


Unit-wise questions - Software Engineering

[Syllabus](#)[Notes](#)[Old Questions & solutions](#)[Yearly Questions](#)[Text & reference books](#)[Software Engineering](#)[Compiler Design and Construction](#)[E-Governance](#)[NET Centric Computing](#)[Technical Writing](#)[Applied Logic](#)[E-commerce](#)[Automation and Robotics](#)[Neural Networks](#)[Computer Hardware Design](#)[Cognitive Science](#)[Real Time System \(old-course\)](#)[Unit: 1 Introduction](#)[24 questions](#)[Unit: 2 Software Processes](#)[25 questions](#)

2. Explain the waterfall model with its merits and demerits.

 **hide solution**
asked in 2068(II)

Solution

The **waterfall model** is a sequential approach, where each fundamental activity of a process represented as a separate phase, arranged in linear order. Each phase is carried out completely (for all requirements) before proceeding to the next. The process is strictly sequential - no backing up or repeating phases. This SDLC model includes gradual execution of every stage completely. This process is strictly documented and predefined with features expected to every phase of this SDLC model.

 SDLC – Lawrence's A-Level Computer Science

Fig: The waterfall model

The sequential phases in Waterfall model are –

- 1. Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- 2. System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- 3. Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- 4. Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- 5. Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- 6. Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.


Merits of Waterfall model

- This model is simple to implement also the number of resources that are required for it is minimal.
- The requirements are simple and explicitly declared; they remain unchanged during the entire project development.
- The start and end points for each phase is fixed, which makes it easy to cover progress.
- It gives easy to control and clarity for the customer due to a strict reporting system.
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.

Demerits of Waterfall model

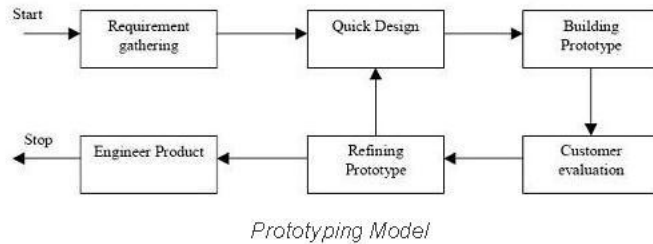
- No working software is produced until late during the life cycle.
- In this model, the risk factor is higher, so this model is not suitable for more significant and complex projects.
- This model cannot accept the changes in requirements during development.
- It becomes tough to go back to the phase.
- Since the testing done at a later stage, it does not allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare.

2. Explain the prototyping model of software development.

 **hide solution**
asked in 2069

Solution

Prototyping is a *rapid, iterative, and incremental* process of systems development in which requirements are converted to a working system that is continually revised through close work between the development team and the users.



1. Requirements gathering and analysis: A prototyping model starts with requirement analysis. In this phase, the requirements of the system are defined in detail. During the process, the users of the system are interviewed to know what is their expectations from the system.

2. Quick design: In this stage, a simple design of the system is created. However, it is not a complete design. It gives a brief idea of the system to the user. The quick design helps in developing the prototype.

3. Build a Prototype: In this phase, an actual prototype is designed based on the information gathered from a quick design. It is a small working model of the required system.

4. Initial user evaluation: In this stage, the proposed system is presented to the client for an initial evaluation. It helps to find out the strength and weaknesses of the working model. Comment and suggestions are collected from the customer and provided to the developer.

5. Refining prototype: If the user is not happy with the current prototype, we need to refine the prototype according to the user's feedback and suggestions. This phase will not over until all the requirements specified by the user are met. Once the user is satisfied with the developed prototype, a final system is developed based on the approved final prototype.

6. Implement Product and Maintain: Once the final system is developed based on the final prototype, it is thoroughly tested and deployed to production. The system undergoes routine maintenance for minimizing downtime and prevent large-scale failures.

2. What is software process model? Discuss reuse-oriented development in detail.

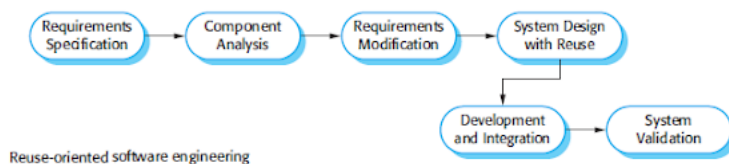
 **hide solution**
asked in 2071

Solution

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

The **reuse-oriented development** is where they reuse programming or software in previous projects or existing software components. Reuse-oriented development is based on systematic reuse where systems are integrated from existing components. This development can reduce the overall cost of software development and it can also save time because each phase of the process builds on the previous phase which has already been refined.

