Noorul Islam College of Engineering
Department of Software Engineering

Part A Questions and Answers

XSE 351 Software Architecture

S5 M.Sc. Software Engg.

Unit I

1. What is Software Architecture?

Software architecture involves the description of elements from which systems are built, interactions among those elements, patterns that guide their composition, and constraints on these patterns.

2. What is Engineering?

Engineering is a disciplined application of scientific knowledge to resolve conflicting constraints and requirements for problems of immediate, practical significance.

3. List the categories of Software architecture styles.

- Dataflow systems
- Call-and-return systems
- Independent components
- Virtual machines
- Data-centered systems

4 What are Pipes and Filters?

In a Pipe-and-filter architecture style, each component has a set of inputs and a set of outputs. A component reads streams of data on its inputs and produces streams of data on its outputs. The components are termed as **filters**. The connectors of this style serve output of one filter to input for another. The connectors are termed as the **pipes**.

5 Write the two important aspects of Data abstraction and Object-Oriented Organization style.

The two important aspects of Data abstraction and Object-Oriented Organization style are (1) An object is responsible for preserving the integrity of its representation and (2) The representation is hidden from other objects.

6 Write about the disadvantages of object-oriented systems.

The disadvantages of object-oriented systems are, if for one object to interact with another via procedure call, it must know the identity of that object. There can also be side-effect problems: if A uses object B and C also uses B, then C’s effects on B look like unexpected side effects to A, and vice versa.

7 Explain about Event based Implicit Invocation.

In an implicit invocation style a component can announce one or more events, instead of invoking a procedure directly. Other components can register an interest in an event by associating a procedure with it. When the event is announced, the system itself invokes all of the procedures that have been registered for the event. Thus an event announcement "implicitly" causes the invocation of procedures in other modules.

8. Write the disadvantages of implicit invocation.

The components relinquish control over the computation performed by the system. When a component announces an event, it cannot assume other components will respond to it. Another problem concerns exchange of data.

9. Write about layered system architecture.

A layered system is organized hierarchically, each layer providing service to the layer above it and serving as a client to the layer below. The components implement a virtual machine.

10. Name any two virtual machines.

- Interpreters
- Rule-based systems

11. "The type of transaction in an input stream triggers selection of process to execute". Identify the type of repository.

If the types of transaction in an input stream trigger selection process to execute, the repository can be a **traditional database**.

12. What do you mean by a Black board?

Black board is a repository in which the current state of the central data structure is the main trigger for selecting processes to execute. The black board is usually presented with three major parts:

1. The knowledge sources
2. The black board data structure
3. Control structure

13. What is an interpreter?

An interpreter includes the pseudoprogram being interpreted and the interpretation engine itself. The pseudoprogram includes the program itself and the interpreter's analog of its execution state. The interpretation engine includes both the definition of the interpreter and the current sate of its execution.

14. What are components of an interpreter?

The components of an interpreter include:

- An interpretation engine to do work
- A Memory that contains the pseudo code to be interpreted
- A representation of the control state of the interpretation engine

- A representation of the current state of the program being simulated.

15. What is a set point?

    Set point is the desired value for a controlled variable.

16. Define Open-loop system.

    Open-loop system is defined as the system in which information about process variables is not used to adjust the system.

17. Define Closed-loop system.

    Closed-loop system is defined as the system in which information about process variables is used to manipulate a process variable to compensate for variations in process variables and operating conditions.

18. Define Feedback control system.

    Feedback control system is defined as the controlled variable is measured, and the result is used to manipulate one or more of the process variables.

19. Define Feedforward control system.

    Feedforward control system is defined as the process variables are measured, and anticipated disturbances are compensated for without waiting for changes in the controlled variable to be visible.

20. What are process variables?

    Process variables are the properties of the process that can be measured; several specific kinds are often distinguished.

21. What are controlled variable?

    Controlled variables are the process variable whose value the system is intended to control.

22. What are manipulated variable?

    Manipulated variables are process variable whose value can be changed by the controller.

