming (XP)



An XP approach includes 1-2 week sprints. The team can implement changes even if the sprint has already started. However, constant changes can impede the successful delivery of a solution. That's why the team should pay attention to software testing, CI/CD practices, and simple software design.

Extreme Programming (XP)

- This approach is suitable for scenarios demanding continuous client-developer communication, swift adaptation to change, flexible planning, prioritizing working software, measuring success by meeting client needs and team efforts, and enabling remote project collaboration.
- XP model is best suitable for real-time collaboration applications like Google Docs, Trello, or Slack. These apps benefit from XP's emphasis on quick iterations and constant communication.
- Additionally, XP's emphasis on testing and adaptability suits the needs of financial trading apps. These platforms require reliability, quick updates, and responsiveness to market changes, making XP an ideal approach.

Extreme Programming (XP)

Pros:

- Allows for close cooperation with the client
- Eliminates the need for unnecessary development
- Ensures stable results due to continuous testing
- Provides flexibility for the teams, eliminating any overtime
- Allows for last-minute changes

Cons:

- Requires additional work in terms of frequent testing and CI
- It is necessary for the client to be involved in the process
- Requires a lot of time
- Necessitates version management

Use cases:

- Projects with frequent requirement changes
- Projects requiring a small development team
- Projects using technologies that allow for unit and functional testing

Kanban

Kanban doesn’t focus on well-defined iterations. Even if there are any, Kanban requires to make them short, in the form of daily sprints. The primary focus is on the project visualization. Utilizing tools like the Kanban Board, the team creates a visual depiction of project progress, including tasks, project roles, and activities. Kanban increases project transparency and allows for more effective prioritization. The model also enables changes at any sprint. Continuous communication with the client enables them to review work results and set up regular meetings with the team.

Kanban

Pros:

- Easy to implement any changes and fix the issues
- The team doesn’t rely on a rigid plan or detailed project requirements since Agile approach allows for updates
- The development phase is usually short
- Teams can easily organize and move through the stages
- Early user feedback allows to implement needed changes as soon as possible and adjust the product to the market demand
- Better alignment with client expectations since they are more involved in the development process

