Software development life cycle models

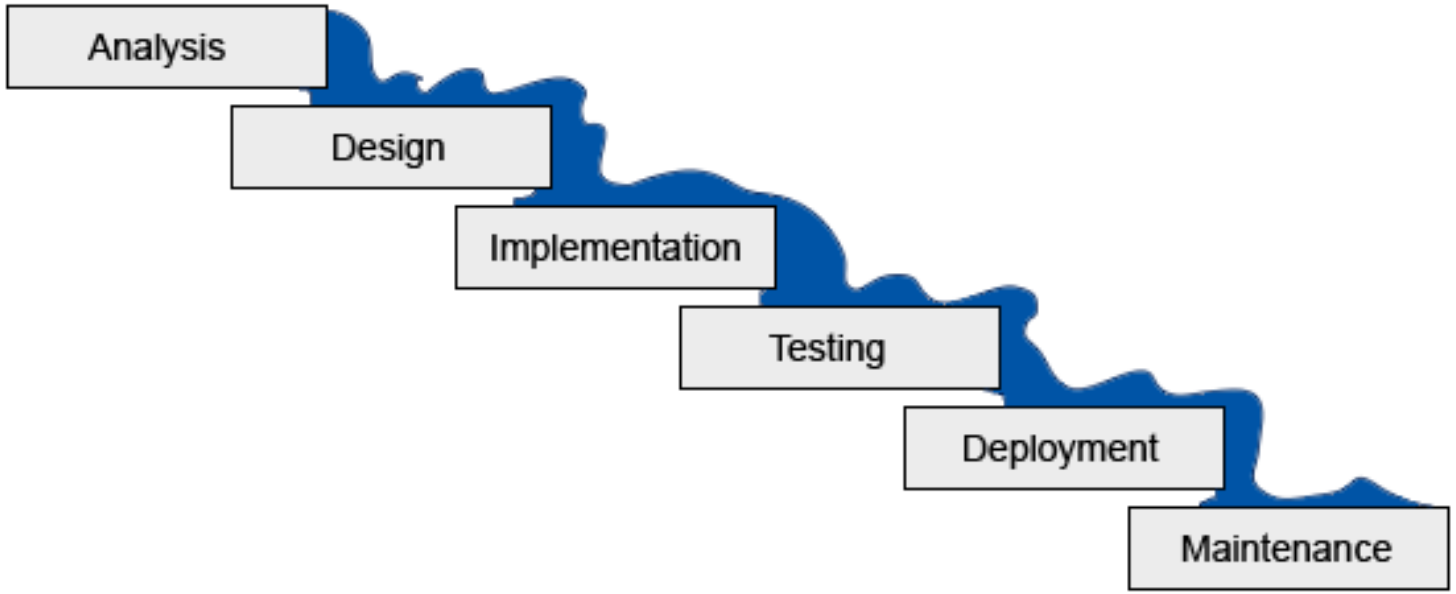
Rakibul Hassan
Lecturer
Dept. of ECE, RUET

SDLC Models

Some popular models that are followed during software development process are:

- Waterfall model
- Iterative model
- Incremental model
- Spiral model
- Agile model
- V model

Waterfall model



- It was introduced in 1970 by Winston Royce.
- It is a sequential model that divides software development into pre-defined phases.
- Each phase must be completed before the next phase can begin with no overlap between the phases.
- Each phase is designed for performing specific activity during the SDLC phase.

Waterfall model

When to use SDLC Waterfall Model?

- Requirements are not changing frequently
- Application is not complicated and big
- Project is short
- Requirement is clear
- Environment is stable
- Technology and tools used are not dynamic and is stable
- Resources are available and trained

Waterfall model

Advantages	Dis-Advantages
❑ Before the next phase of development, each phase must be completed	❑ Error can be fixed only during the phase
❑ Suited for smaller projects where requirements are well defined	❑ It is not desirable for complex project where requirement changes frequently
❑ They should perform quality assurance test (Verification and Validation) before completing each stage	❑ Testing period comes quite late in the developmental process
❑ Elaborate documentation is done at every phase of the software's development cycle	❑ Documentation occupies a lot of time of developers and testers
❑ Project is completely dependent on project team with minimum client intervention	❑ Clients valuable feedback cannot be included with ongoing development phase
❑ Any changes in software is made during the process of the development	❑ Small changes or errors that arise in the completed software may cause a lot of problems

Iterative model

- We can start with some of the software specifications and develop the first version of the software.
- After the first version if there is a need to change the software, then a new version of the software is created with a new iteration.
- Every release of the Iterative Model finishes in an exact and fixed period that is called iteration.
- The Iterative Model allows the accessing earlier phases, in which the variations made respectively.
- The final output of the project renewed at the end of the Software Development Life Cycle (SDLC) process.

Iterative model

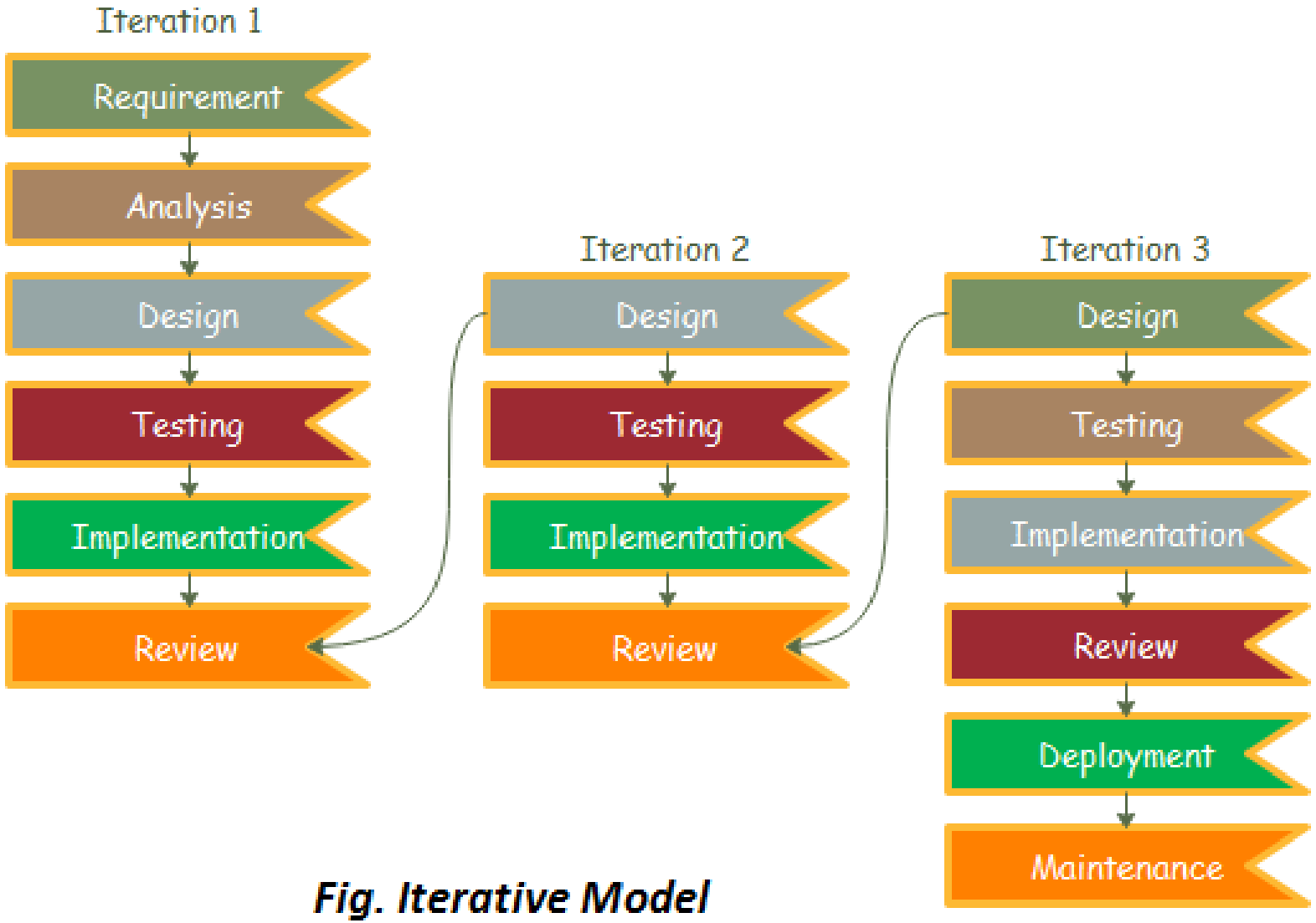


Fig. Iterative Model

Iterative model

When to use the Iterative Model?