23. Define State transition systems.

    State transition systems are defined in terms of a set of states and a set of named transitions that move a system from one state to another.

24. Write about Domain-specific architecture.

    By specializing the architecture to the domain, it is possible to increase the descriptive power of structures. Such type of architectures are called Domain-specific architecture.

25. What is Heterogeneous architecture?

The combination of several "pure" architectural styles in several ways is called Heterogeneous architecture.

Unit – II

1. What is KWIC?

The KWIC (Key Word In Context) index system accepts an ordered set of lines; each line is an ordered set of words, and each word is an ordered set of characters. Any line may be "circularly shifted" by repeatedly removing the first word and appending it at the end of the line. The KWIC index system outputs a listing of all circular shifts of all lines in alphabetical order.

2. What are the software design changes proposed by Parnas?

- Changes in the processing algorithm
- Changes in the data representation

3. What are the considerations given by Garlan, Kaiser, and Notkin?

The considerations given by Garlan, Kaiser, and Notkin are as follows:

- Enhancement to system function
- Performance
- Reuse

4. Give reasons, why Main Program/Subroutine with shared data style is not suitable for KWIC?

A change in data storage format will affect almost all of the modules. Also changes in the overall processing algorithm and enhancements to system function are not easily accommodated. Finally this decomposition is not particularly supportive of reuse.

5. Why Abstract data type style is more suitable for KWIC?

Abstract data type style is more suitable for KWIC, because both the algorithms and data representations can be changed in individual modules without affecting others. Also reuse is better supported, because modules make fewer assumptions about the others with which they interact.

6. Is the Software design changes proposed by Parnas support for KWIC using implicit invocation? Give reason.

The software design changes proposed by Parnas can be obtained in KWIC using implicit invocation. Since the additional modules can be attached to the system by registering them to be invoked on data-changing events. Thus the solution obtains functional enhancement. Also the solution insulates computations from changes in data representation.

7. What is an Oscilloscope?

An Oscilloscope is an instrumentation system that samples electrical signals and displays pictures of them on a screen. Oscilloscopes also perform measurements on the signals, and display them on the screen.

8. Give the reason for moving from Pipe-and filter architecture to modified pipe-and-filter architecture in Oscilloscope.

The Pipe-and-filter architecture was not clear how the user should interact with it. In modified pipe-and-filter architecture the introduction of a control interface solves a large part of the user interface problem.

9. What are the basic requirements for a mobile robot's architecture?

- Deliberative and reactive behavior
- Uncertainty
- Account for dangers
- Flexibility

10. What are all the requirements supported by the control loop architecture in Mobile Robotics?

Account for dangers can be reduced, since simplicity makes duplication easy and reduces the chance of errors creeping into the system.

The major components of robot architecture such as supervisor, sensors, motors are separated from each other and can be replaced independently. Thus it satisfies the flexibility requirement.

11. Give the functions of sensor integration and sensor interpretation in layered architecture in Mobile robotics.

The sensor integration and sensor interpretation deals with input from the real world. They perform sensor interpretation by analyzing the data from the sensor and perform sensor integration by combining the analysis of different sensor inputs.

12. What is TCA?

TCA means Task-Control Architecture used to control numerous mobile robots. TCA is based on hierarchies of tasks, or task trees. In TCA the parent tasks initiate child tasks. The software designer can define temporal dependencies between pairs of tasks.

13. What are the functions supported by TCA's implicit invocation mechanisms?

TCA's implicit invocation mechanisms support three functions:

- Exceptions
- Wiretapping
- Monitors

14. What do you mean by wiretapping?

Messages can be intercepted by tasks superimposed on an existing task tree. For instance, a safety-check component can use this feature to validate outgoing motion commands.

15. What are Monitors?

Monitors read information and execute some action if the data fulfills a certain criterion. Monitors will offer a convenient way of dealing with fault-tolerance issues by setting aside agents to supervise the system.

