Unit 3

Date
Page

Architecture - model architecture

↳ System development is basically concerned with developing models of the certain system i.e identifying and describing objects in certain information space.

↳ There are several ideas regarding to find a good object, however good object doesn't exist on its own

↳ Another object can be perfectly right for one model but completly wrong for another model.

↳ The important criteria is that it should robust against modification and should help understand the system

We have to work with five diffrent models These models are the following

- The requirement model aims at capturing the functional requirements,
- The analysis model aims at giving the system a robust and changeable obj. structure
- The design model aims at adopting and refining the object structure to the correct implementation environment.
- The implementation model aims at implementing the system.
- The test model aims at verifying the system

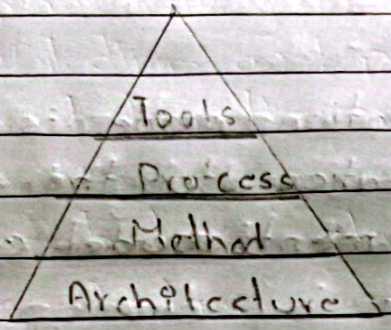


fig :- The architecture layer of OOSE

Each model tries to capture some part or aspect of the system to be built. These models are the output of the activities as shown in figure given bellow

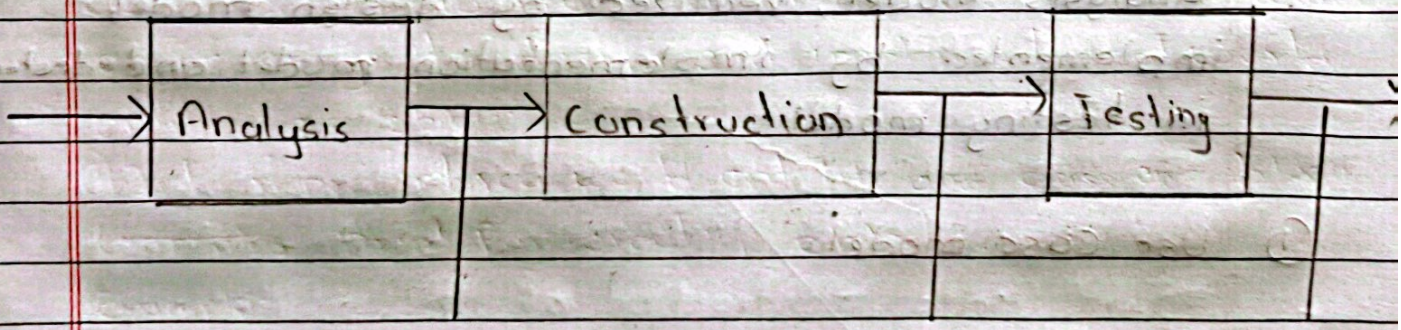


fig - The model developed associated with the process that produce them.

Requirements model

- The first transformation made is from the requirement specification to the requirements model. The requirement model consist of:

(i) A use case model
(ii) Interface descriptions.
(iii) A problem domain model

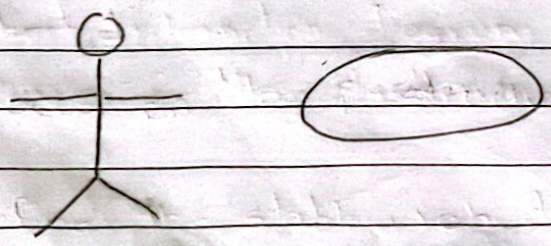
- Requirement model defines the system boundaries that should offer.

- The requirement model will be structured by analysis model, realized by design model, implemented by implementation model and tested in testing model.

(i) Use case model

A use case is specific way of using the system by performing some part of functionalities.

It consist of ~~us~~ actor and use case.



Actor

usecase.

These ~~sy~~ concept are simply an aid to defining what exists outside the system (actor) and what should be performed by the system (use case) The actor represent what interact with the system

The set of all use case description specifies ~~att use~~ to the complete functionality of the system To support ~~interface~~ the use case model it is often appropriate to develop also interface of the use case. A prototype of the user inter-face is a perfect tool.