The general process model for reuse based development is shown in figure below:



1. Requirement Specification: All possible requirements of the system to be developed are captured in this phase.


2. Component Analysis: According to given requirement, component is selected to implement that requirement specification. That is not possible the selected component provide the complete functionality, but that is possible the component used provide some of the functionality required.

3. Requirement Modification: Information about component that is selected during component analysis is used to analysis requirement specification. Requirements are modified according to available components. Requirement modification is critical then component analysis activity is reused to find relative solution.

4. System design with reuse: During this stage the design of the system is build. Designer must consider the reused component and organize the framework. If reused component is not available then new software is develop.

5. Development and Integration: Components are integrated to develop new software. Integration in this model is part of development rather than separate activity.

2. Explain the software process model with example.

 **hide solution**

Solution


A software process model is an abstract representation of a process that presents a description of a process from some particular perspective. Process models may contain activities, which are part of the software process, software product, and the roles of people involved in software engineering. Some examples of the types of software process models that may be produced are:

1. **A workflow model:** This shows the series of activities in the process along with their inputs, outputs and dependencies. The activities in this model perform human actions.
2. **A dataflow or activity model:** This represents the process as a set of activities, each of which carries out some data transformations. It shows how the input to the process, such as a specification is converted to an output such as a design. The activities here may be at a lower level than activities in a workflow model. They may perform transformations carried out by people or by computers.
3. **A role/action model:** This means the roles of the people involved in the software process and the activities for which they are responsible.

There are several various general models or paradigms of software development:

1. **The waterfall approach:** This takes the above activities and produces them as separate process phases such as requirements specification, software design, implementation, testing, and so on. After each stage is defined, it is "signed off" and development goes onto the following stage.
2. **Evolutionary development:** This method interleaves the activities of specification, development, and validation. An initial system is rapidly developed from a very abstract specification.
3. **Formal transformation:** This method is based on producing a formal mathematical system specification and transforming this specification, using mathematical methods to a program. These transformations are 'correctness preserving.' This means that you can be sure that the developed programs meet its specification.
4. **System assembly from reusable components:** This method assumes the parts of the system already exist. The system development process target on integrating these parts rather than developing them from scratch.

2. Why do we need software process model? Discuss reuse-oriented development in detail.

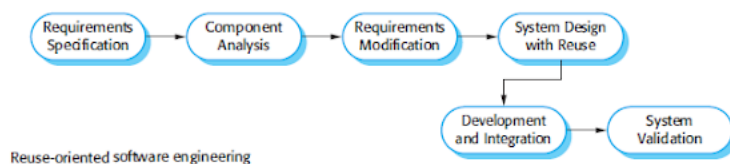
 **hide solution**
asked in 2072

Solution

A **software process model** is an abstract representation of a process. It presents a description of a process from some particular perspective. Software process model is needed because it represents the order in which the activities of software development will be undertaken. It describes the sequence in which the phases of the software lifecycle will be performed.

The **reuse-oriented development** is where they reuse programming or software in previous projects or existing software components. Reuse-oriented development is based on systematic reuse where systems are integrated from existing components. This development can reduce the overall cost of software development and it can also save time because each phase of the process builds on the previous phase which has already been refined.

The general process model for reuse based development is shown in figure below:



1. **Requirement Specification:** All possible requirements of the system to be developed are captured in this phase.
2. **Component Analysis:** According to given requirement, component is selected to implement that requirement specification. That is not possible the selected component provide the complete functionality, but that is possible the component used provide some of the functionality required.
3. **Requirement Modification:** Information about component that is selected during component analysis is used to analysis requirement specification. Requirements are modified according to available components. Requirement modification is critical then component analysis activity is reused to find relative solution.
4. **System design with reuse:** During this stage the design of the system is build. Designer must consider the reused component and organize the framework. If reused component is not available then new software is develop.
5. **Development and Integration:** Components are integrated to develop new software. Integration in this model is part of development rather than separate activity.

Solution

The **waterfall model** is a sequential approach, where each fundamental activity of a process represented as a separate phase, arranged in linear order. Each phase is carried out completely (for all requirements) before proceeding to the next. The process is strictly sequential - no backing up or repeating phases. This SDLC model includes gradual execution of every stage completely. This process is strictly documented and predefined with features expected to every phase of this SDLC model.