16. What are components of the CODGER architecture?

- The components of the CODGER architecture are as follows:
- The “captain” : the overall supervisor.
- The “map navigator”: the high-level path planner.
- The “lookout”: a module that monitors the environment for landmarks.
- The “pilot”: the low-level path planner and motor controller.
- The perception subsystem: the modules that accept the raw input from multiple sensors and integrate it into a coherent interpretation.

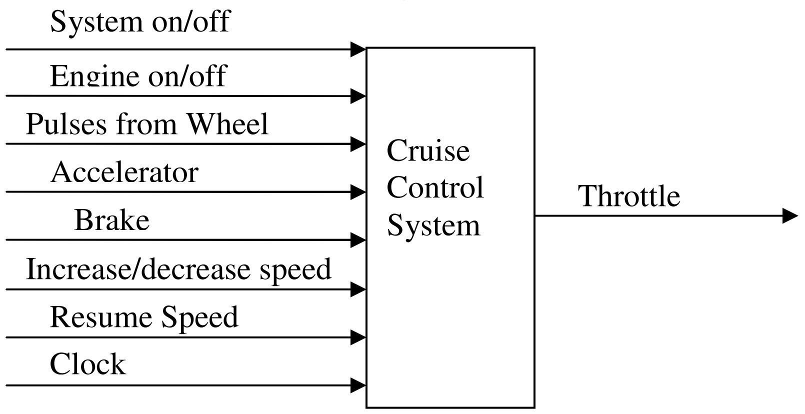
17. Name the architecture style that mostly supports the Mobile Robotics.

The Black board architecture is more suitable for mobile robotics, since it can saturate all the four requirements such as deliberative and reactive behavior, uncertainty, account for dangers, flexibility.

18. Which architectural style is followed by Cruise Control and why?

The Control- Loop architectural style is followed by Cruise Control. The architectural style is appropriate here because the software is embedded in a physical system that involves continuing behavior, especially when the system is subjected to external perturbations.

19. Illustrate the Booch Block Diagram for Cruise Control.



20. Briefly explain the Cruise Control Problem.

The Cruise Control Problem explains the problem in maintaining the speed of a car, even over varying terrain; also, whenever the system is active, determine the desired speed, and control the engine throttle setting to maintain that speed.

21. Identify the essential System elements of the Cruise Control System.

- Computational elements:

    - Process definition
    - Control Algorithm
- Data elements:
    - Controlled variable
    - Manipulated Variable
    - Set point
    - Sensor for controlled variable

22. How does the cruise controller set the throttle?

- On/Off Control
- Proportional Control
- Proportional plus Reset Control

23. Define Rule-based system.

A Rule-based system provides a means of codifying the problem-solving skills of human experts who tend to capture problem-solving techniques as sets of situation-action rules whose execution or activation is sequenced in response to the conditions of the computation rather than by a predetermined scheme.

24. Summarize the basic features of Rule-based systems.

The basic features are:

- It makes heavy use of pattern matching and context
- Adding special mechanisms for these facilities to the design yields the more complicated view.

25. Justify: The rule-based model followed by PROVOX is itself a design structure.

The rule-based model calls for a set of rules whose control relations are determined during execution by the state of the computation. It provides a virtual-machine to support this model.

<u>**Unit - III**</u>

1. What is Shared Information Systems?

Shared Information System is a significant class of system which is responsible for collecting, manipulating, and preserving large bodies of complex information.

2. What are the different domains appear in the form of shared information system?

- Data processing
- Software development environments
- Building design

3. What do you mean by Batch sequential database?

The application consisting of a small number of large stand-alone programs that are performed in sequential updates on flat or unstructured files are called Batch sequential databases.

4. What are the components of a batch sequential database?

- Edit program
- Transaction sort
- Update programs
- Print program

5. What are the two trends forced a movement away from batch sequential processing to repositories?

- Interactive technology
- As organizations grew, the set of transactions and queries grew.

6. Give the way in which the tools can interact with a shared repository.

- Tight coupling
- Open representation
- Conversion boxes
- No contact

7. Define the term tight coupling.

Tight coupling is defined as the shared detailed knowledge of the common, but proprietary, representation among the tools of a single vendor.