To communicate with the potential user, and to get a ~~special~~ stable basis for the descriptions of the use cases, it is often appropriate to sketch a logical and surveyable domain object model of the system
Such an object model should consist of problem d omain object ~~mo~~ and serves as a support for the development of the requirements model

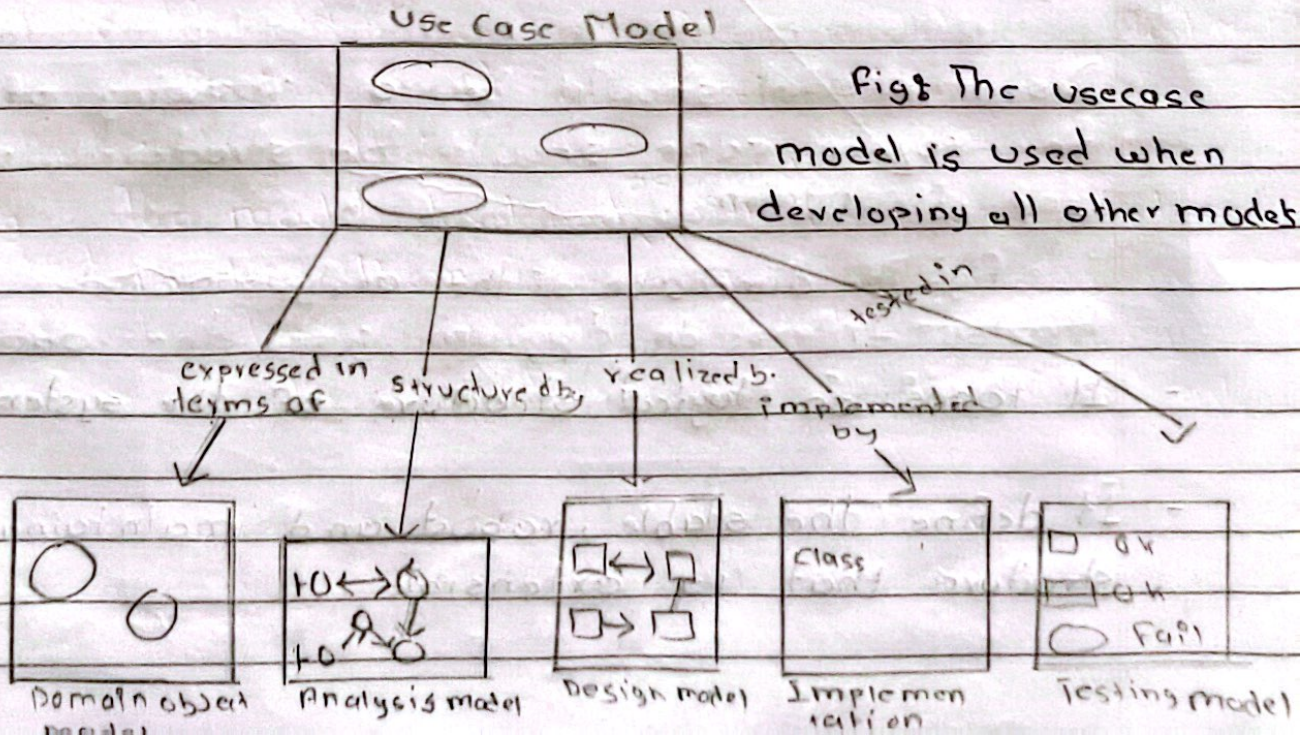


Fig: The usecase model is used when developing all other models

Analysis model

When the requirement model has been developed and approved by the system users, we can start to develop the actual system. This starts by developing the analysis model.

