Software Prototyping Model and Phases

Software Prototyping Model is a software development methodology that involves creating an initial prototype of the software product before developing the final product. The prototype is a working model of the software product that can be used to gather feedback and refine the design before the final product is developed. The prototyping model is particularly useful for projects where the requirements are not well-defined or are likely to change over time. The prototyping model can help to reduce the risk of developing a software product that does not meet the needs of end-users or stakeholders, while also reducing development time and cost. The prototype can be developed using different methods, such as throwaway prototyping, evolutionary prototyping, and incremental prototyping. The prototyping model can be used in conjunction with other software development methodologies, such as agile development, to create software products that meet the needs of end-users and stakeholders.

Prototyping is a software development methodology that focuses on the use of working models that are constantly refined based on feedback from the end user. Prototyping is most commonly used to develop systems with significant end-user interaction and complex user interfaces.

Because it allows for the capture of customer requirements at an early stage of development, software prototyping is becoming increasingly popular as a software development model. It allows for valuable customer feedback and assists software designers and developers in understanding what is expected directly from sales.

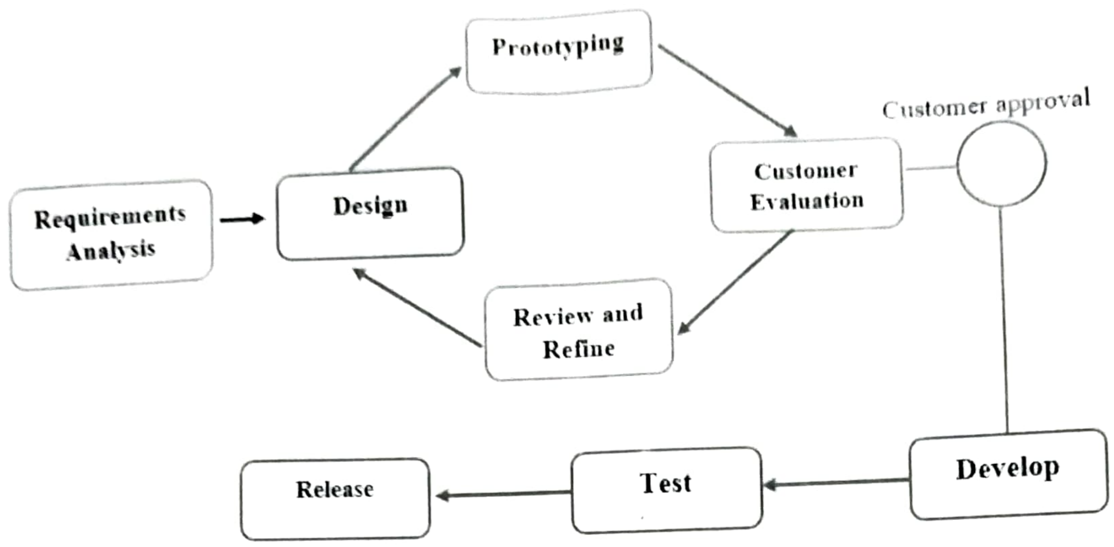
The term "software prototyping" refers to the process of creating software application prototypes that demonstrate the functionality of the product under development but may not contain the exact logic of the original software.

What is Software Prototyping?

- A prototype can be a limited-functionality software performance model.
- The prototype does not always contain the exact logic used in the specific software application and is an additional effort to be considered when estimating effort.
- Prototyping is used to allow users to evaluate and test developer proposals before they are implemented.
- It also aids in understanding the user-specific requirements that will not be considered by the developer during product design.

Prototyping Model Phases:

Prototyping Model has six SDLC phases as follows



Step-1: Requirements gathering and analysis :
Requirement analysis is the first step in developing a prototyping model. During this phase, the system's desires are precisely defined. During the method, system users are interviewed to determine what they expect from the system.

Step-2: Quick design :
The second phase could consist of a preliminary design or a quick design. During this stage, the system's basic design is formed. However, it is not a complete design. It provides the user with a quick overview of the system. The rapid design aids in the development of the prototype.

Step-3: Build a Prototype :
During this stage, an actual prototype is intended to support the knowledge gained from quick design. It is a small low-level working model of the desired system.

Step-4: Initial user evaluation :
The proposed system is presented to the client for preliminary testing at this stage. It is beneficial to investigate the performance model's strengths and weaknesses. Customer feedback and suggestions are gathered and forwarded to the developer.

Step-5: Refining prototype :
If the user is dissatisfied with the current model, you may want to improve the type that responds to user feedback and suggestions. When the user is satisfied with the upgraded model, a final system based on the approved final type is created.

Step-6: Implement Product and Maintain :
The final system was fully tested and distributed to production after it was developed to support the original version. To reduce downtime and prevent major failures, the programmer is run on a regular basis.