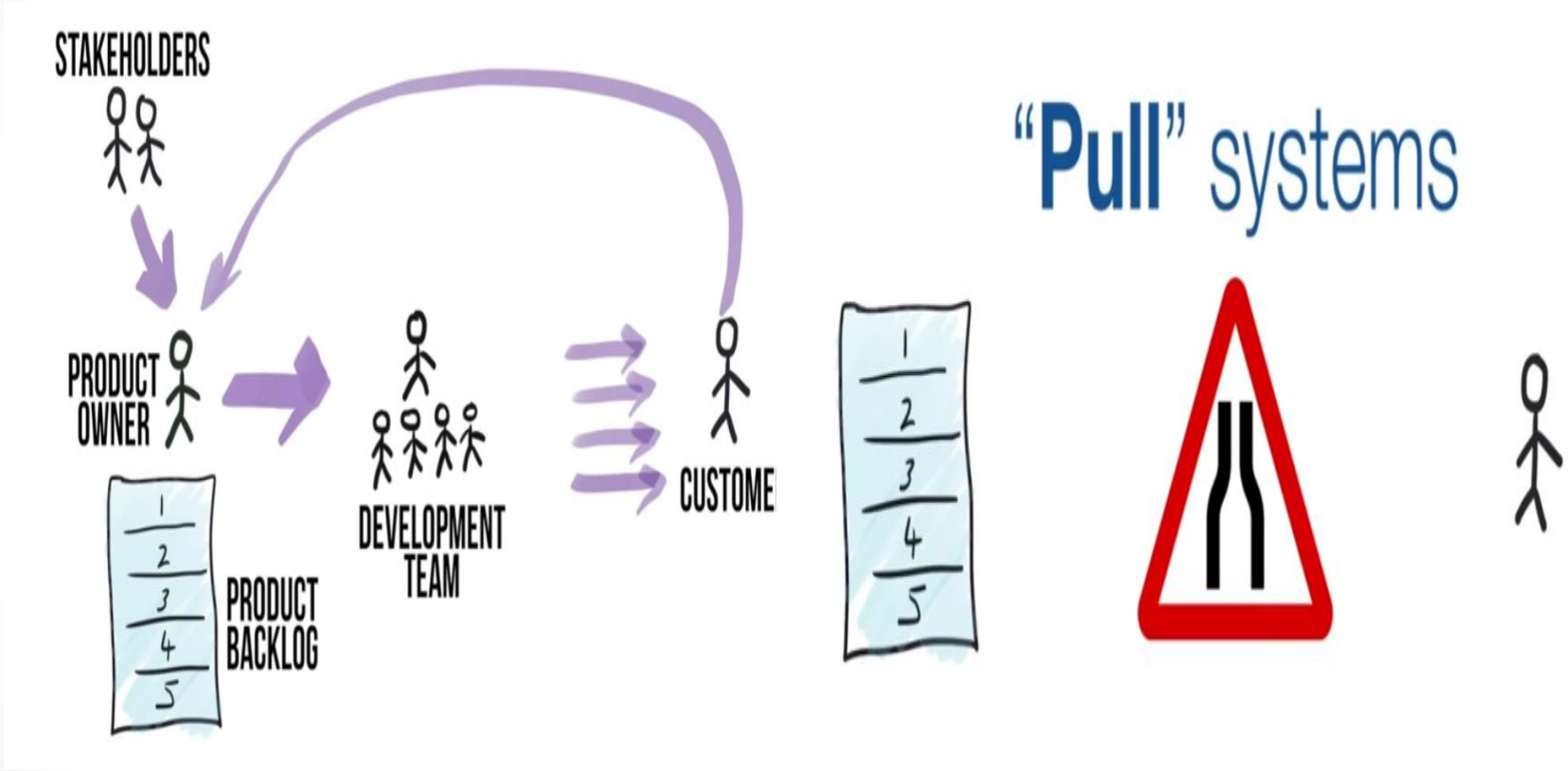
Cons:

- Hard to estimate the final project budget
- Hard to set strict timelines
- Documentation requires more attention from the team
- The final product can be hard to assemble
- Difficult to measure the progress due to the floating KPIs

Use cases:

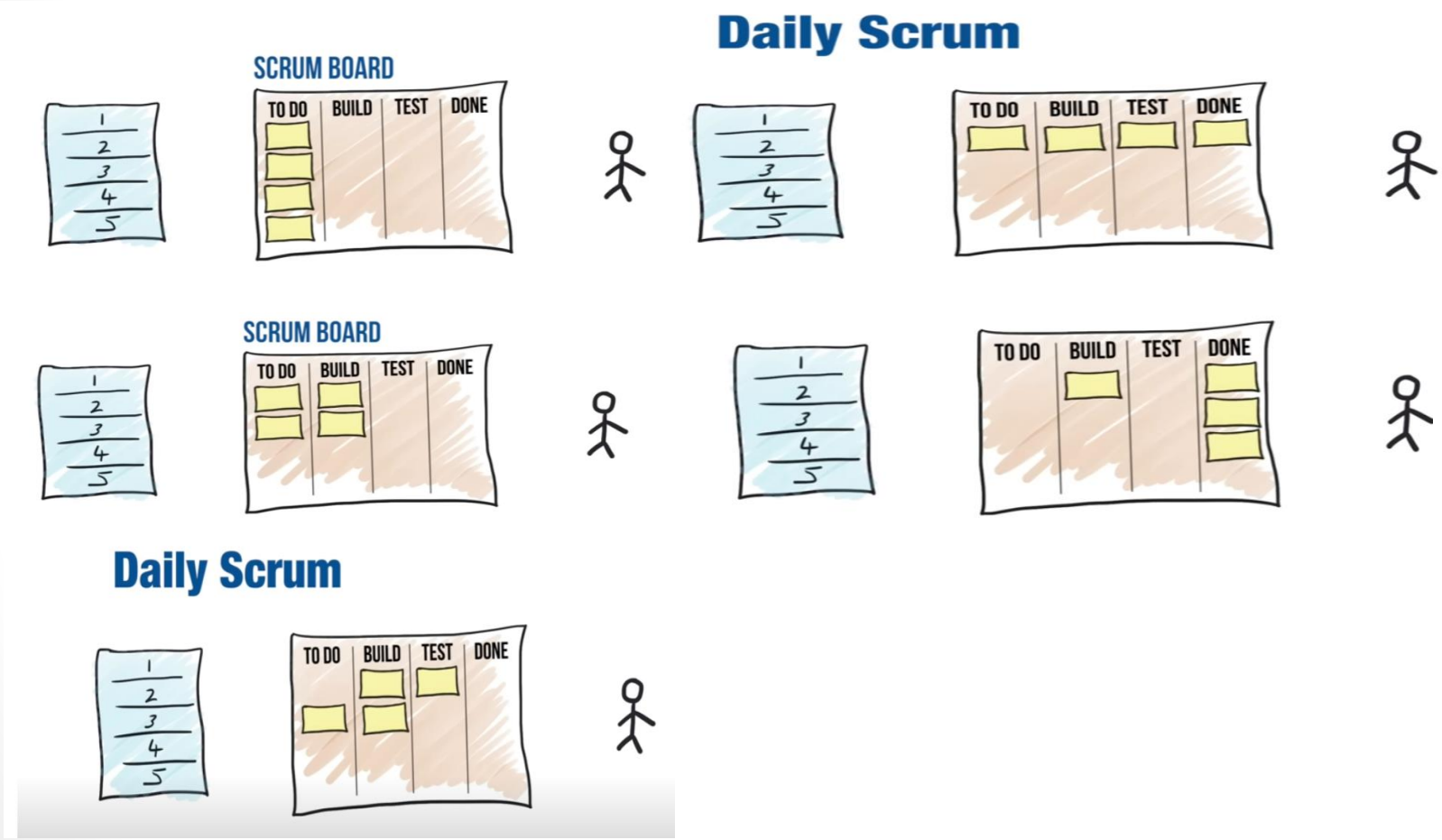
- Ideal for startups since the early feedback from end users is a must
- Turnkey projects with custom features and lack of precise requirements
- Projects that can be divided into several development phases