~~This means that we focus on the logical~~

This model aims at structuring the system ~~the~~ independently of the actual implementation of the system

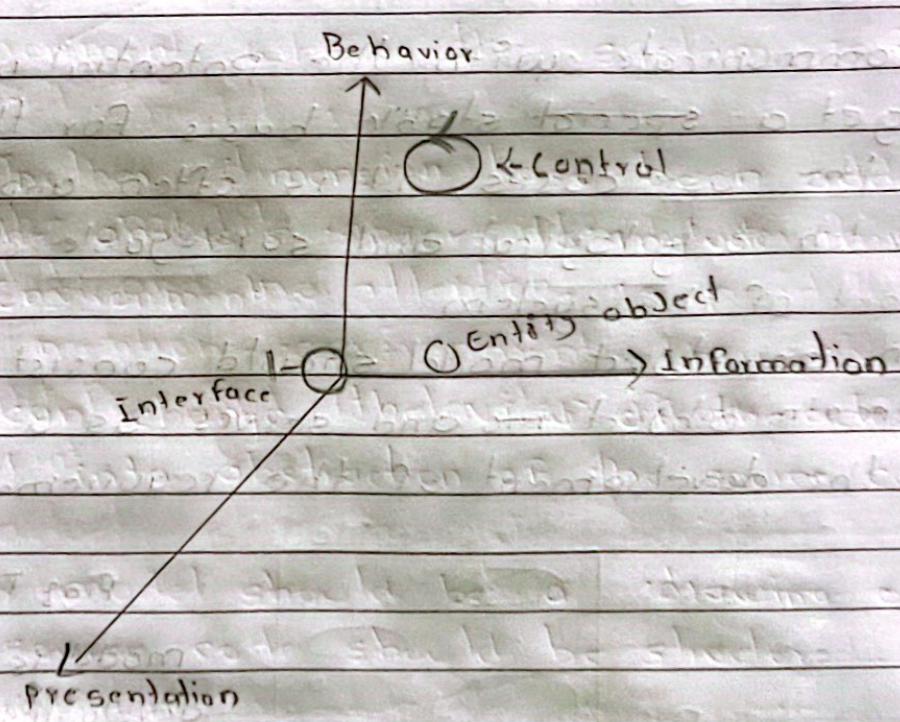
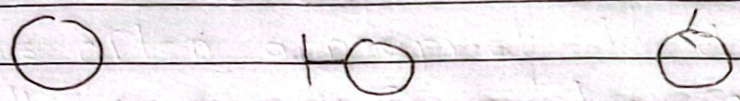


Fig:- The dimension of the analysis model

- It focus on logical structure of the system
- It define the stable, robust and maintainable structure that is extensible

It captures the behaviour, information and presentation

Entity objects, interface object and control object are types used to structure the system in the analysis model



Entity Object Interface object Control object

Fig: The object type used to structure the system in the analysis model

Functionality of usecase is structured by entity, interface and control objects.

Design model

Initially, we create a design model that is a refinement of and formalization of the analysis model. The initial work when developing the design model is to adapt to the actual implemention environment