 SDLC – Lawrence's A-Level Computer Science

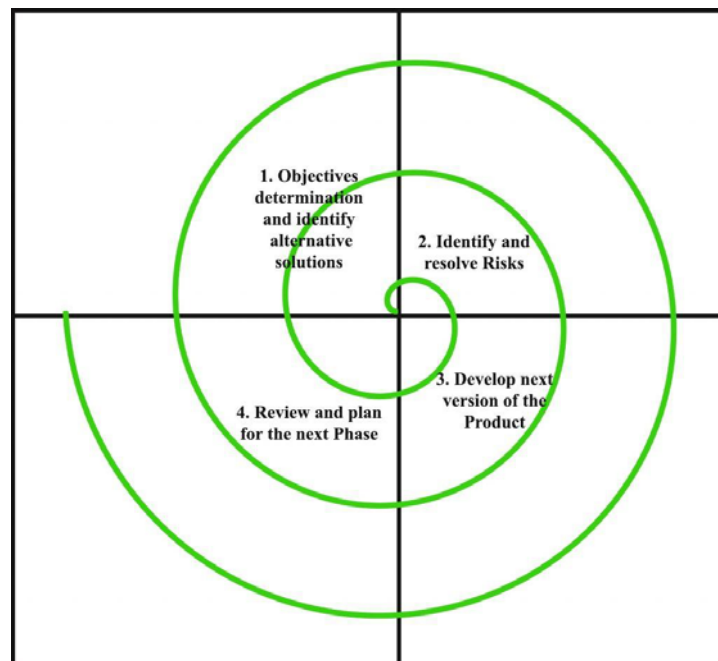
Fig: The waterfall model

The sequential phases in Waterfall model are –

- 1. Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- 2. System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- 3. Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- 4. Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- 5. Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- 6. Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment

Spiral Model is a risk-driven software development process model. It is a combination of waterfall model and iterative model. Spiral Model helps to adopt software development elements of multiple process models for the software project based on unique risk patterns ensuring efficient development process.


In spiral model, total software development process activities are divided into four groups as shown in figure below:



- 1. Objectives determination and identify alternative solutions:** Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase. Then alternative solutions possible for the phase are proposed in this quadrant.
- 2. Identify and resolve Risks:** During the second quadrant, all the possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.
- 3. Develop next version of the Product:** During the third quadrant, the identified features are developed and verified through testing. At the end of the third quadrant, the next version of the software is available.

4. **Review and plan for the next Phase:** In the fourth quadrant, the Customers evaluate the so far developed version of the software. In the end, planning for the next phase is started.

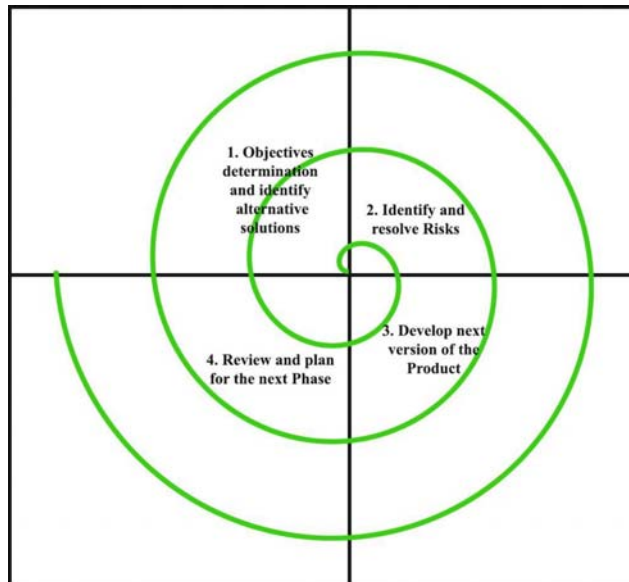
2. Describe spiral model with its advantages and disadvantages.

 **hide solution**
asked in 2074

Solution

Spiral Model is a risk-driven software development process model. It is a combination of waterfall model and iterative model. Spiral Model helps to adopt software development elements of multiple process models for the software project based on unique risk patterns ensuring efficient development process.

In spiral model, total software development process activities are divided into four groups as shown in figure below:



- 1. Objectives determination and identify alternative solutions:** Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase. Then alternative solutions possible for the phase are proposed in this quadrant.
- 2. Identify and resolve Risks:** During the second quadrant, all the possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.
- 3. Develop next version of the Product:** During the third quadrant, the identified features are developed and verified through testing. At the end of the third quadrant, the next version of the software is available.
- 4. Review and plan for the next Phase:** In the fourth quadrant, the Customers evaluate the so far developed version of the software. In the end, planning for the next phase is started.

