Oral Part of The Exam

1. What is software engineering?

* Software engineering is an engineering discipline that deals with practical problems in the development of large software systems.

2. Define software process.

* A software process is a sequentially organized set of steps to create or edit a software asset.

3. Waterfall model of software process.

* The waterfall model is the basic model for the software process.
* It consists of successive steps - Analysis of requirements and their specifications, Software system design, Implementation (coding), Testing and product maintenance.

4. Modification of waterfall model.

* The waterfall model has its drawbacks. Some of them were solved by modification of the basic waterfall model - a model was created:
  + Incremental - contains a number of smaller waterfall models
  + Spiral - contains the phases of prototype creation and evaluation to verify system functionality

5. RUP (Rational unified process)

* RUP defines an approach to assigning tasks in a development organization. Its aim is to create a high quality product within a given time schedule and budget.
* RUP is a process product. It is developed and maintained by Rational Software and integrated with its suite of software development tools available from IBM.
* It consists of the following principles:
  + The software product is developed in an iterative way
  + Requirements are managed
  + Uses existing software components
  + The model is visualized
  + The quality of the product is continuously checked
  + System changes are controlled

6. Software development in an iterative way

* Iterations allow you to gradually refine the target product through incremental expansion, ie software is developed in versions that can be verified with the customer

7. Manage requirements

* The quality of the final product is determined by the degree of satisfaction of the client's requirements.
* The RUP process describes how to literally lure system requirements from the customer, how to organize and document them.

8. Development using components

* The RUP process provides a systematic approach to defining architecture using new or existing components.
* Components are interconnected on a case-by-case basis or through a component architecture using the Internet, CORBA or COM infrastructure.
* Development is shifting to product folding from prefabricated components.

9. Visualization of software system modeling

* The basis for successful application of visualization principles is considered as an industry standard
* Unified Modeling Language (UML) primarily for modeling purposes of software systems.

10. Software product quality verification

* The principle of product quality verification is included in the soft. process.
* They are used objective measurements and criteria (metrics) quantifying the quality of the final product. Therefore, quality assurance is not considered an activity outside the main development line.

11. Change management

* Change management ensures that every change is acceptable and all changes to the system are

traceable. The RUP process describes how to manage, monitor, and monitor changes to make it successful iterative development

12. Cycles, phases and iteration

* Each cycle results in a version of the system that can be passed on to users and implements the requirements they specify.
* Each phase can be divided into iterations.
* Iteration is a complete development loop leading to the creation of an executable version of the system representing a subset of the target product being developed and which is gradually extended by each iteration to into the final form

13. Static structure of the process

* Role: defines the behavior, competence and responsibility of an individual or group. Individuals are mapped according to how competences are compatible with their abilities.
* Artifacts: representing entities that are created, modified, or utilized in the process.
* Activities: (activities) performed by workers to create or modify artifacts.
* Flows: (workflow) activities representing sequences of activities creating the desired products.

14. UML language

* UML is a language for specifying, visualizing, constructing, and documenting software system artifacts.

15. Business modeling

* The purpose of business modeling is to provide a common language for software engineer communities and business professionals.
* Business process models describing a set of interconnected procedures and activities leading to meeting the business goal.
* Domain model affecting the most important objects occurring in the context system. Domain objects are entities existing in the environment in which the system works.

16. UML activity and class diagrams

* It uses the following diagrams for specification purposes:
  + Diagram of activities
  + Diagram classes

17. Activity diagram

* The activity diagram describes the individual processes using activities representing its (action) states and the transitions between them.

18. Class diagram

* The class diagram consists of two types of elements representing actively acting by the <<worker>> keyword and by passive entities as <<entity>>. These are linked by ties expressing a conceptual or physical connection. The multiplicity of these associations expresses the frequency of occurrence of the element.