System is designed during the construction process

Both analysis model and requirement model helps to create design model

Design model should take us closer to the actual source code

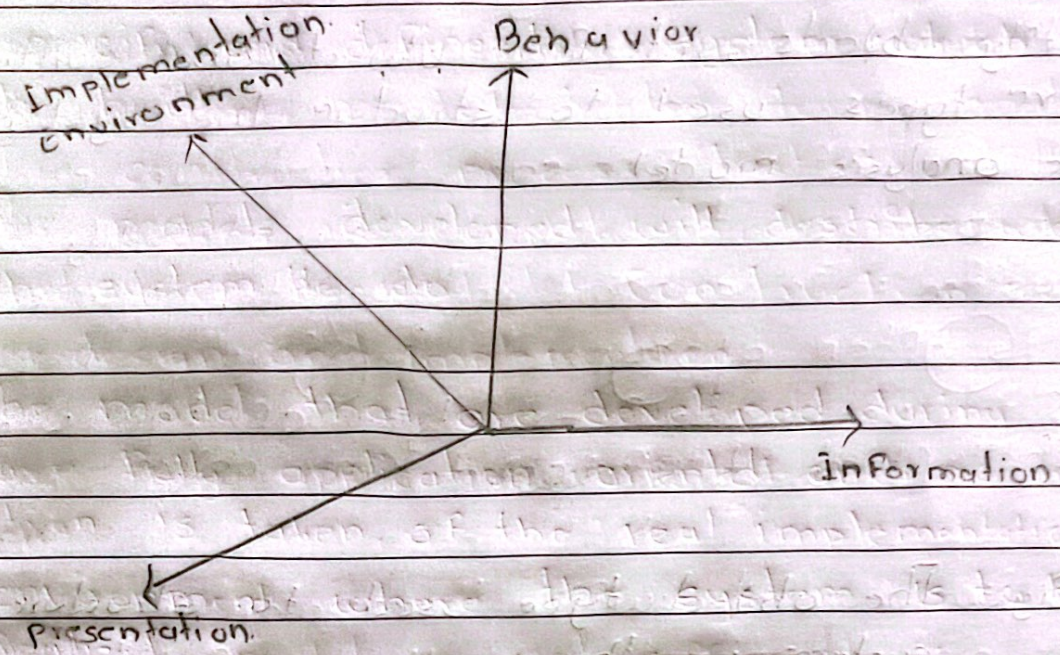


Fig:- Another dimension has been added to the analysis space to include implementation Environment

We can change the view of the design model into abstraction of the source code

Design model should be a drawing of how the source code should be structured, managed and written.

~~Abstract~~ Interaction diagram gives the abstract stimulf of the system.

The implementation model

↳ The implementation model consist of the annotated source code

↳ Specified the interface of each block and also have described of what is expected behind this interface.

↳ A very powerfull implementation tool is the ability to use component

↳ Components can be regarded as completed building element which are already placed in the implementation space

Test model

↳ The test model is the last model developed in the system development

↳ The fundamental concept in testing are mainly the test specification and the test results.

↳ What is tested is initially the lower levels, such as oblect modules and blocks. These are tested by the designer.

Analysis

The aim of analysis phase is to analyze, specify and define the system which is to be built

The models developed will describe what the system to do

The model that are developed during analysis are fully application oriented and no consideration is taken of the real implementation environment where the system is to be realized.

As the models are entirely problem oriented, and no attention is paid to the real implemention environment, they are fairly straight forward to develop from a functionality viewpoint