Types of Prototyping Model:
Four types of Prototyping models are :

1. Rapid Throwaway prototypes
2. Evolutionary prototype
3. Incremental prototype
4. Extreme prototype

1. Rapid Throwaway Prototype :

The preliminary requirement is used for quick throwaway. It is quickly developed to show how the need will appear visually. Customer feedback aids in the movement of demand change, and the prototype is re-created until the need is basic. As a result, the upgraded version is rejected and cannot be the last accepted type. This programme aids in the testing of ideas and the rapid response to customer needs.

2. Evolutionary Prototyping :

It assists you in avoiding both time and effort waste. This is due to the fact that performing a specific type from the start of each method of communication can be time-consuming. This model is useful for a project that involves the use of an unknown replacement technology. It is also used in high-end projects where each feature must be tested only once. It's useful when the need isn't stable or isn't clearly understood at the start.

3. Incremental Prototyping :

The final product is decimated into small prototypes and developed individually in incremental prototyping. The various prototypes are eventually combined into a single product. This method is useful for reducing feedback time between the user and, consequently, the application development team.

4. Extreme Prototyping :

The extreme prototyping method is commonly used in web development. It is divided into three stages that must be completed in order.

1. The HTML format contains a basic prototype of the current page.
2. A prototype services layer will allow you to simulate data processes.
3. The services are built and integrated into the final prototype.

Software Prototyping Application :

Software prototyping is extremely beneficial in the development of systems that require a high level of user interaction, such as online applications. Systems that require users to fill out forms or view various screens before data is processed can benefit greatly from prototyping in order to convey the exact look and feel even before the software is developed. Prototyping is usually not an option for software that involves a large amount of information processing and the majority of the functionality is internal with little programme. Prototype development can be an additional overhead in such projects and should necessitate a significant amount of additional effort.

Advantages of Software Prototyping :

- Users help to shape the future. As a result, errors can be discovered during the first stage of the software development process.
- Prototyping is also considered a risk reduction function because it allows non-existent performance to be seen, lowering the risk of failure.
- Assists team members in effectively communicating.
- Customer satisfaction exists, and he can feel the product from the start.

- There will be no risk of software loss.
- Quick user feedback aids in the development of better software solutions.

Disadvantages of Software Prototyping :

- Prototyping is a time-consuming and labor-intensive process.
- The cost of creating a specific type of waste is completely wasted because the prototype is eventually discarded.
- Prototyping may result in an overabundance of change requests.
- Customers may be unwilling to commit to the iteration cycle for an extended period of time.
- During each customer test, there may be too many variations in software requirements.
- Poor documentation as a result of changing customer needs.

Uses of Software Prototyping Model :

Some of the uses of the software prototyping model include:

1. Requirements gathering: The prototyping model is useful for gathering and refining requirements for a software product. By creating a prototype, developers can get feedback from end-users and stakeholders, which helps to ensure that the software product meets their needs and requirements.
2. User testing: The prototyping model is useful for testing the usability and functionality of the software product. By creating a prototype, developers can get feedback from end-users, which helps to identify any issues or concerns before the final product is developed.
3. Risk reduction: The prototyping model can help to reduce the risk of developing a software product that does not meet the needs of end-users or stakeholders. By creating a prototype, developers can identify issues and refine the design before the final product is developed, which reduces the risk of costly rework and delays.
4. Faster development: The prototyping model can help to speed up the development process. By creating a prototype, developers can get feedback and refine the design quickly, which helps to reduce the time and effort required to develop the final product.
5. Cost reduction: The prototyping model can help to reduce the overall cost of software development. By identifying issues and refining the design early on in the development process, developers can reduce the risk of costly rework and delays, which can ultimately save time and reduce costs.

Issues of Software Prototyping Model :

1. Incomplete requirements: The prototyping model is designed to help refine requirements through feedback from end-users. However, if the initial requirements are incomplete or inaccurate, the prototype may not fully meet the needs of the end-users or stakeholders.
2. Scope creep: Because the prototyping model encourages user feedback and refinement, it can be susceptible to scope creep. This can occur when new features or requirements are added to the project, which can lead to delays and increased costs.

3. Increased development costs: While the prototyping model can help to reduce the overall cost of software development, it can also increase costs in the short term. This is because the creation of prototypes requires time and effort, which can add to the development costs.
4. Technical debt: The prototyping model can also lead to technical debt if the prototype is not properly designed or documented. This can create issues down the line when the final product is developed, which can lead to delays and increased costs.
5. Lack of scalability: The prototyping model may not be suitable for large or complex software projects. This is because the focus on rapid prototyping and user feedback may not be sufficient to address the complexities of a larger project.