8. What do you mean by Open representation?

Open representation is publishing the representation, so that the tools can be developed by many sources. Often these tools can manipulate the data, but they are in a poor position to change the representation for their own needs.

9. Write the function of conversion boxes.

Conversion boxes provide filters that import or export the data in foreign representations.

10. What is the function of "No contact" interface in a repository?

"No contact" will prevent a tool from using the repository, either explicitly, through excess complexity, or through frequent changes.

11. What are the forces which led the evolution of shared information systems in software development?

- The advent of on-line computing
- The concern for efficiency
- The need for management to control entire software development process

12. Define Integrated environments for building design.

Integrated environments for building design are frame works for controlling a collection of stand-alone applications that solve part of the building design problem.

13. What the concerns in developing integrated environments for building designs?

- Efficient in managing problem solving and information exchange
- Flexible in dealing with changes to tools
- Graceful in reacting to changes in information and problem-solving strategies

14. What are the common responsibilities of a system for distributed problem solving?

- Problem partitioning
- Task distribution
- Agent control
- Agent communication

15. Give the characteristics of Batch sequential systems.

- Very coarse-grained
- Unable to provide feedback in real time
- Unable to exploit concurrency
- Unlikely to proceed at an interactive pace

16. Write the characteristics of pipe-and-filter systems.

- Fine-grained
- Able to produce the output in right way
- Able to provide feedback
- Often interactive

17. Define design space.

Design space consists of design rules that indicate good and bad combinations of choices, and use them to select an appropriate system designed based on functional requirements.

18. Define functional design space.

The dimensions that describe functional and performance requirements make up the functional design space.

19. Define structural design space.

The dimensions that describe structural and performance requirements make up the structural design space.

20. What are the type components in systems that are classified by the user-interface systems?

- An application-specific component
- A shared user interface
- A device-dependent

21. What the functional dimensions identified the requirements for a user-interface system?

- External requirements
- Basic interactive behaviour
- Practical considerations.

22. What are the dimensions in a computer organization that classifies the basic nature of the environment?

- Uniprocessing
- Multiprocessing
- Distributed processing

23. What the structural dimensions that represents the decisions determining the overall structure of a user-interface system?

- Division of functions and knowledge among modules
- Representation issues
- Control flow, communication, and synchronization issues

24. Define the term Monolithic program.

In a monolithic program there is no separation between application-specific and shared code, hence no such interface and no device interface. This can be an appropriate solution in small, specialized systems where the application needs considerable control over user-interface details and little processing power is available.

25. What do you mean by the term Nonpreemptive process?

Processes without preemptive scheduling, usually in a shared address space. These are only relatively simple to implement. Guaranteeing short response time is difficult and affects the entire system: long computations must be broken up explicitly.

Unit IV

1. What are the approaches to formalizing software architecture?
   Three approaches to formalizing software architecture are:
   - consider the nature of formal specification as it might apply to software architecture.
   - consider an example of architecture for a specific system.
   - Illustrate a quite different use of formalism as a way to understand the meaning of an architectural style and its associated design space.

2. Mention the features in the usage of Formalisms.
   Formalisms can be used to provide precise, abstract models and to provide analytical techniques based on these models. They can be used to provide

notations for describing specific engineering designs. They are also useful for simulating behavior.

3. Enumerate the different things that might be formalized.
   - The architecture of a specific system
   - An architectural style
   - A theory of software architecture.
   - Formal semantics for architectural description languages.

4. What are the problems faced by olden software systems?
   - No way to directly express the abstractions.
   - Module interconnection languages and the modularization facilities.

5. Mention the steps used to formalize a Digital Oscilloscope.
   The specification has been simplified
   The architecture of the system as a configuration of components connected functionally through inputs and outputs.

6. Explain the drawbacks in formalizing the architecture of a specific system.
   1. Underlying architectural style is not expressed explicitly
   2. High level of abstraction
   3. Architectural connection is implicit

7. Briefly describe the pipe-and-filter architectural style formally.
   Pipes are the connectors of the architecture and the filters are the components of the architecture.