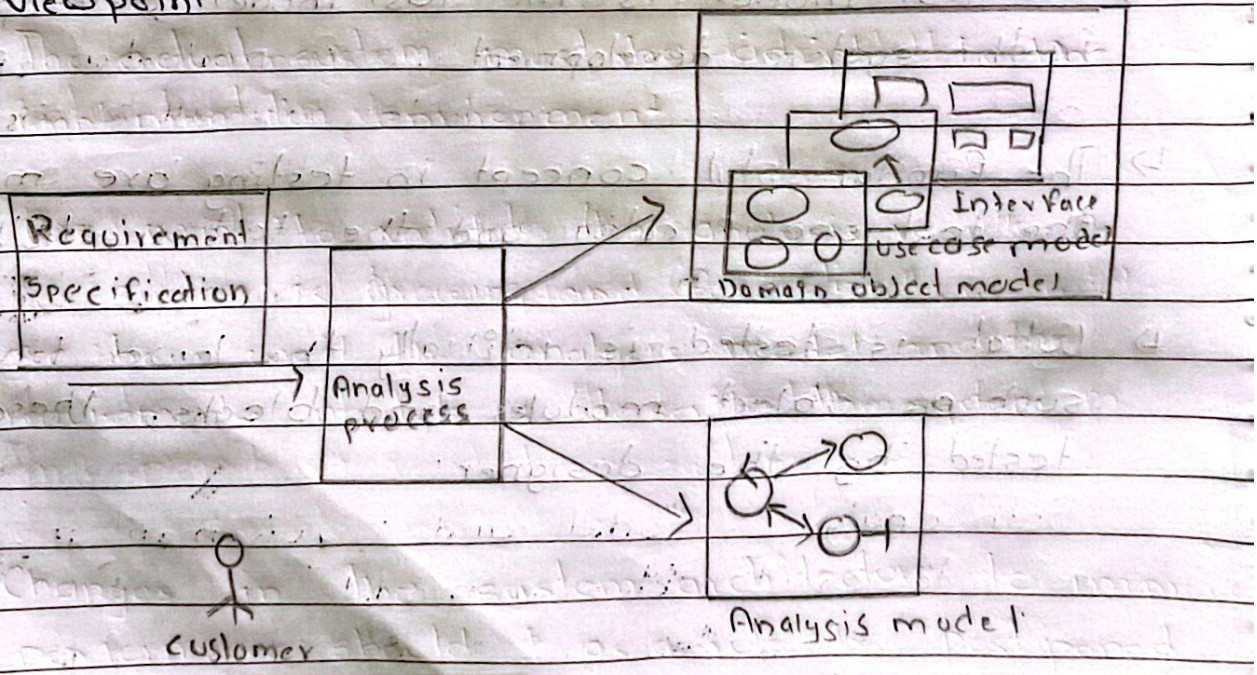


fig: Analysis Process

Two diffrent models are developed in analysis the requirements model and the analysis model

Requirement model define what functionality should take place with it

The analysis model gives a conceptual configuration of the system, consisting of control object entity object and interface objects

The purpose of this model is to develop a robust and extensible structure of the system as a base for construction.

The analysis model develop conceptual picture using problem domain objects

Analysis model gives a configuration of the system consisting of control, entity and interface objects

Requirement model based on user requirement aims delimpting the system and defining what functonality the system should offer

Construction

We build our system in construction, Construction is based on the analysis model and the requirements model created during analysis The construction process lasts until the coding is completed and the included units have been tested. Construction consists of design and implementaion.

The three main reason for having construction phase:

1) Analysis model is not sufficiently formal

- To change source code we must refine the object which operation should be offered, exactly what should the communication between the different objects looks like

2) The actual system must be adopted to the implementation environment

3) We want to validate the analysis results. As our system is growing and formalized, we will see how well the analysis model and the requirements models describe the system

Changes in the system architecture to improve performance should as a rule be postponed.

untill the system is being (partly) built

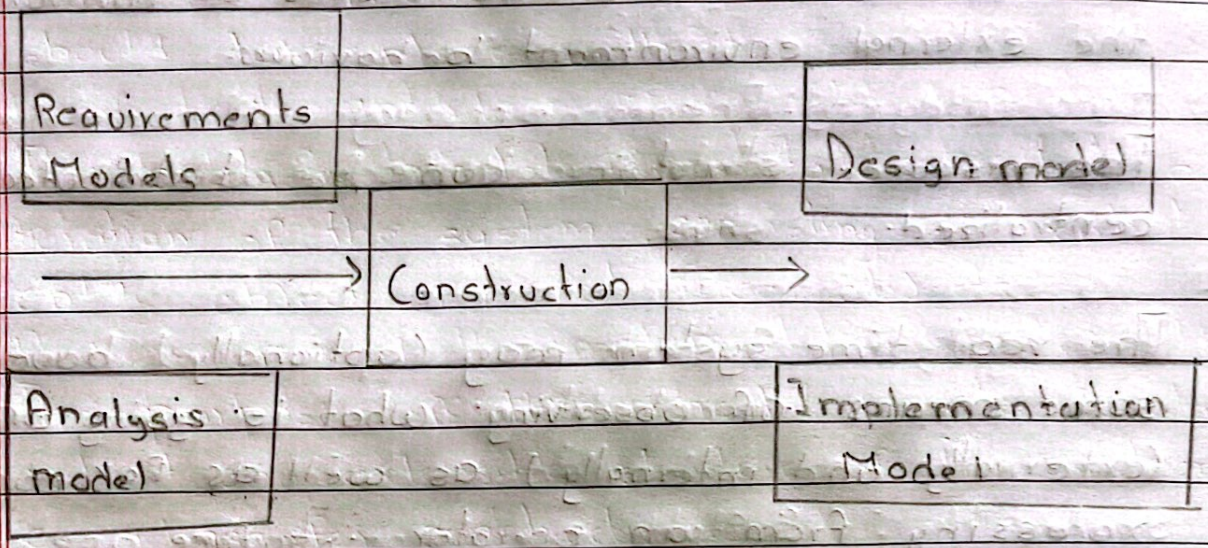


fig: The input and output model of Construction

Real Time classification of real time system

A real-time system is a system where the correctness ~~doesn't~~ depends not only on the logical results of computation but also on the time at which results are produced.