19. Method BPM (Business Process Modeling)

* The process model is built using the BPM method from three relatively independent views formed by a functional, object and coordination model.
* Functional model: used to identify the process architecture, including its customers and products.
* The object model identifies a static structure containing all the entities (objects) that perform in the process.
* The coordination model builds on the previous two and describes how the process will be implemented through the specification and coordination of activities representing interactions between objects.

21. Requirements specification

* The aim of the requirements specification is to describe what the software system should do by specifying its functionality. Requirement specification models are used to reconcile assignments

between the development team and the sponsor.

22. Use case diagrams

* The purpose of the use case diagram is to define what exists outside the system being developed (actors) and what is to be done by the system (use cases). <<uses>> <<extends>>

21. Definition of the term object

* An object is an identifiable single entity given its:
  + identity - uniqueness to distinguish it from others
  + Behavior - services provided in interaction with other objects.
* Secondary Features:
  + attributes, existence, states

22. Relations between objects and their interactions

* A connection is a physical or conceptual link between objects. In relation to the defined cases of use, it is necessary to define such interactions between objects that will lead to the fulfillment of their functionality, the purpose for which they were designed.

22. Analysis and design

* The goal of the software product analysis and design flow is to show what the product will be implemented in the implementation phase.

23. Models and their diagrams

* The models that are created during analysis and design are as follows:
  + Analysis model - Design model
  + Deployment model
* These models are created using diagrams:
  + Diagram classes
  + Sequence diagram
  + Diagram cooperation
  + State diagram
  + Diagram deployment

24. Definition of the term class

* A class is a description of a set of objects having a common structure, behavior, relationships, and semantics.

25. Relationships between classes and objects

* Relationships between classes specify the way in which objects can communicate with each other.

26. Collaboration diagram

* Association describing a group of links (between objects) having a common structure and semantics. The relationship between association and connection is analogous to the relationship between class and object.
* Folding describing the relationship between a whole and its parts, where some objects define components whose composition creates a whole represented by another object. Dependency representing a weaker form of client-service relationship.
* Generalization is a taxonomic relationship between the more general element and its more specified element, which is fully consistent with the first of those listed only adds more specific information to its specification.

27. State diagram

* The state diagram displays the object's life cycle, events that cause transitions from one state to another, and the actions that result from this state change.

28. Package

* The package is a general-purpose mechanism that allows you to organize elements into groups to reduce the complexity of the modeled system.

29. Proposal and its aims

* The design model further refines the analysis model in the light of the actual implementation environment.

30. The resulting system architecture

* The implementation environment consists of software components that fall into the following three layers that complement the analysis objects related solely to the domain:
  + The user interface
  + Distribution
  + Persistence2

31. Design patterns and application frames

* Design patterns can be understood as an abstraction of imitating useful parts of other software products.
* Design patterns are divided into three groups:
  + creating
  + structural
  + behavior

32. Deployment diagram

* The deployment diagram describes how to configure the technical resources to run the software system.

33. Implementation

* The goal of the implementation is to complement the designed architecture (framework) of the application with the program code a to create a complete system. The implementation model specifies how the individual elements (objects and classes) created at the design stage are implemented in terms of software components, which are source codes, executable codes, data, and the like. A software component is defined as a physically existing and interchangeable part of the system satisfying the required set of interfaces and providing their implementation.

34. Testing and deploying a software product

* Testing is performed from the perspective of three basic dimensions represented by the quality, functionality and performance of the system. The testing concerns all created models and theirs diagrams.

35. Objectives of the verification and validation process

* Verification is a testing process to find out if a software product is being created correctly. In other words, we are looking for weaknesses in the software system itself.
* Validation is a testing process to find out if the software being created is correct. In other words, whether it implements the required functionality.

36. Testing models

* The test task specifies how to test the system, including information on what is to be tested with what inputs and under what conditions.
  + testing tasks defining what to test on the system
  + test procedures specifying procedures for performing test tasks,
  + testing components that automate test procedures.

37. Deployment of software system

* The purpose of the activity flow, called deployment, is to successfully create the end product and deliver it to end users.