Advantages:

- Changing requirements can be accommodated.
- Allows extensive use of prototypes.
- Requirements can be captured more accurately.
- Users see the system early.
- Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.

Disadvantages:

- Management is more complex.
- End of the project may not be known early.
- Not suitable for small or low risk projects and could be expensive for small projects.
- Process is complex
- Spiral may go on indefinitely.

2. What is software process model? Discuss waterfall model with its merits and demerits.

 **hide solution**
asked in 2076

Solution

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

The **waterfall model** is a sequential approach, where each fundamental activity of a process represented as a separate phase, arranged in linear order. Each phase is carried out completely (for all requirements) before proceeding to the next. The process is strictly sequential - no backing up or repeating phases. This SDLC model includes gradual execution of every stage completely. This process is strictly documented and predefined with features expected to every phase of this SDLC model.

SDLC – Lawrence's A-Level Computer Science

Fig: The waterfall model

The sequential phases in Waterfall model are –

- 1. Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- 2. System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- 3. Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- 4. Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- 5. Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- 6. Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.


Merits of Waterfall model

- This model is simple to implement also the number of resources that are required for it is minimal.
- The requirements are simple and explicitly declared; they remain unchanged during the entire project development.
- The start and end points for each phase is fixed, which makes it easy to cover progress.
- It gives easy to control and clarity for the customer due to a strict reporting system.
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.

Demerits of Waterfall model

- No working software is produced until late during the life cycle.
- In this model, the risk factor is higher, so this model is not suitable for more significant and complex projects.
- This model cannot accept the changes in requirements during development.
- It becomes tough to go back to the phase.
- Since the testing done at a later stage, it does not allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare.

1. Differentiate between software process and software process model.

 **hide solution**
asked in 2068

Solution

A **software process** is a set of related activities that leads to the production of the software. These activities may involve the development of the software from the scratch, or, modifying an existing system.

Any software process must include the following four activities:

- 1. Software specification** (or requirements engineering): Define the main functionalities of the software and the constraints around them.
- 2. Software design and implementation:** The software is to be designed and programmed.
- 3. Software verification and validation:** The software must conform to its specification and meets the customer needs.
- 4. Software evolution** (software maintenance): The software is being modified to meet customer and market requirements changes.

A **software process model** is the abstract representation of a software process. It is a structure of a software process present the description of a process. Each process model represents a process from a particular perspective.

Examples of process perspectives:

- **Workflow perspective** represents inputs, outputs and dependencies.
- **Data-flow perspective** represents data transformation activities.
- **Role/action perspective** represents the role/activities of the people involved in software process.

There are several process models are available


1. Waterfall model
2. Generic process model
3. Incremental model

4. Agile process model
5. Prototyping model
6. Spiral model
7. Iterative development model, etc.

1. Explain the different software life cycle models and compare them with advantages and disadvantages.

asked in Model
Question

2. What is waterfall model? Describe the activities of waterfall model and also mention its drawbacks.

 **hide solution**
asked in 2075

Solution

The **waterfall model** is a sequential approach, where each fundamental activity of a process represented as a separate phase, arranged in linear order. Each phase is carried out completely (for all requirements) before proceeding to the next. The process is strictly sequential - no backing up or repeating phases. This SDLC model includes gradual execution of every stage completely. This process is strictly documented and predefined with features expected to every phase of this SDLC model.

 SDLC – Lawrence's A-Level Computer Science

Fig: The waterfall model


The sequential phases in Waterfall model are –

- 1. Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- 2. System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- 3. Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- 4. Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- 5. Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- 6. Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

Drawbacks of Waterfall model

- No working software is produced until late during the life cycle.
- In this model, the risk factor is higher, so this model is not suitable for more significant and complex projects.
- This model cannot accept the changes in requirements during development.
- It becomes tough to go back to the phase.
- Since the testing done at a later stage, it does not allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare.

3. Explain the software specification, software validation and software evolution with example.

 **hide solution**
asked in 2071(II)

Solution

Software specification

A **software requirements specification (SRS)** is a detailed description of a software system to be developed with its functional and non-functional requirements. The SRS is developed based the agreement between customer and contractors. It may include the use cases of how user is going to interact with software system. The software requirement specification document consistent of all necessary requirements required for project development. To develop the software system we should have clear understanding of Software system. To achieve this we need to continuous communication with customers to gather all requirements.