Ex: control of modern air craft, telephone exchange

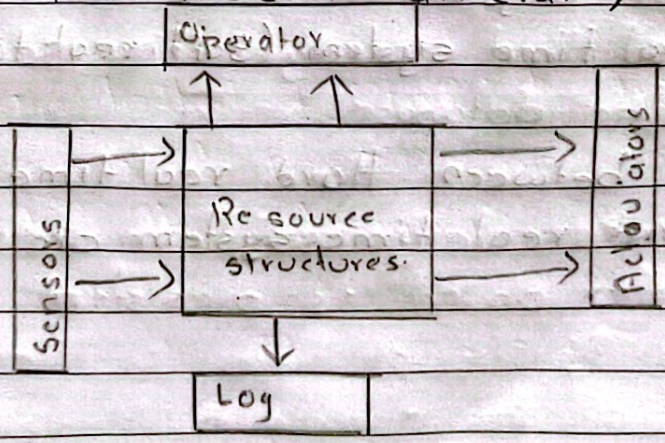


fig: Model of real time system

In general we can state that the sensor and actuator provide respectively a view of the application behaviour in external real time environment and the means of controlling the external environment behaviours.

The resource structure can be distributed or centralized.

The real time system may (optionally) have some means of observing what is going on (externally and internally) as well as for controlli processing from an operator interface

There may optionally be some means of keeping a history of what has transpired via some form of logging media

Classification of Real Time system

We can majorly categoriezed real time system into two parts.

(I) Hard real time system
(II) Non Hard real time system (Soft real time system)

The diffrence between Hard real time system and non hard real time system are

Hard real time system	Soft real time system
System with hard deadline that must meet otherwise a castastrophe can occur	Systems where services are provided in real time but castastrophe will not occur if service is not provided immediately
Deterministic predictibility of processes execution is essential	The execution of resource are stochastically distributed based upon the quantities of resource available and loading of system
Ex: control of modern aircraft airbag system, pacemakers etc	Digital telephone exchange gaming system, multimedia sys
Missing deadlines leads to critical failures.	Missing deadlines results in reduced performance

database - RDBMS

An RDBMS is a type of database management system that stores data in a strudured format using rows and columns

The feature of DBMS are

- Concurrency
- Recovery
- Query facilities

In relational database, information is stored in table

Advantages of RDBMS

Data integrity: Ensures accuracy and consistency of data through constraints and relationships.

Flexibility: Supports complex queries and data manipulation.

Scalability: Suitable for large scale application.

Security: provide robust security features to protect data.

Object DBMS

The relation model cannot capture the semantics of complex objects

The idea of an object DBMS (ODBMS) is store the objects so that bridge the sementic gap all way to the database

The ODBMS is very useful in application where complex object is presistant

we use the programming language to store and retrive data rather than DML.

What are the ~~first~~ criteria for an OODBMS?

Complex objects: It should support the notion of complex object.

Object identify: Each object shall have an identy independent of its internal values.

Encapsulation: It shall support encapsulation of data and behavior in the objects.

Hierarchies: It should support the notion of inheritance.

Late binding: It should support overridding and late binding

Types or classes.

Completeness

Extensibility

The major benifits of using OODBMS are

- The object can be stored in the database
- No conversion is needed for the DBMS type system
- The language of the DBMS can be integrated with an object oriented programming language.

Components

↳ A component is a standard bulding unit in an organization that is used to develop application.

↳ To build with components and to buildd with objects are two entierly diffrent activities

↳ To buld with components is bottom up activity.

↳ Component must be designed to be reused.

↳ To make a component reuseable in various applications, it must be independent of the application for which it was designed.