8. Briefly explain an Architectural Design space formally.
   By assuming the existence of basic sets of events, methods, and component names, we can model an architectural component as an entity that has a name and an interface consisting of a set of methods and a set of events.

9. What is ‘Z’? Briefly explain its features.
   The ‘Z’ notation is a mathematical language developed mainly by the Programming Research Group. The notation uses standard logical connectives and set-theoretic operations with their standard semantics. Using the language of Z we can provide a model of a mathematical object.

10. What are the requirements for Architecture- Description Languages?

- The Linguistic Character of Architectural Description Languages
- Desiderata for Architecture-Description Languages
- Problems with Existing Languages

11. Mention the classes of components that appear regularly in architectural descriptions.

- Computation
- Memory
- Manager
- Controller
- Link

12. Briefly describe the critical elements of a design language.

- Components
- Operators
- Abstraction
- Closure
- Specification

13. Define desiderata. What are its features?
    The broad outlines of a system to support architectural design are relatively clear from informal experience. Such a system must provide models, notations, and tools to describe architectural components and their interactions. It must handle large-scale, high-level designs.

14. Mention the desiderata for ADLs.
    Composition
    Abstraction
    Reusability
    Configuration
    Heterogeneity
    Analysis

15. What are the notations for describing software architectures?

- Informal diagrams
- Modularization facilities
- Module interconnection languages
- Support for alternative kinds of interaction
- Specialized notations for certain architectural styles

16. Mention the language used to describe the use of module interfaces.

The language used to describe the use of module interfaces is the Module Interconnection Languages.

17. What is a First-Class connector? What are the problems with the current practice in the usage of these connectors?
    Architectural descriptions treat software systems as compositions of components. They focus on the components, leaving the description of interactions among these components implicit, distributed, and difficult to identify. If the interfaces to the components are explicit, they usually consist of import/export statements.

18. Why should abstractions be hidden from external view?
- Inability to associate related elements and name the cluster
- Inability to specify relations among related elements
- Inability to specify aggregate properties of a collection of elements.

19. Briefly explain the features of Components.
    Components roughly correspond to the compilation units of conventional programming language. Components are the loci of computation and state. Components may be either primitive or composite.

20. Briefly explain the features of Connectors.
- Connectors mediate interactions among components
- Connectors are the loci of relations among components.
- Connectors may be either primitive or composite

21. What are the advantages of using Connectors over Components?
- Connectors may be quite sophisticated
- The definition of a connector should be localized
- Connectors are potentially abstract
- Connectors may require distributed system support

22. Mention the numerous advantages to implicit invocations.
- It provides strong support for reuse
- It eases system evolution
- One module can be replaced by another

23. Briefly describe the two categories of implicit invocation systems.
- Tool-integration frameworks
- Implicit invocation systems based on special-purpose languages and application frameworks.

24. Explain the categories into which the design decisions are grouped into.
- Event definition
- Event Structure
- Event bindings
- Event announcement
- Concurrency
- Delivery policy

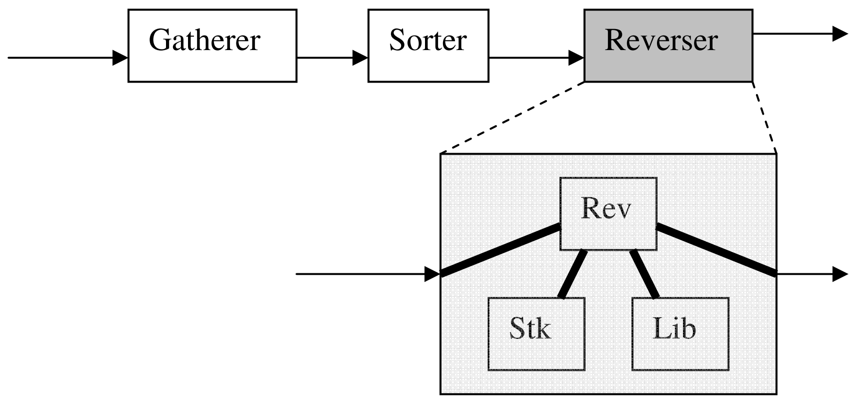
25. What are the three options in concurrency for describing the components?
    - Package
    - Packaged task
    - Free task

Unit V

1. Name three examples for research systems that aim to support architectural design and analysis.
   - UniCon: a language and set of tools for architectural description.
   - Aesop: a toolkit for constructing style-specific architectural design environments.
   - Wright: a language which supports architectural specification.