Software validation

Software validation is the process of checking the software to ensure that it works exactly according to the requirement specification. During software validation process, the software is tested with different types of users on actual working data. If software works properly with specified environment and actual operating data, then it is said to that the software confirms user requirement specification. It involves checking process such as inspection and reviews at the stage of the software process from the user requirement definition to program development. The majority of validation costs are incurred after implementation, when the operational system is tested. System should not be tested as a single unit. It should be performed in different steps.

is tested. System should not be tested as a single unit. It should be performed in different stage.

Software evolution

Software Evolution refers to the process of developing software initially, then timely updating it for various reasons, i.e., to add new features or to remove obsolete functionalities etc. The evolution process includes fundamental activities of change analysis, release planning, system implementation and releasing a system to customers.

The cost and impact of these changes are accessed to see how much system is affected by the change and how much it might cost to implement the change. If the proposed changes are accepted, a new release of the software system is planned. During release planning, all the proposed changes (fault repair, adaptation, and new functionality) are considered. A design is then made on which changes to implement in the next version of the system. The process of change implementation is an iteration of the development process where the revisions to the system are designed, implemented and tested.

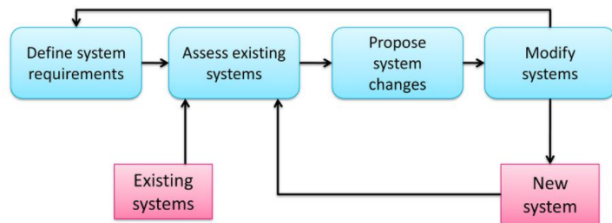


Fig: Software evolution

4. Why an evolutionary prototyping is used in software development? Explain.

hide solution
asked in 2069

Solution

In evolutionary prototyping, the prototype developed initially is incrementally refined on the basis of customer feedback till it finally gets accepted. It helps us to save time as well as effort. That's because developing a prototype from scratch for every interaction of the process can sometimes be very frustrating.

This model is helpful for a project which uses a new technology that is not well understood. It is also used for a complex project where every functionality must be checked once. It is helpful when the requirement is not stable or not understood clearly at the initial stage.

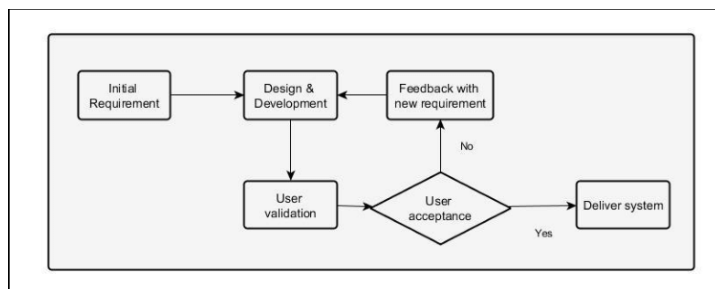


Fig: Evolutionary prototyping model

Evolutionary prototyping is used in software development due to the following reasons:

- The efforts required in developing the final system is reduced as the final system is implemented after all the specifications are clearly understood and there are fewer chances of final system being incorrect.
- It helps the customer to easily realize the required modification before final implementation of the system.
- The customer does not have to wait for a long to see the working model of the final system.
- There are more chances of the developed system is more user-friendly.
- It helps us to save time as well as effort.
- User satisfaction is achieved.

4. Explain the concept of incremental model with example.

hide solution
asked in 2073

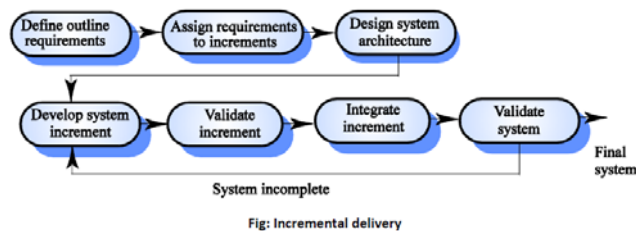
Solution

In an incremental model, customers identify, in outline, the services to be provided by the system. They identify which of the services are most important and which are least important to them. A number of delivery increments are then defined, with each increment providing a sub-set of the system functionality. The allocation of services to increments depends on the service priority with the highest priority services delivered first.

Once the system increments have been identified, the requirements for the services to be delivered in the first increment are defined in detail, and that increment is developed. During development, further requirements analysis for later increments can take place, but requirements changes for the current increment are not accepted.