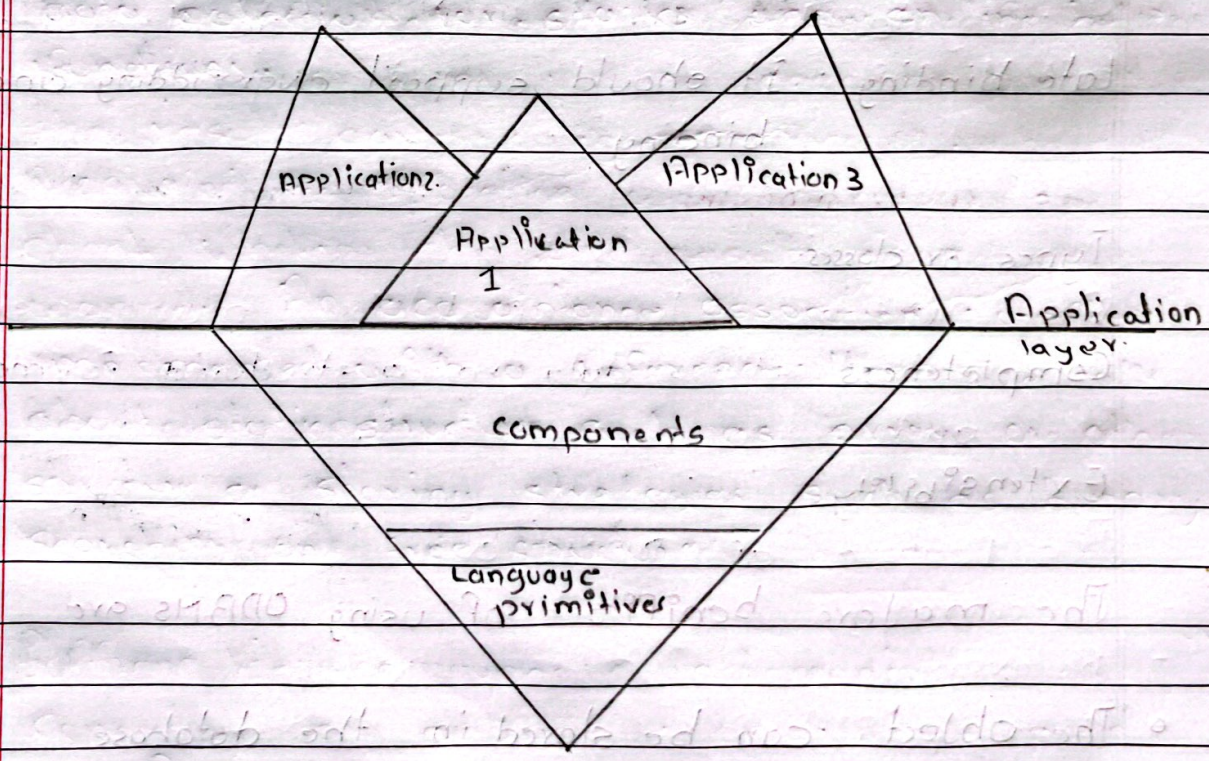


fig:- Components are used to build bottom-up

↳ It should provide general abstraction so that it can be widely used.

↳ Components need to be well tested and well documented so that it is accepted by all developers

↳ The components must have a well designed interface easy retreival and be accompained by good and easy to use documentation. The components should be packed for reuse.

↳ Components that one can modify according to one's need or component that needs to be modified before reusing it is called white box component

↳ Components that doesn't need to modify for use or component are designed to solve specific domain problem are called black box component.

Use of Components

Components should be viewed as an important part of the implemention language. Generally, everything could be implemented using components. However, often the right component does not exist and so it is not available for use.

The analysis and design models provide a strong frame work for finding where component can be used. Here are some example where components could be used.

General Entity Object: An entity object that are used to develop other entity object and whose information should be stored in a database, a general framework can be developed as component. Ex ORM (object relation Mapping)

Interface object:

Interface objects can be implement using components. Ex windows buttons and scroll bars

Control objects

Some control objects will have general functions such as logging activities, data collection for statics and on-line help functions.

