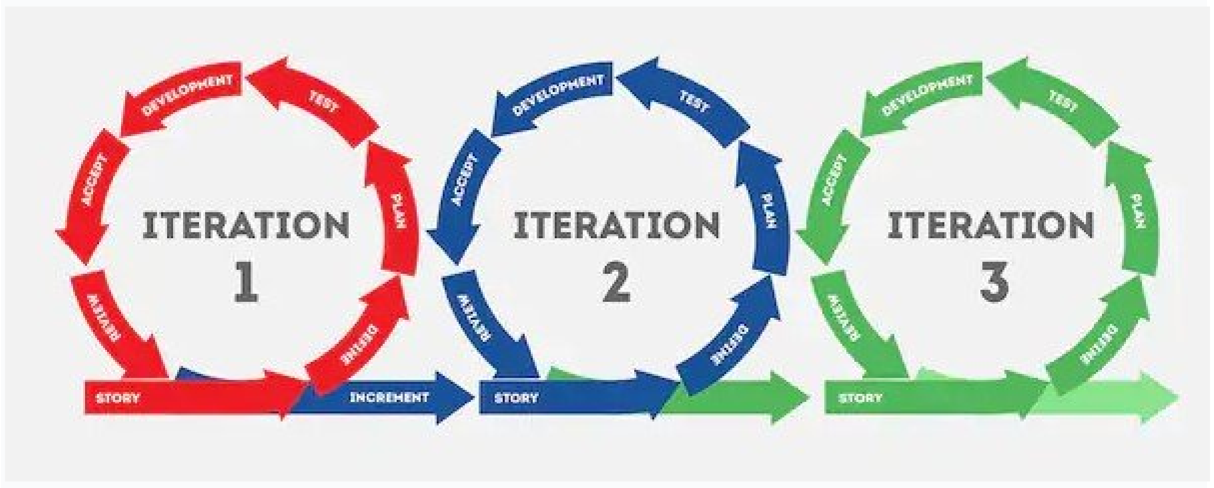
Iteratiivne mudel / Inkrementaalne mudel(Inglise keeles)

What is Iterative model

The iterative model is a software development life cycle (SDLC) approach in which initial development work is carried out based on well-stated basic requirements, and successive enhancements are added to this base piece of software through iterations until the final system is built. We get a working piece of software very early in the lifecycle because the iterative model begins with a simple execution of a small collection of software requirements, which iteratively improves the evolving variants until the entire system is executed and ready to be redistributed. Every Iterative model release is created over a certain and predetermined time period known as iteration. Bugs and errors from the previous iteration do not propagate to the next iteration, and this model is flexible enough to incorporate customer feedback in every iteration.



Phases of Iterative Model

1. Requirement Gathering & Analysis

The business requirements are gathered during this phase of the iterative model. Then, an analyst determines whether they can be met within the financial constraints. This phase details the business needs, and system information (hardware or software) is acquired and assessed for viability.

2. Design

During this phase of the iterative model, the project team receives the complete list of criteria for starting work in a specific direction. Then, they use various diagrams, like a data flow diagram, class diagram, activity diagram, state transition diagram, and so on, to gain explicit knowledge of the program design and to help them progress with development. Based on their investigation, developers provide viable solutions. Furthermore, the project's scale and criticality are crucial factors in deciding the complexity of the design for the project.

3. Implementation

At this point in the project, according to the iterative model, the actual coding of the system begins. This stage will be influenced by the Design Stage's analysis and design. All needs, planning, and design plans have been carried out. The chosen design will be implemented by the developer using predefined coding and metrics standards. They must implement a unit test at each stage of code development and should strive to produce a fully functional, testable system for that iteration. The complexity of work and time spent on this iteration will vary depending on the project.

4. Testing

This stage entails comparing the current build iteration to a set of rules and norms to determine whether or not it fits them. This sort of testing includes performance testing, stress testing, security testing, requirements testing, usability testing, multi-site testing, disaster recovery testing, and so on. The tester can create new test cases or reuse those from previous releases, but testing is a key priority because any failures would affect the software's specification, affecting the business. We can also check in with the project stakeholders to perform some tests and get their input. A developer or tester must guarantee that correcting one bug does not result in the appearance of new bugs in the system.

5. Deployment

After completing all the phases, the software is deployed to its work environment.

6. Review

In this phase, after the product deployment, we check the behavior and validity of the deployed product. And if any errors are found, the process starts again from requirement gathering.

7. Maintenance

In the maintenance phase, after software deployment in the working environment, there may be some bug fixes or new updates required.

Advantages of the Iterative Model

- A working product is produced much early in the lifecycle, unlike the waterfall model, where a working product is available only at the end of the lifecycle.
- We can detect errors and bugs at an early stage and prevent them from flowing downwards. We test the output of every iteration and do not let bugs from the previous iteration propagate to the next iteration.
- Changing the requirements does not incur much cost in this model, although it may not always be possible to accommodate new requirements due to system structure and design constraints.

- In this model, less time is spent on documenting and more time on designing and developing.

What is incremental model

The incremental build model is a method of software development where the product is designed, implemented, and tested incrementally (a little more is added each time) until the product is finished. It involves both development and maintenance. The product is defined as finished when it satisfies all of its requirements.

Advantages

1. After each iteration, regression testing should be conducted. During this testing, faulty elements of the software can be quickly identified because few changes are made within any single iteration.
2. It is generally easier to test and debug than other methods of software development because relatively smaller changes are made during each iteration. This allows for more targeted and rigorous testing of each element within the overall product.
3. Customers can respond to features and review the product for any needed or useful changes.
4. Initial product delivery is faster and costs less.

Disadvantages

1. The resulting cost may exceed the cost of the organization.
2. As additional functionality is added to the product, problems may arise related to system architecture which were not evident in earlier prototypes

Tasks involved

1. Communication: helps to understand the objective.
2. Planning: required as many people (software teams) to work on the same project but with different functions at the same time.
3. Modeling: involves business modeling, data modeling, and process modeling.
4. Construction: this involves the reuse of software components and automatic code.
5. Deployment: integration of all the increments.

Info:

1.Iterative

https://www.scaler.com/topics/software-engineering/iterative-model-in-software-engineering/

2.incremental

https://en.wikipedia.org/wiki/Incremental_build_model