Scrum Vs. Kanban

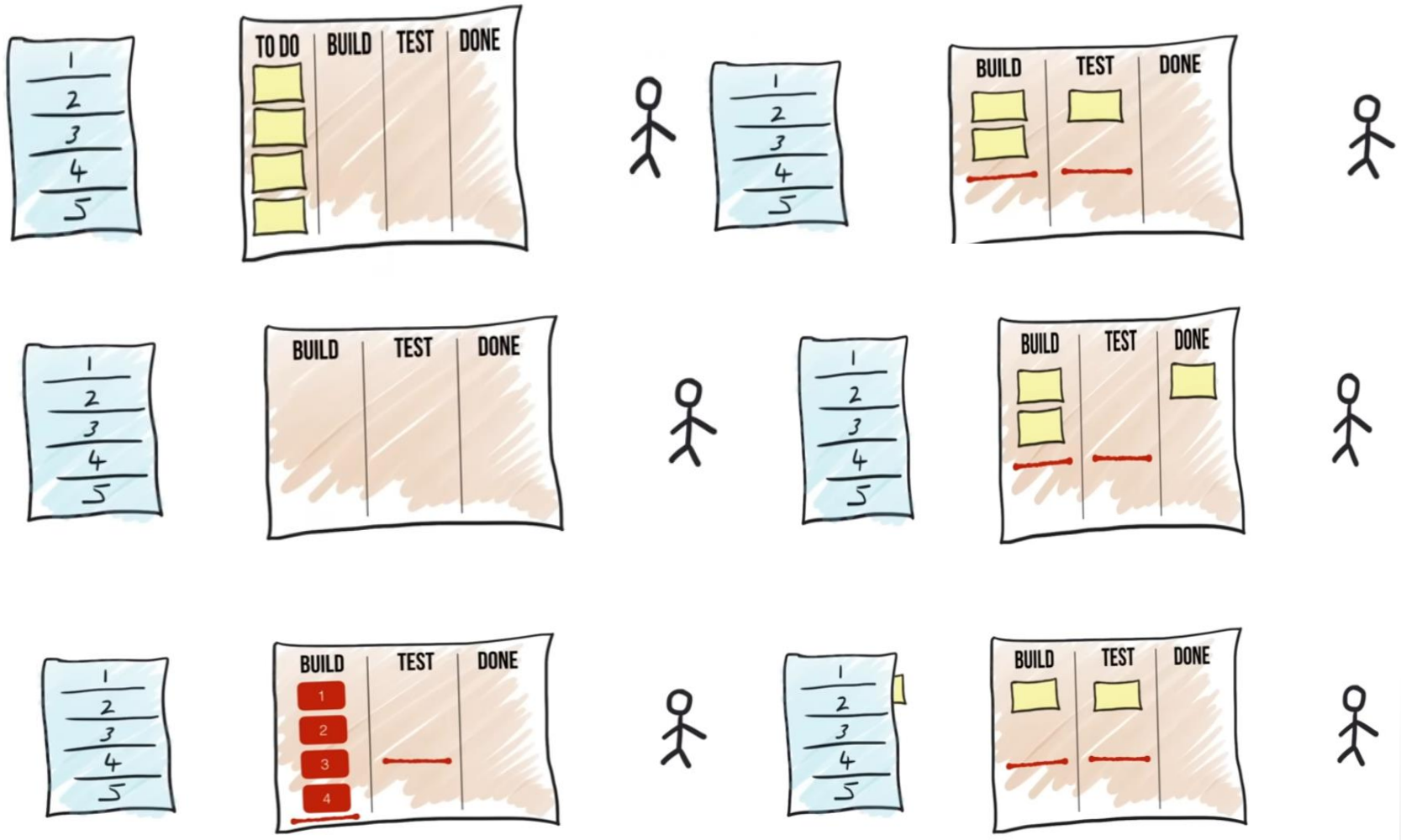


What happen between the product backlog and the stakeholders is the difference between Scrum and Kanban