- When requirements are defined clearly and easy to understand.
- When the software application is large.
- When there is a requirement of changes in future.

Iterative model

Advantages of Iterative Model:

- Testing and debugging during smaller iteration is easy.
- A Parallel development can plan.
- It is easily acceptable to ever-changing needs of the project.
- Risks are identified and resolved during iteration.
- Limited time spent on documentation and extra time on designing.

Iterative model

Disadvantages of Iterative Model:

- It is not suitable for short projects.
- More Resources may be required.
- Design can be changed again and again because of imperfect requirements.
- Requirement changes can cause over budget.
- Project completion date not confirmed because of changing requirements.

Incremental model

- Incremental Model is a process of software development where requirements divided into multiple standalone modules of the software development cycle.
- In this model, each module goes through the requirements, design, implementation and testing phases.
- Every subsequent release of the module adds function to the previous release.
- The process continues until the complete system achieved.

Incremental model

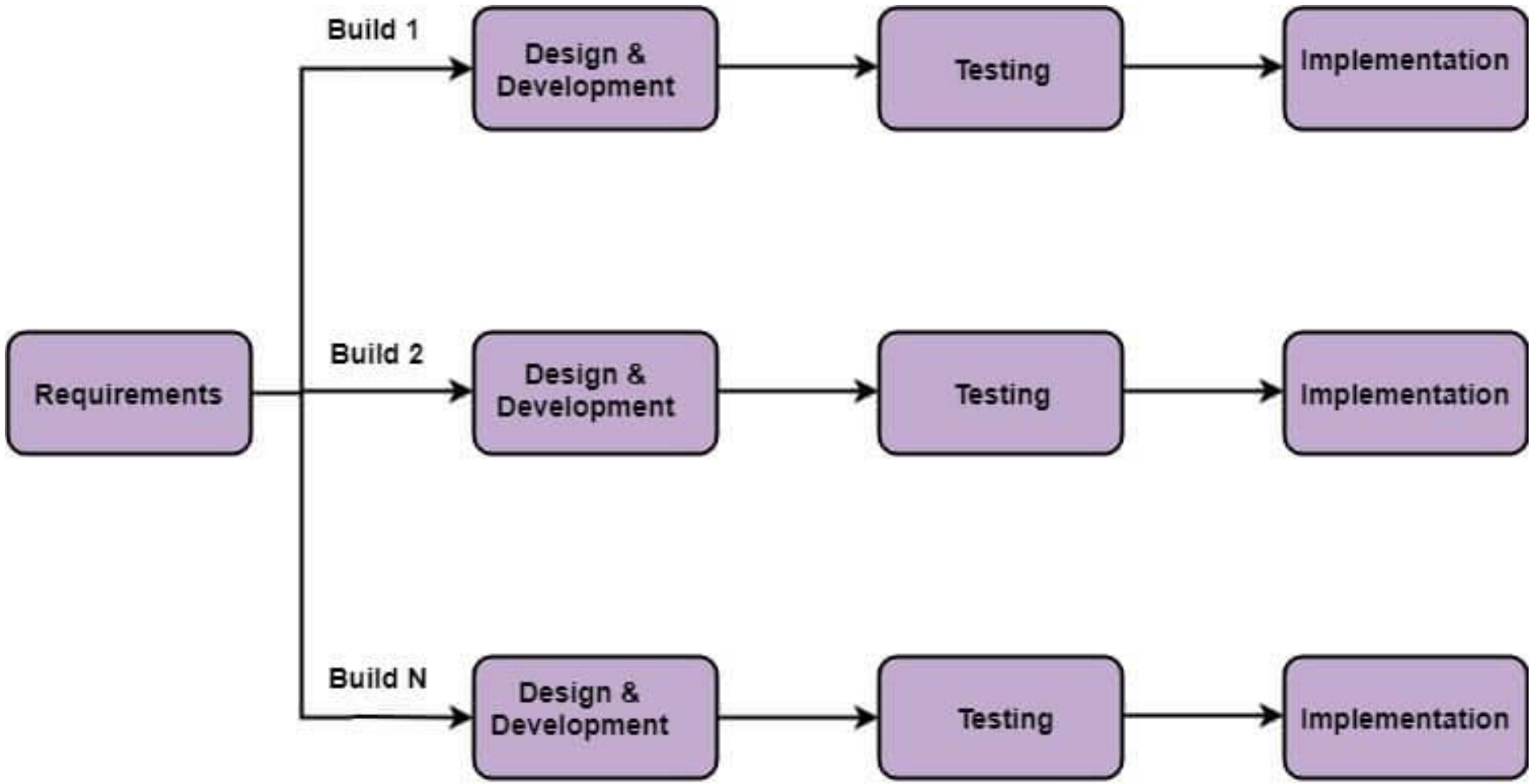


Fig: Incremental Model

Incremental model

When to use Incremental models?

- Requirements of the system are clearly understood
- When demand for an early release of a product arises
- When software engineering team are not very well skilled or trained
- When high-risk features and goals are involved
- Such methodology is more in use for web application and product based companies