Once increment is completed and delivered, customers can put it into services they can experiment with the system that helps to clarify their requirements for later increments and for later versions of the current increment. As new increments are completed, they are integrated with existing increments so that the system functionality improves each delivered increment. The common services may be implemented early in the process or may be implemented incrementally as functionality is required by an increment.

(In incremental model, rather than deliver the system as a single delivery, the development and delivery is broken down into increments with each increment delivering part of the required functionality. User requirements are prioritised and the highest priority requirements are included in early increments. Once the development of an increment is started, the requirements are frozen though requirements for later increments can continue to evolve).



5. Discuss evolutionary prototyping and throw-away prototyping in the software process.

hide solution
asked in 2071

Solution

Evolutionary prototyping

In this method, the prototype developed initially is incrementally refined on the basis of customer feedback till it finally gets accepted. It helps us to save time as well as effort. That's because developing a prototype from scratch for every interaction of the process can sometimes be very frustrating.

This model is helpful for a project which uses a new technology that is not well understood. It is also used for a complex project where every functionality must be checked once. It is helpful when the requirement is not stable or not understood clearly at the initial stage.

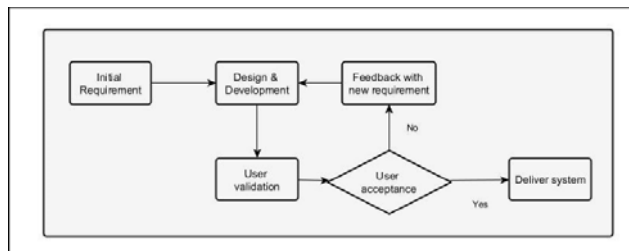


Fig: Evolutionary prototyping model

Throw-away prototyping

Throwaway prototyping is based on the preliminary requirement. It is quickly developed to show how the requirement will look visually. The customer's feedback helps drives changes to the requirement, and the prototype is again created until the requirement is baselined. In this method, a developed prototype will be discarded and will not be a part of the ultimately accepted prototype. This technique is useful for exploring ideas and getting instant feedback for customer requirements.

Throw-away Prototyping Model

Fig: Throw-away prototyping model

5. Discuss different types of rapid prototyping techniques.

hide solution
asked in 2072

Solution

Solution

Rapid prototyping involves creating a working model of various parts of the system at a very early stage, after a relatively short investigation. The model then becomes the starting point from which user can re-examine their expectations and clarify their requirements. When this goal has been achieved, the prototype model is thrown away and the system is formally developed based on the identified requirements.

Rapid prototyping techniques

Various techniques may be used for rapid development:

1. Using high level language:

- High level language include many powerful data management facilities.
- Using high level language we can create prototype with very little programming effort.
- Some languages offer excellent UI development facilities.

2. Reuse component:

The time need to develop a prototype can be reduced if many parts of the systems can be reused rather than designed and implemented. The reusable components may also be used in the final system thus reducing its development cost.

3. Ignore error handling:

In many system as much as one-half of the software is concerned with error handling. The time need to develop a prototype can be reduced If we ignore the error while designing prototype.

4. Omit features:

Omit features which are costly and takes large time to develop. So their omission make construct of prototype quicker.

5.Ignore functionality:

Only focus on establishing an acceptable user-interface.

6. Database programming language:

It includes a database query language, a screen generator, a report generator and a spreadsheet. These may be integrated with a CASE toolset. These are cost effective for small to medium sized business system.

5. What is rapid prototyping technique? Mention the types of rapid prototyping techniques and explain it.

hide solution
asked in 2074

Solution

Rapid prototyping involves creating a working model of various parts of the system at a very early stage, after a relatively short investigation. The model then becomes the starting point from which user can re-examine their expectations and clarify their requirements. When this goal has been achieved, the prototype model is thrown away and the system is formally developed based on the identified requirements.

Rapid prototyping techniques

Various techniques may be used for rapid development:

1. Using high level language:

- High level language include many powerful data management facilities.
- Using high level language we can create prototype with very little programming effort.
- Some languages offer excellent UI development facilities.

2. Reuse component:

The time need to develop a prototype can be reduced if many parts of the systems can be reused rather than designed and implemented. The reusable components may also be used in the final system thus reducing its development cost.

3. Ignore error handling:

In many system as much as one-half of the software is concerned with error handling. The time need to develop a prototype can be reduced If we ignore the error while designing prototype.

4. Omit features:

Omit features which are costly and takes large time to develop. So their omission make construct of prototype quicker.


5.Ignore functionality:

Only focus on establishing an acceptable user-interface.