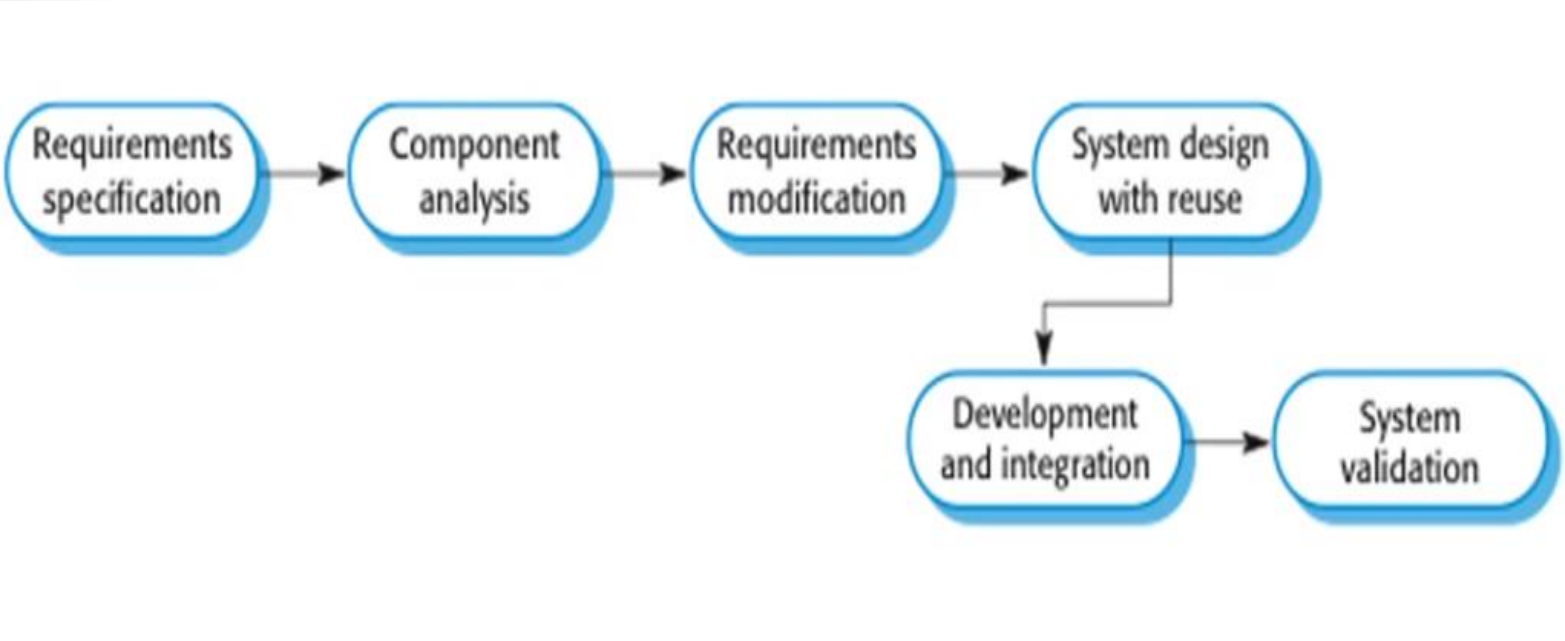
Scrum Board


Daily Scrum
SCRUM BOARD
TO DO
BUILD
TEST
DONE
1
2
3
4
5
SCRUM BOARD
TO DO
BUILD
TEST
DONE
Daily Scrum
TO DO
BUILD
TEST
DONE

Kanaban


1
2
3
4
5
TO DO
BUILD
TEST
DONE
BUILD
TEST
DONE
BUILD
TEST
DONE
BUILD
TEST
DONE
1
2
3
4
BUILD
TEST
DONE

Reuse Oriented Model



https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=116955862820be5e1f46d3a4bce4b30db09b70cd

Reuse Oriented Model

Reuse Oriented Model (ROM), also known as reuse-oriented development (ROD), it can be steps of the software development for specific duration in which software is redesigned through creating a sequence of prototypes known as models, every system is derived from the previous one with constant series of defined rules.

The reuse-oriented model isn't always sensible in its pure form due to cause of an entire repertoire of reusable additives that might not be available. In such cases, several new system components need to be designed. If it is not done, ROM has to compromise in perceived requirements, leading to a product that does not meet exact requirements of user. This model depends upon perception that maintenance might be viewed as a pastime involving reuse of existing system components.

Reuse Oriented Model

The reuse model has 4 **fundamental steps** which are followed :

- To identify components of old system that are most suitable for reuse.
- To understand all system components.
- To modify old system components to achieve new requirements.
- To integrate all of modified parts into new system.

A specific framework is required for categorization of components and consequently required modification. The complete reuse version may begin from any segment of the existence cycle – need, planning, code, design, or analyze data – not like other models.

Reuse Oriented Model

Advantages :

- It can reduce total cost of software development.
- The risk factor is very low.
- It can save lots of time and effort.
- It is very efficient in nature.