Acquaintance assocation

An acquaintance assocation is often implemented with a reference. Here we talk about static structure such as array or other structure or storing strategies such as binary searching and tree structure.

Diffrent kinds of types

Diffrent kinds of types will occur at various phases for exampk attribute tyes, local types etc, when implementing the block

some of this types are general and can be implemented as components.

Component Management

↳ The development of components is often more expensive than the development of ordinary software.

↳ The real benefits comes when a component ~~should~~ could be managed in multiple projects and product ~~which~~. Therefore component management should be based on multiple projects.

↳ A special component management department or group is necessary, that is responsible for regulating component library for the organization

↳ With the use of OOP, it encourage the use of component.

↳ There are two types of activities for component management

i) One for the design of a complete component system

ii) One for construction of individual components.

Several various criteria for good and reuseable component designed have been purposed which are

- Reduce the number of parameters.
- Avoid the using options in parameters
- No direct access to instance variables.
- Naming should be consistent

Testing

The test activities are normally divided into verification and validations. Verification is the work involved in checking wheather the result agrees with specification, where as validation ~~user~~ is the work necessary to check whether the end results is actually wanted.

Verification: are we building the system correctly

validation: are we building the correct system

- Software development is incremental activity consisting of analysis, construction and testing.

↳ Testing can be done on document level by verifying that the analysis as well as design are done appropriately

↳ A failure occures when a program misbehave A fault exist on the program code when the code is ~~wor~~ wrong, the fault can be ~~cha~~ fixed by changing the code. An error is a human action that results in software containing fault

Test types

There are many types of test.

(i) Unit testing.
(ii) Integration testing.
(iii) System testing.
(iv) Regression testing
(v) Operational testing.
(vi) Full scale test
(vii) Performance test / capicity test
(viii) stress test
(ix) negative test
(x) Test based on requirement specification
(xi) Ergonomic test
(xii) Testing of user documentation
(xiii) Acceptance testing (Alpha testing, beta testing)

Here we will only discussed about three type of testing (unit, Integration, System testing) as per syllabus

1) Unit testing

Unit testing means that one and only one unit is tested as such ~~unit~~. This test requires that the unit is independent of other units.

11) Integration testing

Integration testing involves tests with the purpose of verifying that the units are working together correctly.

111) System testing

System testing is a level of software testing where a complete and integrated software system is tested as whole. The purpose of this test is to evaluate the system's compliance with the specified requirements.

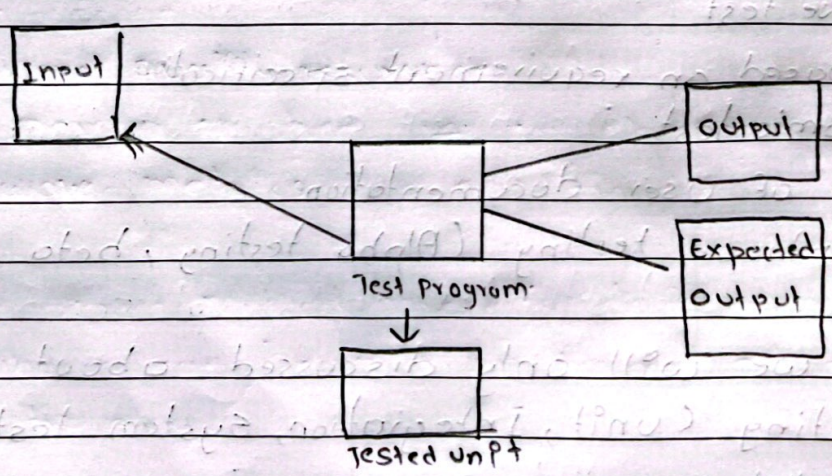


Fig:- Schematic illustration of automated testing

Testing techniques

The goal should be to automate as much as possible of the testing. This can be done through special test programs with the associated test data

The test program fetches sequence and data from input data. Then the unit/system is fed with the sequence and the tested system's response is observed by the test program.

There are two types of testing techniques.

(i) Functional Testing techniques.
(ii) Non-functional Testing techniques