6. Database programming language:

It includes a database query language, a screen generator, a report generator and a spreadsheet. These may be integrated with a CASE toolset. These are cost effective for small to medium sized business system.

5. What are rapid prototyping techniques? Briefly explain different rapid prototyping techniques.

 **hide solution**
asked in 2076

Solution

Prototyping is a technique for building a quick and rough version of a desired system or parts of that system. The prototype illustrates the system to users and designers. It allows them to see flaws and invent ways to improve the system. It serves as a communications vehicle for allowing persons who require the system to review the proposed user interaction with the system.

Rapid prototyping techniques

Various techniques may be used for rapid development:

1. Using high level language:

- High level language include many powerful data management facilities.
- Using high level language we can create prototype with very little programming effort.
- Some languages offer excellent UI development facilities.

2. Reuse component:

The time need to develop a prototype can be reduced if many parts of the systems can be reused rather than designed and implemented. The reusable components may also be used in the final system thus reducing its development cost.

3. Ignore error handling:

In many system as much as one-half of the software is concerned with error handling. The time need to develop a prototype can be reduced If we ignore the error while designing prototype.

4. Omit features:

Omit features which are costly and takes large time to develop. So their omission make construct of prototype quicker.


5.Ignore functionality:

Only focus on establishing an acceptable user-interface.

6. Database programming language:

It includes a database query language, a screen generator, a report generator and a spreadsheet. These may be integrated with a CASE toolset. These are cost effective for small to medium sized business system.

4. Why program are developed using evolutionary development are likely to be difficult to maintain? Explain.


 **hide solution**
asked in 2068

Solution

When a system is produced using the evolutionary development model, features tend to be added without regard to an overriding design. With each modification, the software becomes increasingly disorganized. System maintenance hampered by these problems, as it is harder identify the source of bugs in poorly designed systems. Also, keeping the documentation up to date over successive "evolution" is uncommon. Poor documentation also makes maintenance more difficult.

- It leads to implementing and then repairing way of building systems.
- Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.
- Incomplete application may cause application not to be used as the full system was designed.
- there results incomplete or inadequate problem analysis.

5. Explain why, for large systems development, it is recommended that prototypes should be throw-away prototypes.


 **hide solution**
asked in 2075

Solution

:**Large systems** are usually **developed** by different teams and these require a common reference framework (the **system** requirements) for **developing** the system. **Throwaway prototypes** can be used to help **develop** and validate these requirements to increase confidence that these are **appropriate**.

Large systems usually have a long lifetime, so maintainability is an important issue. Since prototypes generally undergo many changes during development, the resultant loss in design integrity ("structurdness") can make maintenance very difficult. So throwaway prototypes is used.

7. Explain the rapid prototyping techniques with example.

 **hide solution**
asked in 2068

Solution

Rapid prototyping involves creating a working model of various parts of the system at a very early stage, after a relatively short investigation. The model then becomes the starting point from which user can re-examine their expectations and clarify their requirements. When this goal has been achieved, the prototype model is thrown away and the system is formally developed based on the identified requirements.

Rapid prototyping techniques

Various techniques may be used for rapid development:

1. Using high level language:

- High level language include many powerful data management facilities.
- Using high level language we can create prototype with very little programming effort.
- Some languages offer excellent UI development facilities.

2. Reuse component:

The time need to develop a prototype can be reduced if many parts of the systems can be reused rather than designed and implemented. The reusable components may also be used in the final system thus reducing its development cost.

3. Ignore error handling:

In many system as much as one-half of the software is concerned with error handling. The time need to develop a prototype can be reduced If we ignore the error while designing prototype.

4. Omit features:

Omit features which are costly and takes large time to develop. So their omission make construct of prototype quicker.


5.Ignore functionality:

Only focus on establishing an acceptable user-interface.

6. Database programming language:

It includes a database query language, a screen generator, a report generator and a spreadsheet. These may be integrated with a CASE toolset. These are cost effective for small to medium sized business system.

9. Explain the rapid prototyping techniques.

 **hide solution**
asked in 2068(II)

Solution

Rapid prototyping involves creating a working model of various parts of the system at a very early stage, after a relatively short investigation. The model then becomes the starting point from which user can re-examine their expectations and clarify their requirements. When this goal has been achieved, the prototype model is thrown away and the system is formally developed based on the identified requirements.

Rapid prototyping techniques

Various techniques may be used for rapid development:

1. Using high level language:

- High level language include many powerful data management facilities.
- Using high level language we can create prototype with very little programming effort.
- Some languages offer excellent UI development facilities.