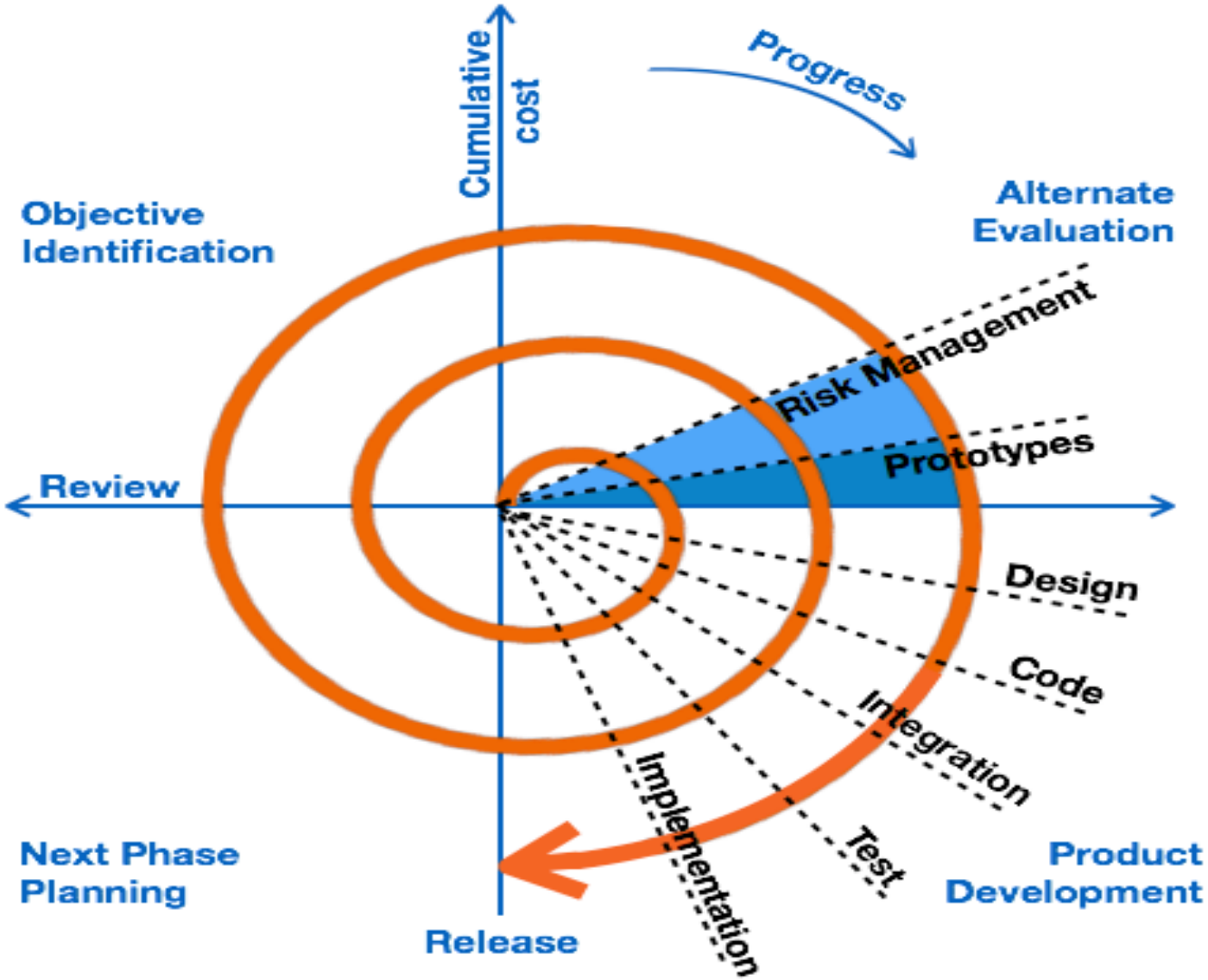
Incremental model

Advantages	Disadvantages
• The software will be generated quickly during the software life cycle	• It requires a good planning designing
• It is flexible and less expensive to change requirements and scope	• Problems might cause due to system architecture as such not all requirements collected up front for the entire software lifecycle
• Throughout the development stages changes can be done	• Each iteration phase is rigid and does not overlap each other
• This model is less costly compared to others	• Rectifying a problem in one unit requires correction in all the units and consumes a lot of time
• A customer can respond to each building	
• Errors are easy to be identified	

Spiral model

- Spiral model is a combination of both, iterative model and one of the SDLC model.
- It can be seen as if we choose one SDLC model and combine it with cyclic process (iterative model).
- This model considers **risk**, which often goes un-noticed by most other models.
- The model starts with determining objectives and constraints of the software at the start of one iteration.
- Next phase is of prototyping the software. This includes risk analysis.
- Then one standard SDLC model is used to build the software. In the fourth phase of the plan of next iteration is prepared.

Spiral model



Spiral model

When to use Spiral Model?

- When deliverance is required to be frequent.
- When the project is large
- When requirements are unclear and complex
- When changes may require at any time
- Large and high budget projects

Spiral model

Advantages	Disadvantages
• Additional functionality or changes can be done at a later stage	• Risk of not meeting the schedule or budget
• Cost estimation becomes easy as the prototype building is done in small fragments	• Spiral development works best for large projects only also demands risk assessment expertise
• Continuous or repeated development helps in risk management	• For its smooth operation spiral model protocol needs to be followed strictly
• Development is fast and features are added in a systematic way in Spiral development	• Documentation is more as it has intermediate phases
• There is always a space for customer feedback	• Spiral software development is not advisable for smaller project, it might cost them a lot

Big Bang model

- In this model, developers do not follow any specific process.
- Development begins with the necessary funds and efforts in the form of inputs.
- The result may or may not be as per the customer's requirement, because in this model, even the customer requirements are not defined.
- This model is ideal for small projects like academic projects or practical projects.
- One or two developers can work together on this model.

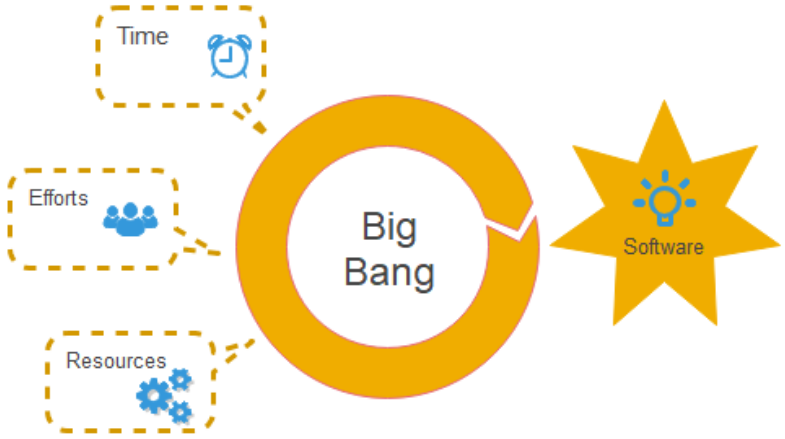


Fig. Big Bang Model

Big Bang model

When to use Big Bang Model?

- This model is required when this project is small like an academic project or a practical project.
- This method is also used when the size of the developer team is small and when requirements are not defined, and the release date is not confirmed or given by the customer.

Big Bang model

Advantage of Big Bang Model:

- There is no planning required.
- Simple Model.
- Few resources required.
- Easy to manage.
- Flexible for developers.

Disadvantage(Cons) of Big Bang Model:

- There are high risk and uncertainty.
- Not acceptable for a large project.
- If requirements are not clear that can cause very expensive.

RAD MODEL

RAD (Rapid Application development) is a linear sequential software development process model that emphasizes a concise development cycle using an element based construction approach. If the requirements are well understood and described, and the project scope is a constraint, the RAD process enables a development team to create a fully functional system within a concise time period.

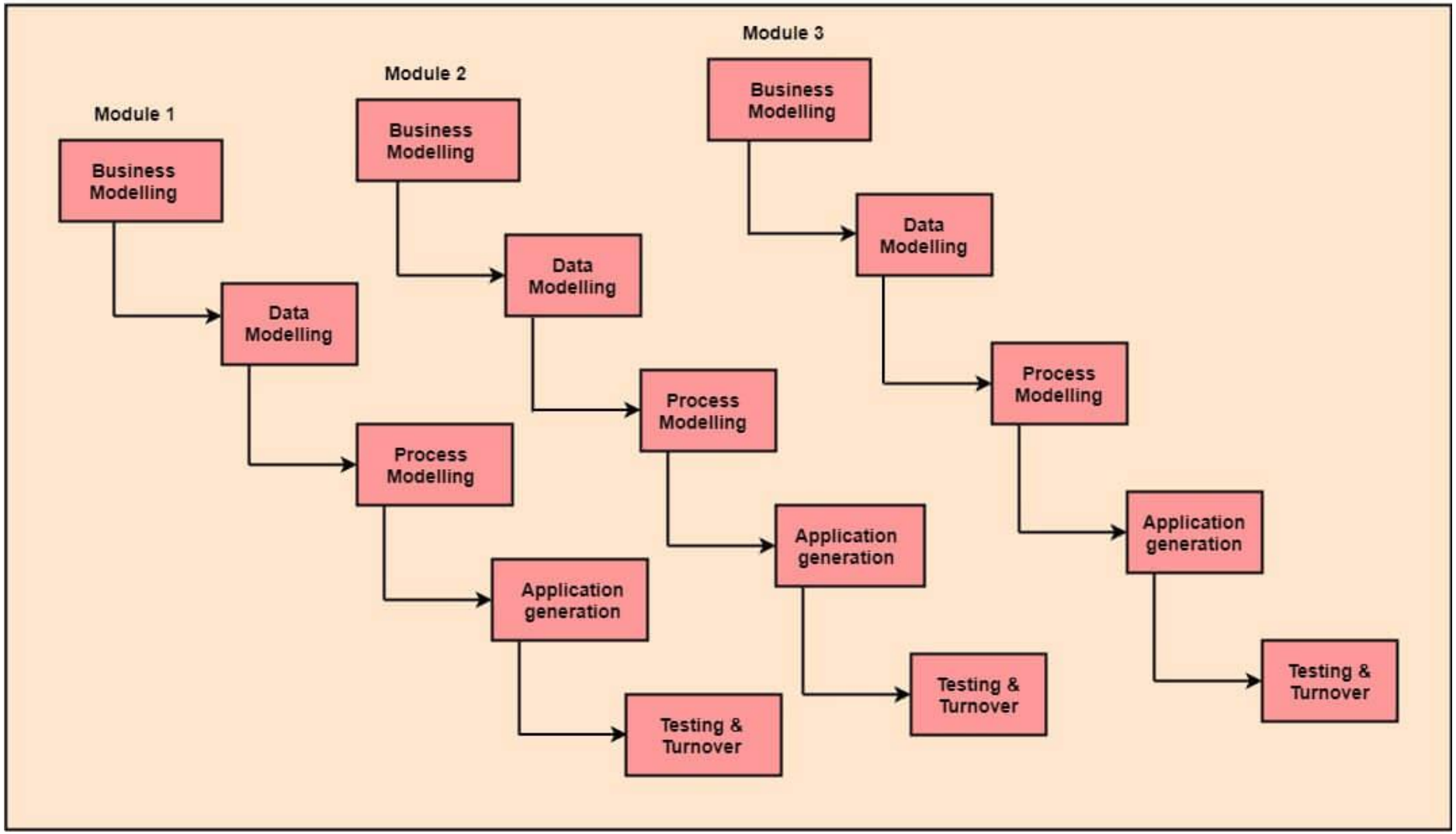
RAD MODEL

RAD (Rapid Application Development) is a concept that products can be developed faster and of higher quality through:

- Gathering requirements using workshops or focus groups
- Prototyping and early, reiterative user testing of designs
- The re-use of software components
- A rigidly paced schedule that refers design improvements to the next product version
- Less formality in reviews and other team communication

RAD MODEL

Fig: RAD Model



RAD MODEL

The various phases of RAD are as follows:

1.Business Modelling: The information flow among business functions is defined by answering questions like what data drives the business process, what data is generated, who generates it, where does the information go, who process it and so on.