2. Briefly explain UniCon.
   UniCon is an architectural-description language intended to aid designers in defining software architectures in terms of abstractions that they find useful, and in making a smooth transition to code.

3. Illustrate the Heterogeneous implementation of a pipeline using both pipes and procedures.



4. Define the structure for Components and Connectors.
   Components contain the following:
   - Interface
   - Component type
   - Player
   - Implementation

   Connectors contain the following:
   - Protocol
   - Connector type
   - Role
   - Implementation

5. Define components in UniCon.

Components are the loci of computation and state. Each component has an interface specification that defines its properties, which include the component's type or subtype, functionality, guarantees about global invariants, performance characteristics etc.

6. Define Connectors in UniCon.

Connectors are the loci of relations among components. They mediate interactions; they provide the rules for hooking up.

7. What are the types of components?

The two types of components are primitive or composite.

Primitive components could be coded in a conventional programming language, in shell scripts of the OS or as software developed in an application such as a spreadsheet.

Composite components define configurations in a notation independent of conventional programming languages.

8. What are the parts of implementation for a composite element?
   - a parts list (components and connectors)
   - composition instructions (association between roles and players)
   - abstraction mapping (relation between internal players and players of the composite)
   - Other related specifications.

9. Name some of the idiomatic patterns of components and connectors.

Common system-composition idioms, such as pipeline, client-server, or blackboard, can be defined as idiomatic patterns, or styles of components and connectors.

10. Briefly explain the abstraction mapping of the component.

In case of abstraction mapping of the component, the definition must explicitly indicate the correspondence among one or more external players, one or more players of the constituent components, and the implementation rule.

11. What is the expansion of RMA?

RMA: rate-monotonic analysis: originated from research in scheduling algorithms for real-time systems.

12. What is Aesop?

Aesop is a system for developing style-oriented architectural design environments.

13. Name some of the features of Aesop.
   - a generic object model for representing architectural designs;
   - the characterization of architectural styles as specializations of the object model;
   - a toolkit for creating an open architectural design environment from a description of a specific architectural style.

14. Mention the categories into which architectural styles fall in.

The categories are:
   - Idioms and patterns

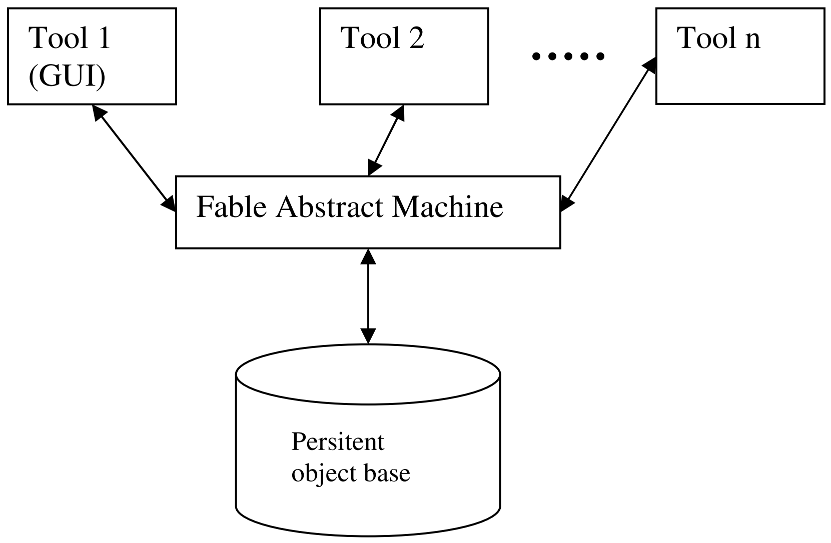
- Reference models

15. Name the Properties of architectural styles.
    - they provide a vocabulary of design elements
    - they define a set of configuration rules
    - they define a semantic interpretation
    - they define analyses