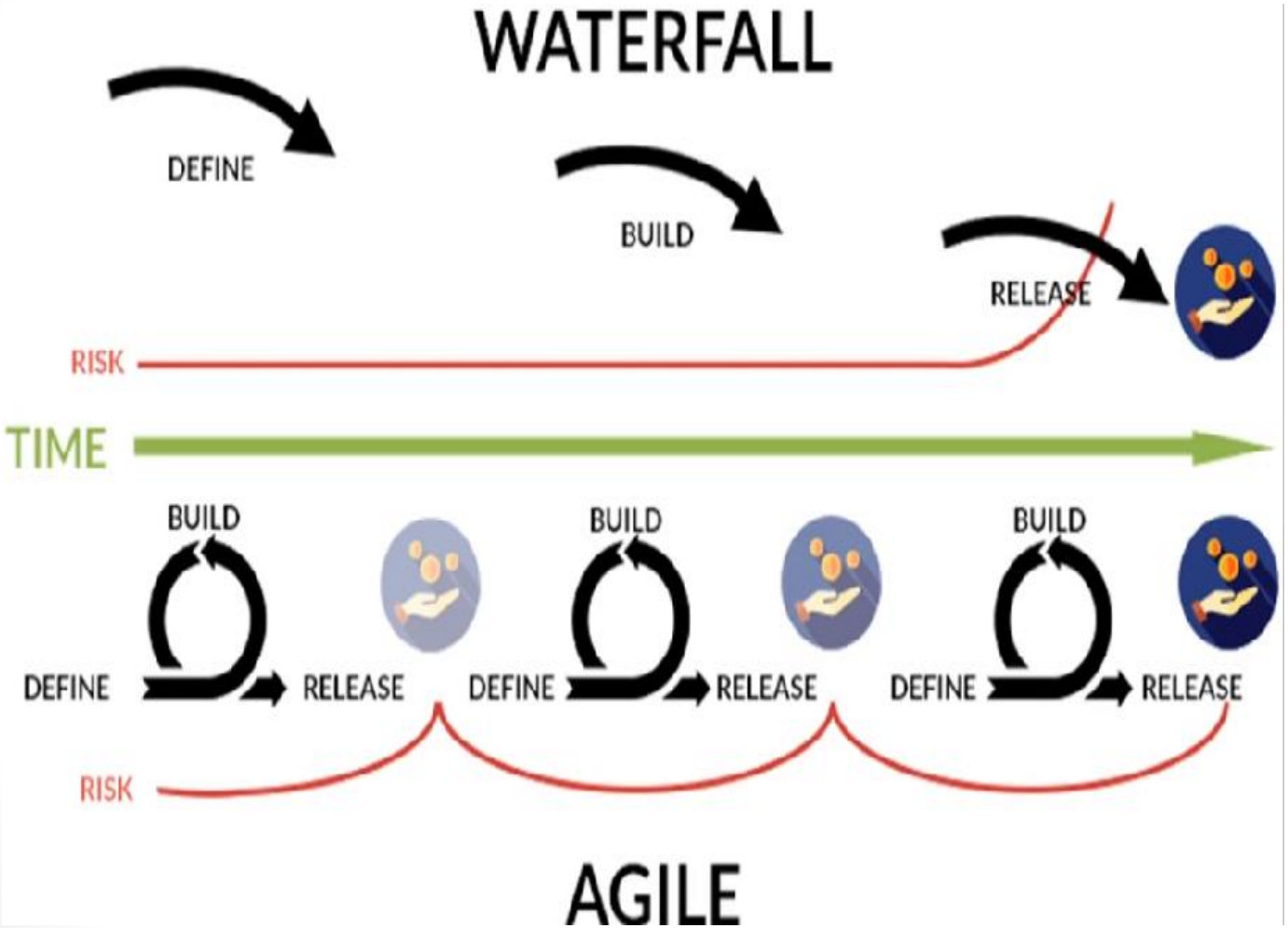
Disadvantages :

- Reuse-oriented model is not always worked as a practice in its true form.
- Compromises in requirements may lead to a system that does not fulfill requirement of user.
- Sometimes using old system component, that is not compatible with new version of component, this may lead to an impact on system evolution.