2. Data Modelling: The data collected from business modeling is refined into a set of data objects (entities) that are needed to support the business. The attributes (character of each entity) are identified, and the relation between these data objects (entities) is defined.

RAD MODEL

3. Process Modelling: The information object defined in the data modeling phase are transformed to achieve the data flow necessary to implement a business function. Processing descriptions are created for adding, modifying, deleting, or retrieving a data object.

4. Application Generation: Automated tools are used to facilitate construction of the software; even they use the 4th GL techniques.

5. Testing & Turnover: Many of the programming components have already been tested since RAD emphasis reuse. This reduces the overall testing time. But the new part must be tested, and all interfaces must be fully exercised.

RAD MODEL

When to use RAD Model?

- When the system should need to create the project that modularizes in a short span time (2-3 months).
- When the requirements are well-known.
- When the technical risk is limited.
- When there's a necessity to make a system, which modularized in 2-3 months of period.
- It should be used only if the budget allows the use of automatic code generating tools.

AGILR MODEL

- Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product.

- Agile Methods break the product into small incremental builds. These builds are provided in iterations.

- Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas

AGILR MODEL

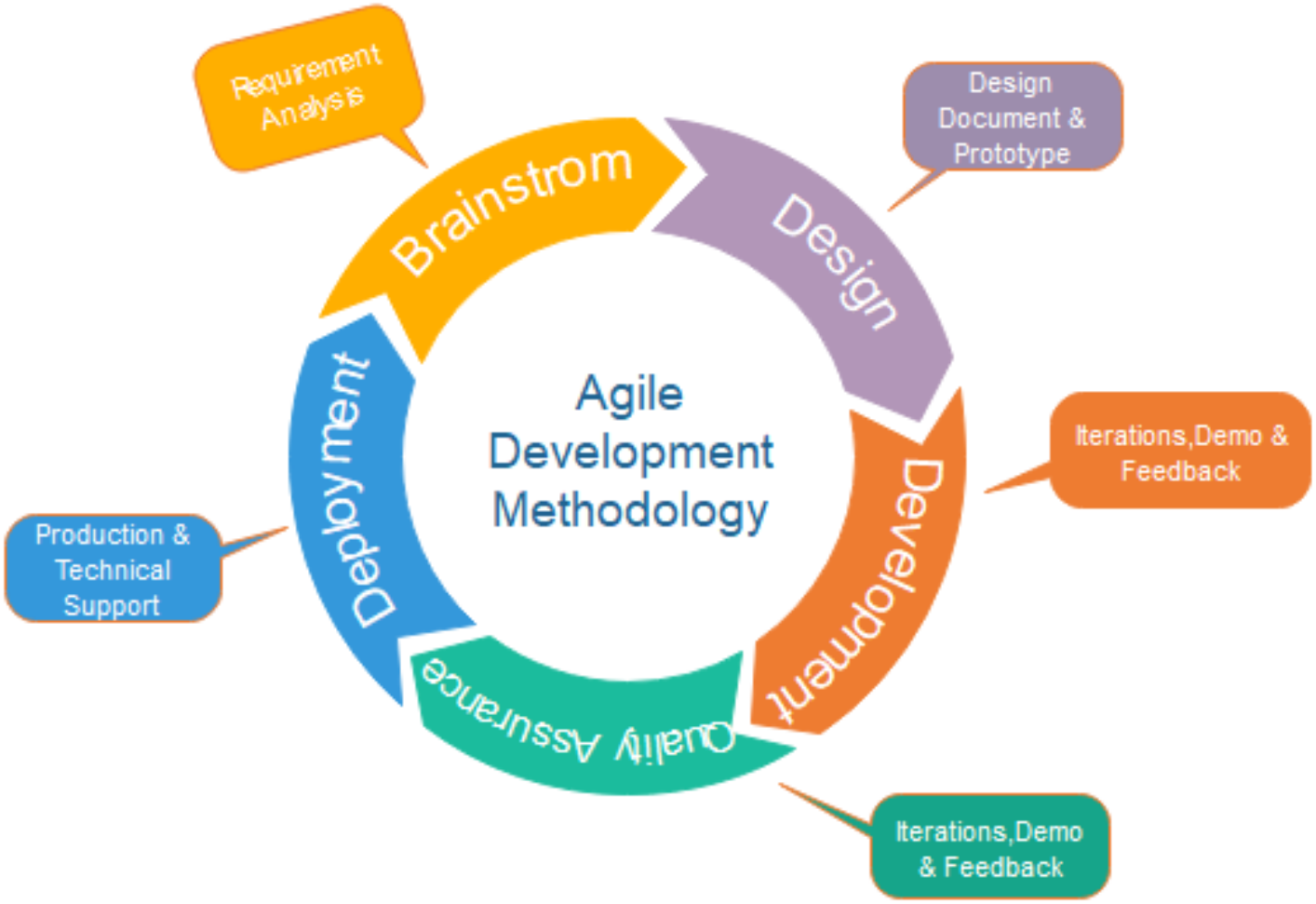


Fig. Agile Model

AGILR MODEL

- In the Agile model, both development and testing activities are concurrent, unlike the Waterfall model.

- The **Agile software development** methodology is one of the simplest and effective processes to turn a vision for a business need into software solutions. Agile is a term used to describe software development approaches that employ continual planning, learning, improvement, team collaboration, evolutionary development, and early delivery. It encourages flexible responses to change.

AGILR MODEL

The agile software development emphasizes on four core values.

- Individual and team interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

AGILR MODEL

Agile development takes an incremental approach to design. Similarly, **Agile testing** includes an incremental approach to testing. In this type of testing , features are tested as they are developed.