16. What are the significant benefits in the use of architectural styles?
    - it provides design reuse
    - Using architectural styles can lead to significant code reuse.
    - It is easier for others to understand a systems' organization if conventional structures are used.
    - Use of standardized styles supports interoperability.

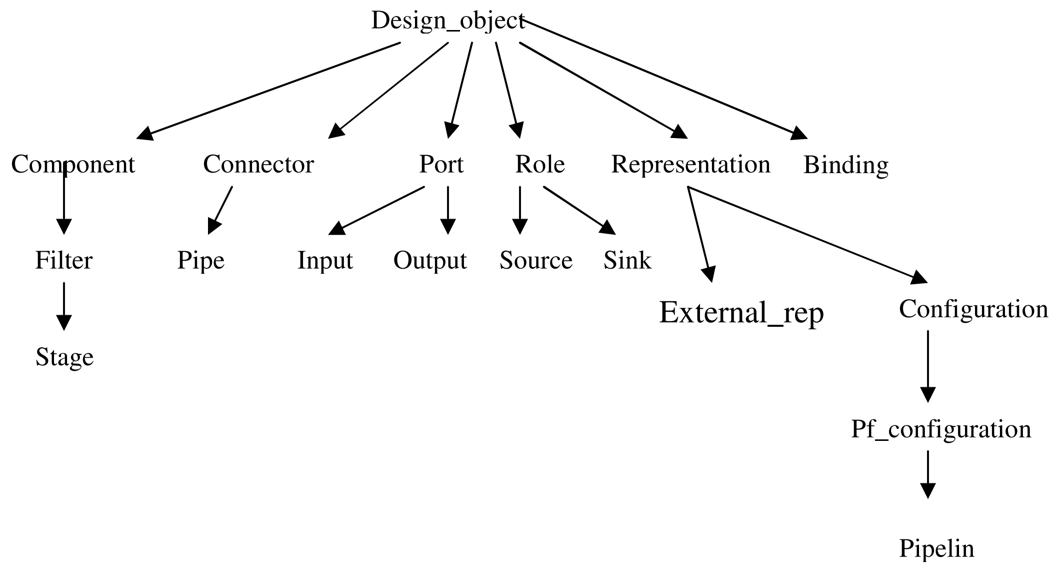
17. Define Fable.
Aesop combines a description of a style with a shared toolkit of common facilities to produce an environment, called a Fable, specialized to that style (or styles).

18. Illustrate the structure of a Fable.



19. Name the basic elements of architectural description.
- Components: Components are the loci of computation and state
- Connectors: Connectors are the loci of relations among components
- Configurations: they define the topologies of components and connectors.

20. Illustrate an example for style definition as subtyping.



21. Briefly explain the pipeline style.
It is a simple specialization of the pipe-and filter style. It incorporates all aspects of that style except that the filters are connected in a linear order with only one path of dataflow.

22. Differentiate the pipeline style and a real-time style.
The former is a simple specialization of the pipe-and –filter style. It uses all aspects of that style except that the filters are connected in a linear order with only one path of dataflow.
The latter defines three subtypes of component: devices, processes, resources.

23. Define an event-based style.
In an event-based style, components register their interest in certain kinds of events, and then can announce events and receive them according to their interest.

24. What is the expansion of MIL?
MIL stands for Module interconnection Language.

25. Briefly explain the key issue in the design of an MIL/IDL.
The nature of glue, which resolves the definition/, use relationships by indicating for each use of a facility where its corresponding definition is provided. Here, each module defines or provides a set of facilities that are available to other modules, and uses or requires facilities provided by other modules.