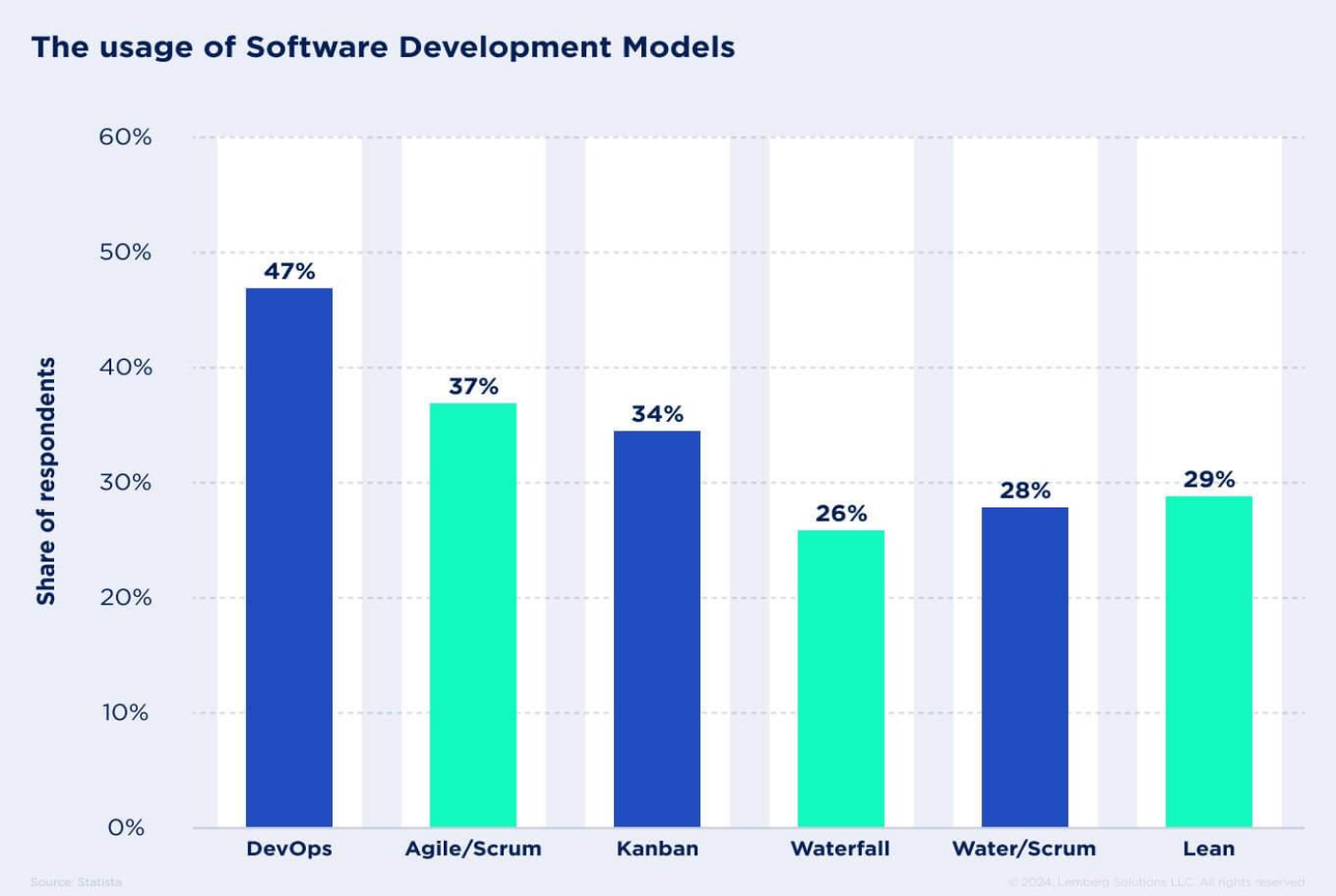
Plan-driven and agile processes

- **A plan-driven development is** a process where all the activities are planned in advance, and we measure the progress in advance against that plan. So as it appears from the term ‘plan-driven development, it is always planned, and the results are to be shown at the end of the product.
- **Agile development,** planning is always incremental, where you can change the plan according to your customer requirements. So, as the term ‘agile development’ suggests, it is the process-based method where one can alter the way things and planning and replace them with the more suitable ones.
- https://novateus.com/blog/how-is-plan-driven-development-different-from-agile-development/

Waterfall against Agile



According to Statista, 37 % of the companies prefer using the Agile/Scrum model, 34% apply Kanban, and 26% follow the Waterfall model. Read below to learn more about those and other software development life cycle models



https://www.mediafire.com/file/nooez408mtcz23w/fff.pdf/file

Questions??