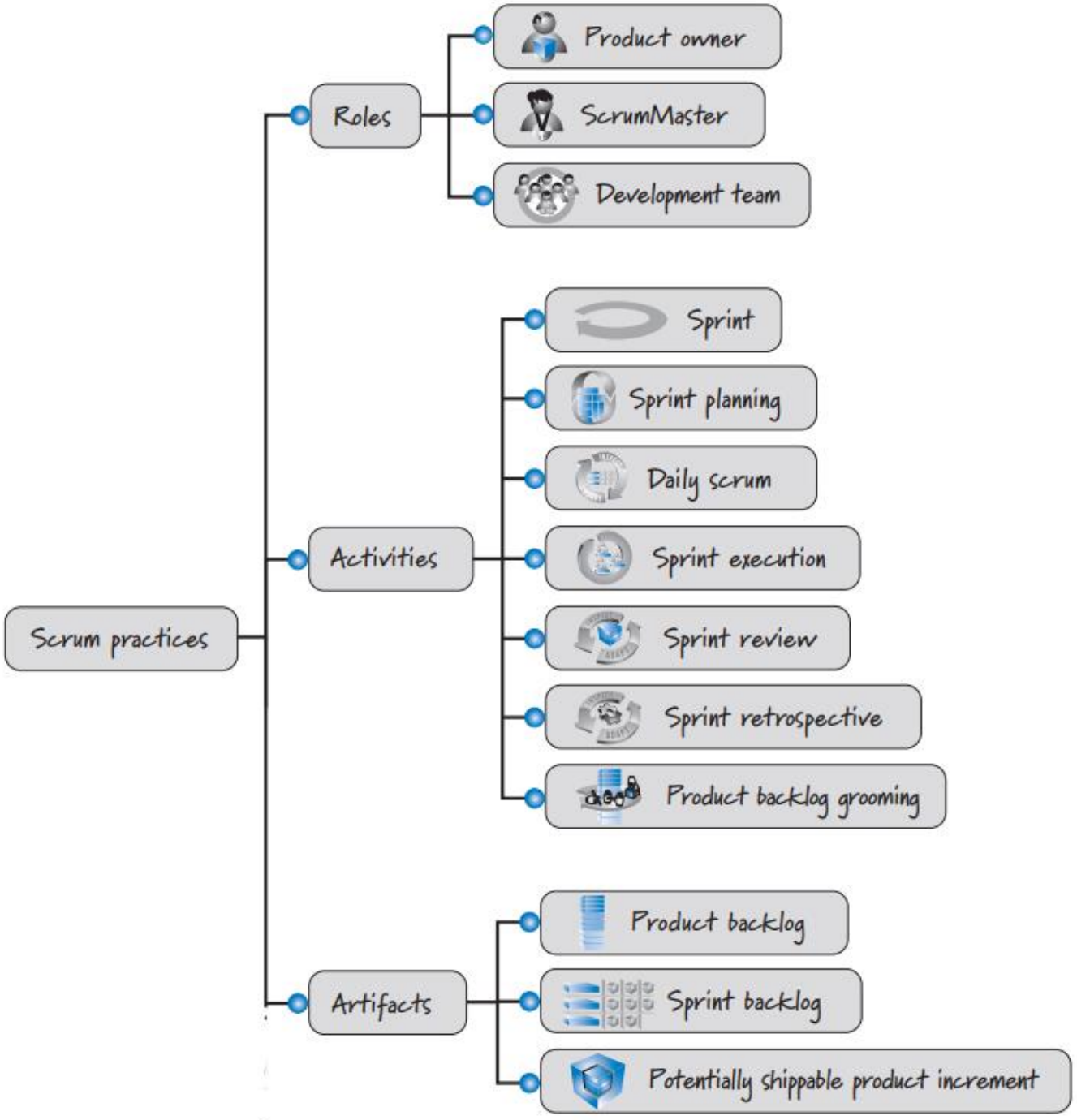
Agile Testing Methods:

- Scrum
- Crystal
- Dynamic Software Development Method (DSDM)
- Feature Driven Development(FDD)
- Lean Software Development
- eXtreme Programming(XP)

SCRUM

Scrum is not a standardized process where you methodically follow a series of sequential steps that are guaranteed to produce, on time and on budget, a high-quality product that delights customers. Instead, Scrum is a framework for organizing and managing work.

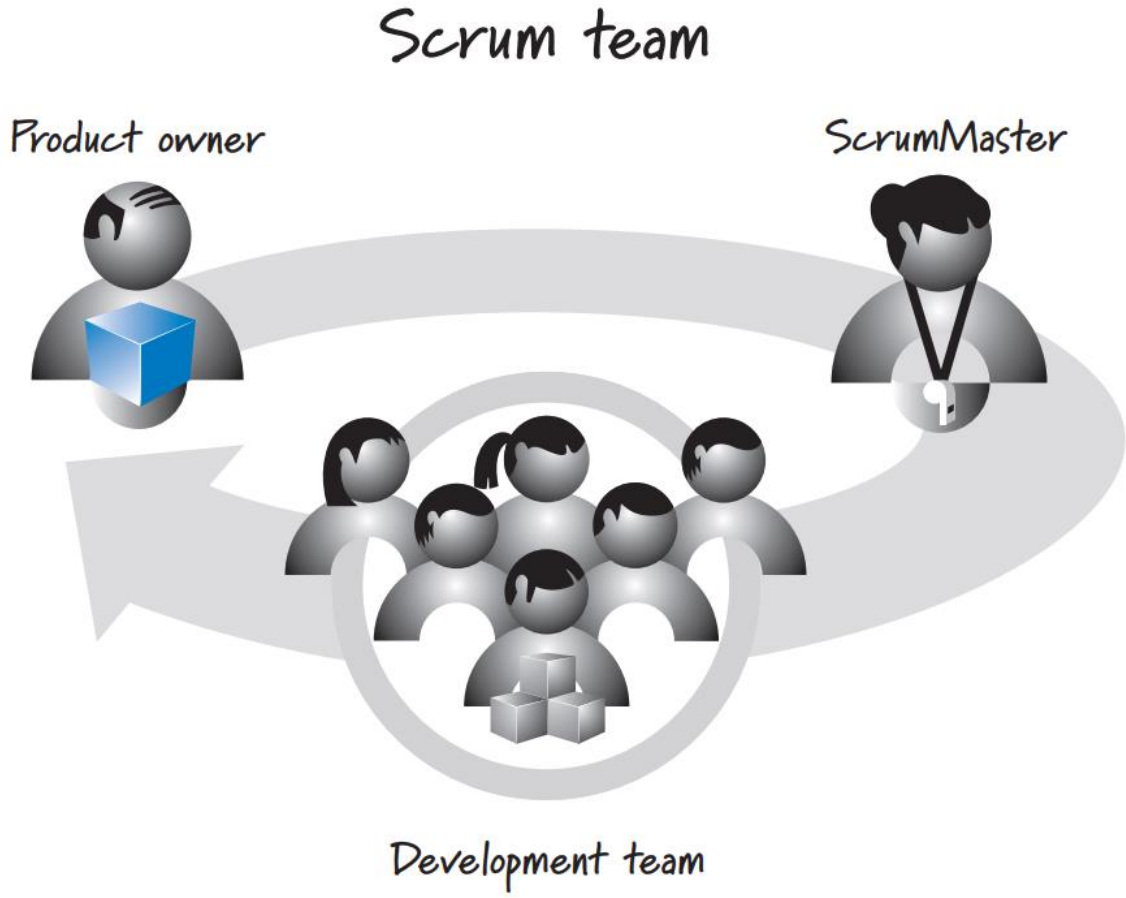
SCRUM


Scrum practices
Roles
Product owner
ScrumMaster
Development team
Activities
Sprint
Sprint planning
Daily scrum
Sprint execution
Sprint review
Sprint retrospective
Product backlog grooming
Artifacts
Product backlog
Sprint backlog
Potentially shippable product increment

SCRUM ROLE

Scrum development efforts consist of one or more Scrum teams, each made up of three Scrum roles:

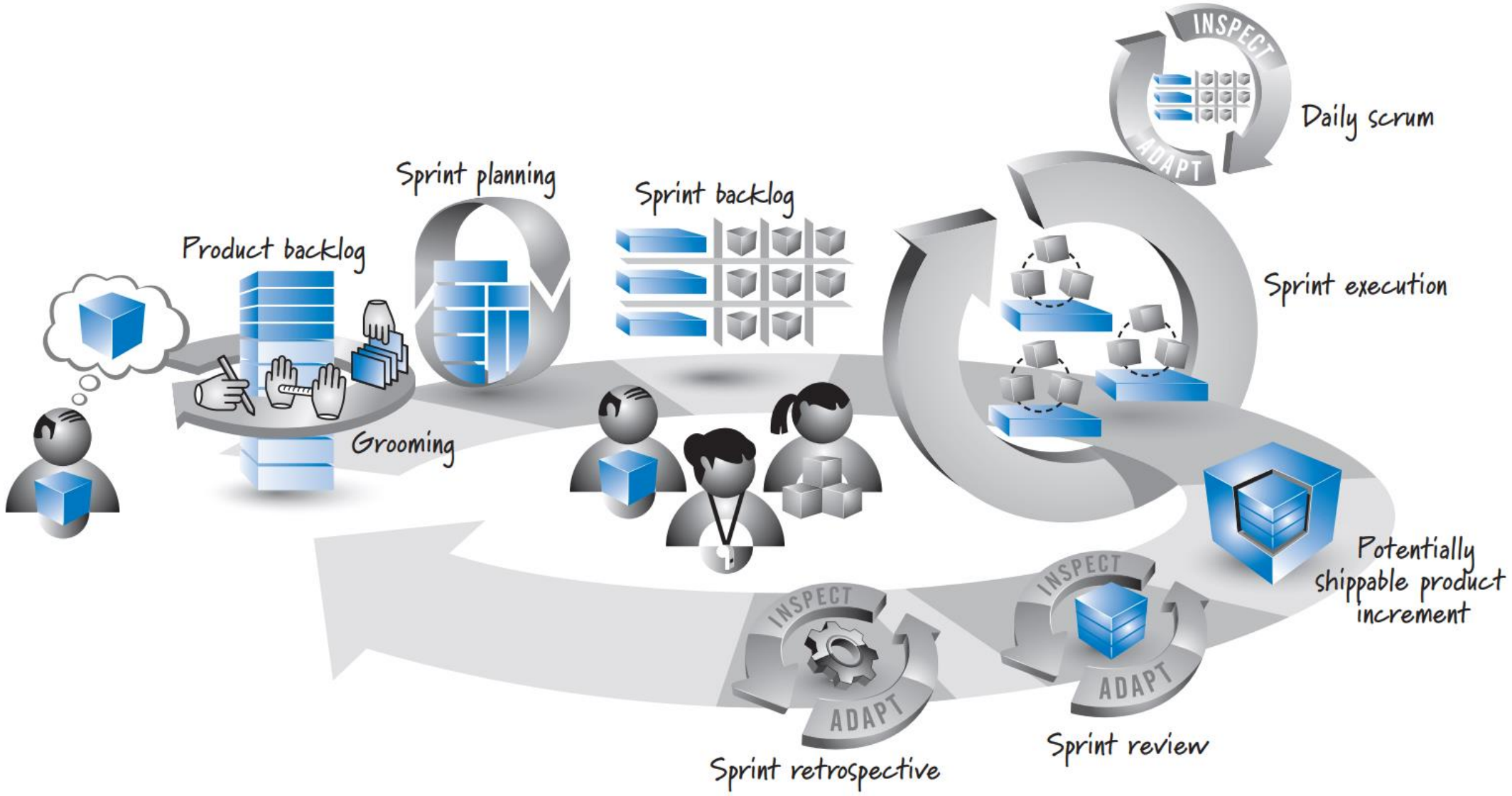
- Product owner,
- ScrumMaster and
- The development team



SCRUM ROLE

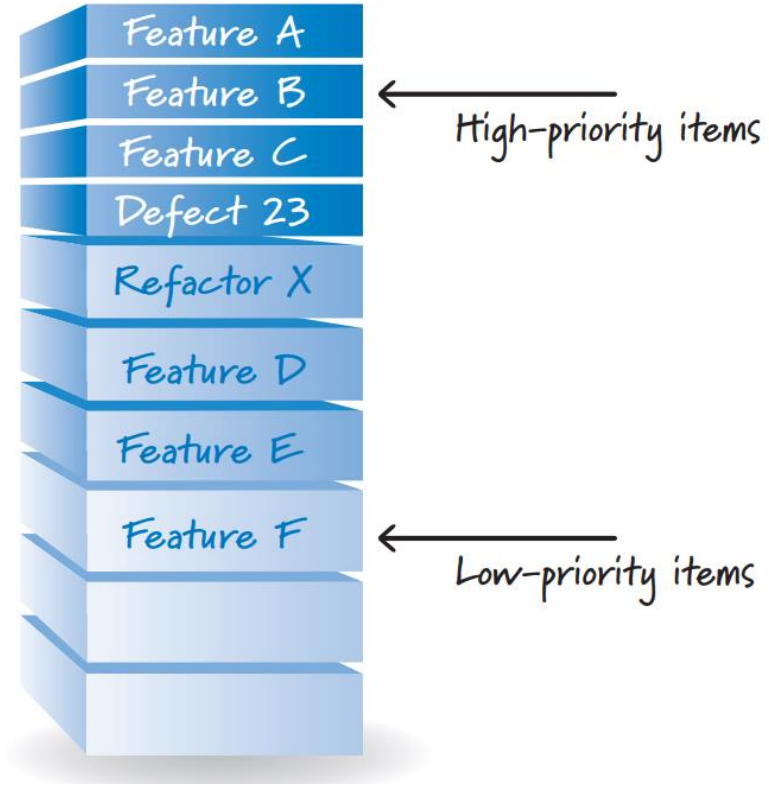
- The **product owner** is responsible for what will be developed and in what order.
- The **ScrumMaster** is responsible for guiding the team in creating and following its own process based on the broader Scrum framework.
- **The development team** is responsible for determining how to deliver what the product owner has asked for.

SCRUM Activities & Artifacts



SCRUM Activities & Artifacts

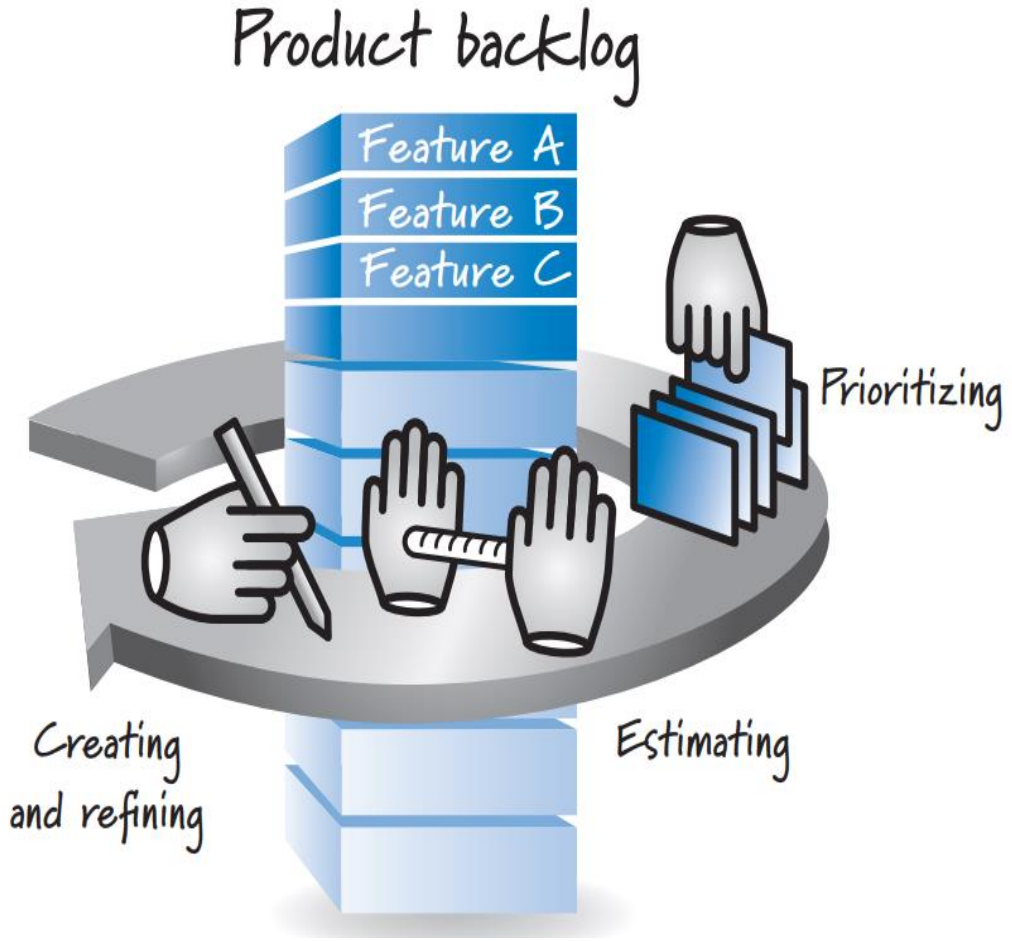
The **product owner** has a vision of what he wants to create (the big cube). Because the cube can be large, through an activity called **grooming** it is broken down into a set of features that are collected into a prioritized list called the **product backlog**.



Product backlog

SCRUM Activities & Artifacts

The activity of creating and refining product backlog items, estimating them, and prioritizing them is known as **grooming**.