2. Reuse component:

The time need to develop a prototype can be reduced if many parts of the systems can be reused rather than designed and implemented. The reusable components may also be used in the final system thus reducing its development cost.

3. Ignore error handling:

In many system as much as one-half of the software is concerned with error handling. The time need to develop a prototype can be reduced If we ignore the error while designing prototype.

4. Omit features:

Omit features which are costly and takes large time to develop. So their omission make construct of prototype quicker.


5.Ignore functionality:

Only focus on establishing an acceptable user-interface.

6. Database programming language:

It includes a database query language, a screen generator, a report generator and a spreadsheet. These may be integrated with a CASE toolset. These are cost effective for small to medium sized business system.

5. Differentiate between V-shape model and spiral model.

 **hide solution**
asked in Model
Question

Solution

Answered by Sneha

The difference between V- model and Spiral model are as follows:-

V - MODEL


1. V model is software development model but development and testing are not concurrent.
2. In V-model testing activities start with the first stage.
3. Cost of V-model is expensive.
4. Flexibility of V model is little flexible.
5. Guarantee of success through V model is high.
6. User involvement in V-model is also only in beginning.
7. It is not iterative.

Spiral Model

1. Spiral model is a software development model and is made with features of incremental, waterfall or evolutionary prototyping models.
2. Testing is done in spiral model at the end of the engineering phase.
3. While cost of spiral model is also very expensive.
4. Flexibility to change in spiral model is not that difficult.
5. Guarantee of success through Spiral model is low.
6. In this user Involvement is only at the beginning.
7. It is iterative

12. Write short notes on (any two):

- a. User Interface Prototyping
- b. Software Inspection
- c. Source Code Translation

 **hide solution**
asked in 2069

Solution

a. User Interface Prototyping.

User interface (UI) prototyping is an iterative development technique in which users are actively involved in the mocking-up of the UI for a system. The aim of prototyping is to allow users to gain direct experience with the interface. Without such direct experience, it is impossible to judge the usability of an interface.

UI prototype involves two stages:

or prototype iterates the stages.

1. Early in the process, develop paper prototypes.
2. The design is then refined and increasingly sophisticated automated prototypes are then developed.

b. Software Inspection

Software inspection is a static verification & validation process in which a software system is reviewed to find errors, omissions and anomalies. Inspection is used to determine the defects in the code and remove it efficiently. This prevents defects and enhances the quality of testing to remove defects. This software inspection method achieved the highest level for efficiently removing defects and improving software quality.

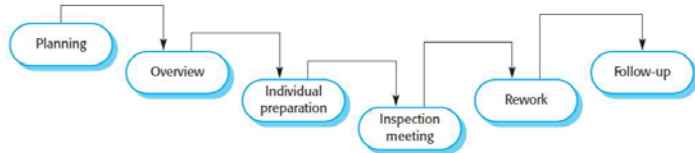


Fig: Inspection Process

c. Source Code Translation

The simplest form of software re-engineering is source code translation where source code in one programming language is automatically translated to source code in some other language. The structure and organisation of the program itself is unchanged. The target language may be an updated version of the original language (e.g. COBOL-74 to COBOL-85) or may be a translation to a completely different language (e.g. FORTRAN to C).

Program translation process:

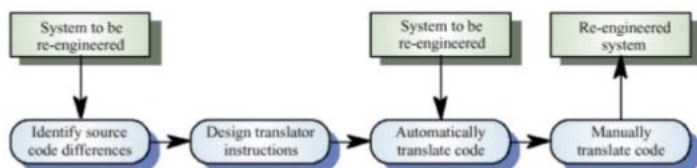


Fig: Program translation process

8. Explain the component base software engineering and its advantages.

asked in Model Question

Unit: 3 Agile Software Development questions

Unit: 4 Requirements Engineering 24 questions

Unit: 5 System Modeling 4 questions

Unit: 6 Architectural Design 12 questions

Unit: 7 Design and Implementation 10 questions

Unit: 8 Software Testing 26 questions

Unit: 9 Software Evolution 4 questions

Unit: 10 Software Management 14 questions

Suggest Us

Please give us feedback and suggestions to improve collegenote.

collegenoteofficial@gmail.com

Help

[About Us](#)

[Term & conditions](#)

[Privacy Policy](#)

Links and Resourses

[CSIT](#)

[BIT](#)

[BCA](#)

[Jobs and Internships](#)

[Blog](#)

[Contact](#)

Find us @