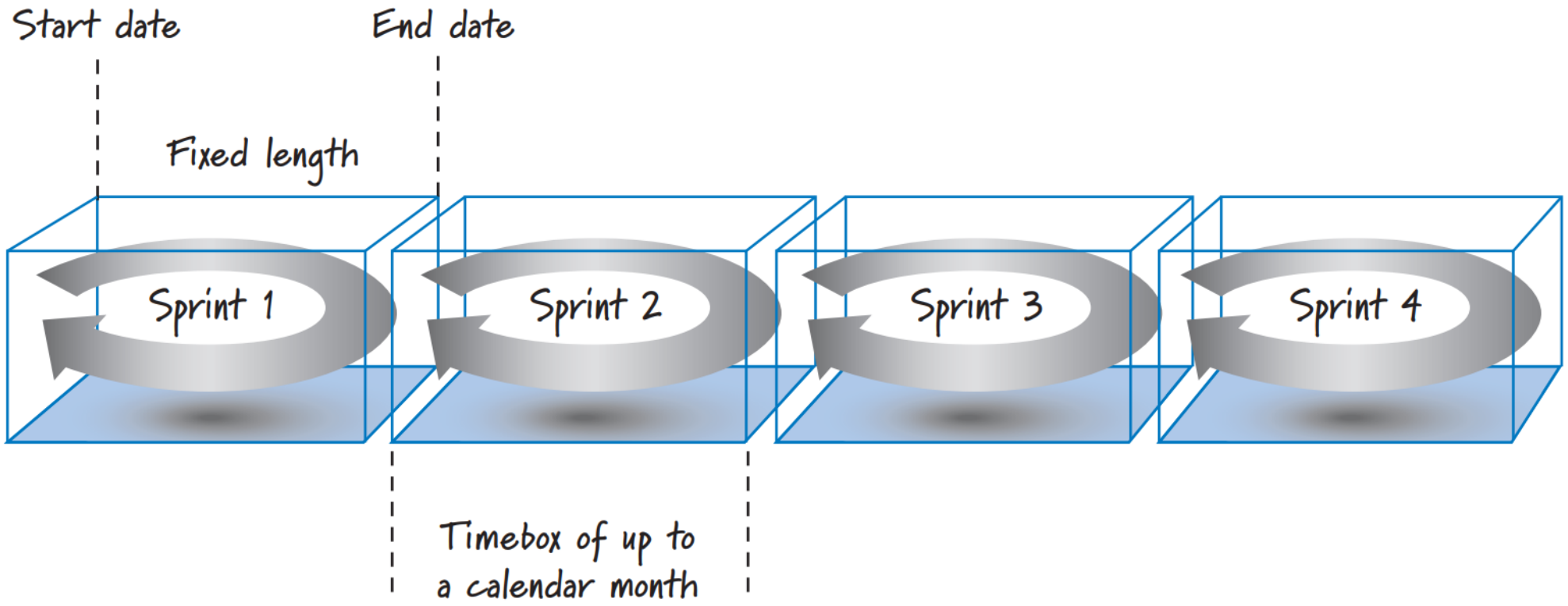


Product backlog grooming

SCRUM Activities & Artifacts

- In Scrum, work is performed in iterations or cycles of up to a calendar month called **sprints.**
- The work completed in each sprint should create something of tangible value to the customer or user.
- Sprints are timeboxed so they always have a fixed start and end date, and generally they should all be of the same duration.
- A new sprint immediately follows the completion of the previous sprint.
- As a rule we do not permit any goal-altering changes in scope or personnel during a sprint; however, business needs sometimes make adherence to this rule impossible.

SCRUM Activities & Artifacts

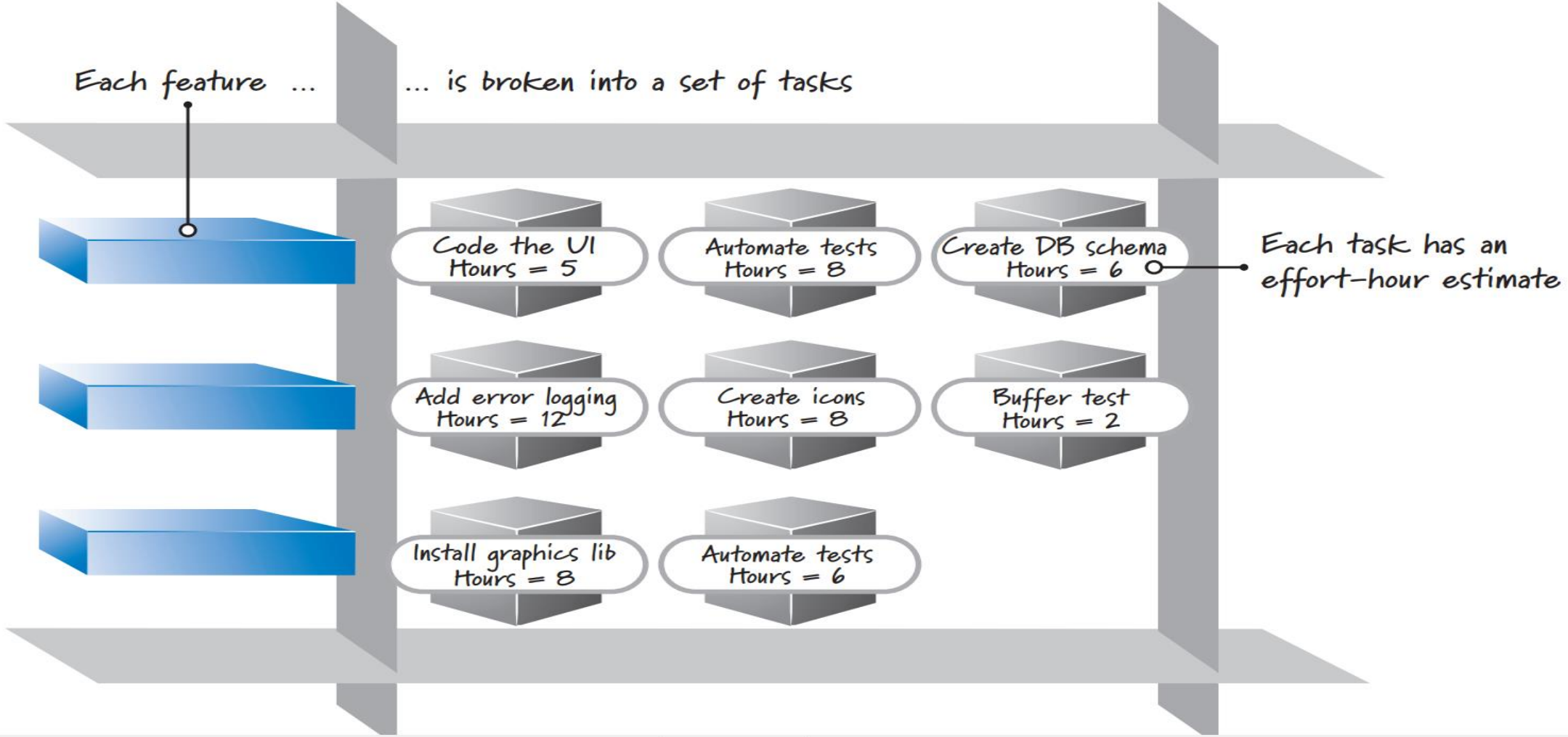


Sprint characteristics

SCRUM Activities & Artifacts

A product backlog may represent many weeks or months of work, which is much more than can be completed in a single, short sprint. To determine the most important subset of product backlog items to build in the next sprint, the product owner, development team, and ScrumMaster perform **sprint planning**. During **sprint planning**, the product owner and development team agree on a **sprint goal** that defines what the upcoming sprint is supposed to achieve.

SCRUM Activities & Artifacts



Sprint backlog

SCRUM Activities & Artifacts

Sprint Execution

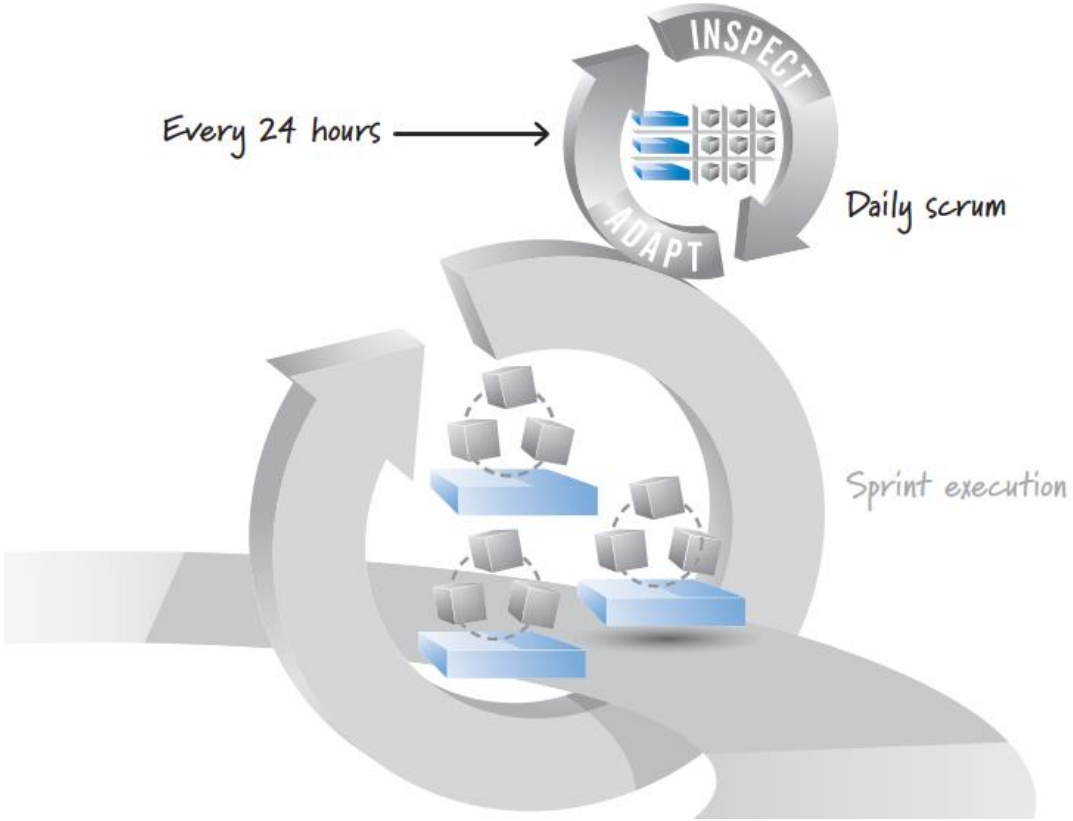
Once the Scrum team finishes sprint planning and **agrees on the content** of the next sprint, the development team, guided by the ScrumMaster’s coaching, **performs all of the task-level** work necessary to get the features **done**, where “done” means there is a high degree of confidence that all of the work necessary for producing good-quality features has been completed.

Sprint backlog

SCRUM Activities & Artifacts

Daily Scrum

Each day of the sprint, ideally at the **same time**, the development team members hold a *timeboxed* **(15 minutes or less)** daily scrum. This inspect-and adapt activity is sometimes referred to as the **daily stand-up** because of the common practice of everyone standing up during the meeting to help promote brevity.



SCRUM Activities & Artifacts

A common **approach** to performing the **daily scrum** has the ScrumMaster facilitating and each team member taking turns answering **three questions** for the benefit of the other team members:

1. What did I accomplish since the last daily scrum?
2. What do I plan to work on by the next daily scrum?
3. What are the obstacles or impediments that are preventing me from making progress?

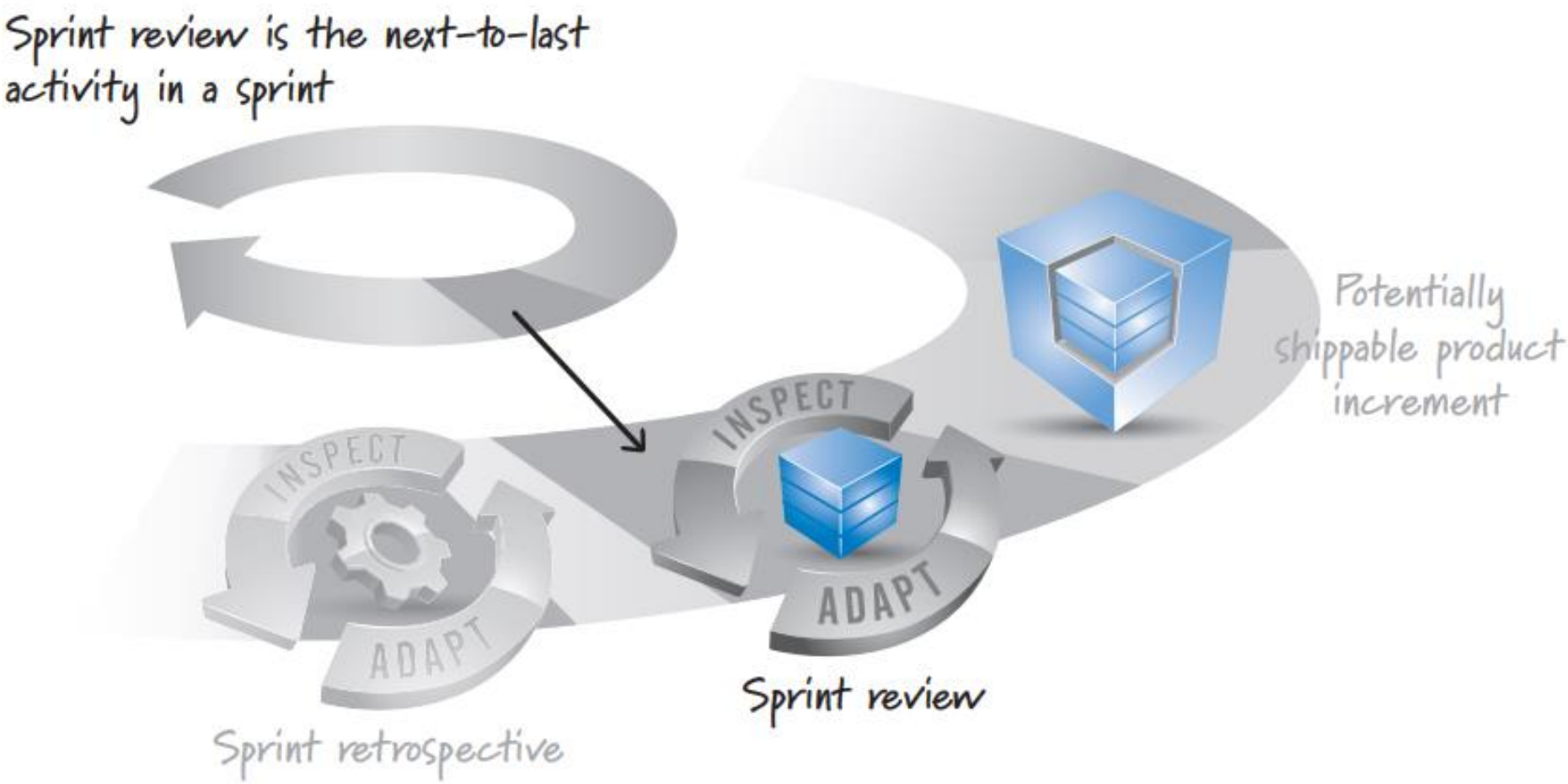
SCRUM Activities & Artifacts

Sprint Review

At the **end of the sprint** there are two additional inspect-and-adapt activities. One is called the **sprint review**. The goal of this activity is to **inspect and adapt** the product that is **being built**. Critical to this activity is the conversation that takes place among its participants, which include the **Scrum team, stakeholders, sponsors, customers, and interested members** of **other teams**. The conversation is focused on reviewing the **just-completed features** in the context of the overall development effort. Everyone in attendance gets clear visibility into what is occurring and has an opportunity to help guide the forthcoming development to ensure that the most business-appropriate solution is created.

SCRUM Activities & Artifacts

Sprint Review



SCRUM Activities & Artifacts

Sprint Retrospective

The second inspect-and-adapt activity at the end of the sprint is the sprint retrospective This activity frequently occurs after the sprint review and before the next sprint planning. Whereas the **sprint review is a time to inspect and adapt the product**, the **sprint retrospective** is an opportunity to inspect and adapt the **process**.

Thank you