Engineering in One Video (EIOV)

Watch video on YouTube

B.Tech

Software Engineering

KCS-601

With Notes

UNIT-3

Software Design

(in one video)

AKTU Exam

Engineering in One Video (EIOV)

Watch video on YouTube

Topics to be covered...

Concept of Software Design
Architectural and Procedural Design
Modularization
Design Structure Charts
Pseudo Codes
Coupling and Cohesion Measures
Function Oriented Design
Object Oriented Design
Top-Down and Bottom-Up Design
Measurement and Metrics
Halestead’s Software Science
Function Point (FP) Based Measures
Cyclomatic Complexity Measures
Happy Ending!

EIOV

Engineering

IN ONE VIDEO

CONNECT, LEARN AND GROW

Engineering in One Video (EIOV)

Watch video on YouTube

Concept of Software Design

Concept of Software Design

Software design is a mechanism to transform user requirements into some suitable form, which helps the programmer in software coding and implementation.

- It allows the software engineer to create the model of the software that is to be developed.
- The software design phase is the first step in SDLC (Software Design Life Cycle), which moves the concentration from the problem domain to the solution domain.

General task involved in design process:

1. Design the overall the system processes.
2. Segmenting the system into smaller modules.
3. Designing the database structure.
4. Documenting the system design.

Concept of Software Design

Objectives:

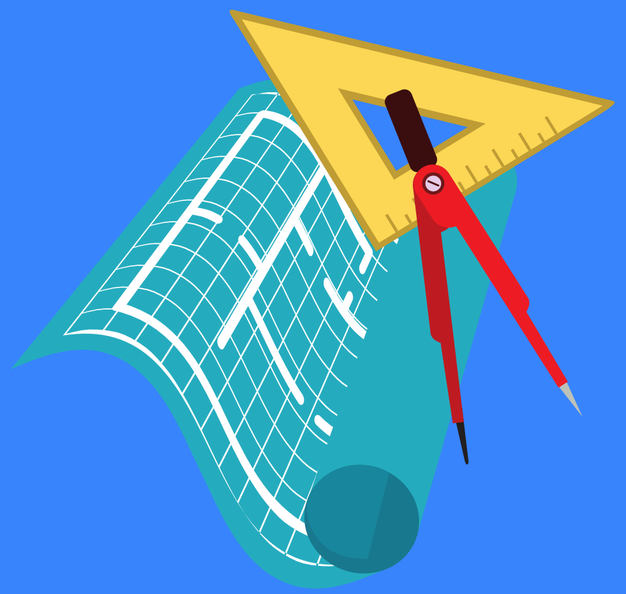
1. **Correctness:** Software design should be correct as per requirement. It should correctly implement all the functionalities of the system.
2. **Completeness:** The design should have all components like data structures, modules, and external interfaces, etc.
3. **Efficiency:** Resources should be used efficiently by the program.
4. **Flexibility:** Able to modify on changing needs.
5. **Consistency:** There should not be any inconsistency in the design.
6. **Maintainability:** The design should be so simple so that it can be easily maintainable by other designers.

Concepts of Software Design:

The software design concept simply means the idea or principle behind the design. It describes how you plan to solve the problem of designing software, the logic, or thinking behind how you will design software. It allows the software engineer to create the model of the system or software or product that is to be developed or built.

Engineering in One Video (EIOV)

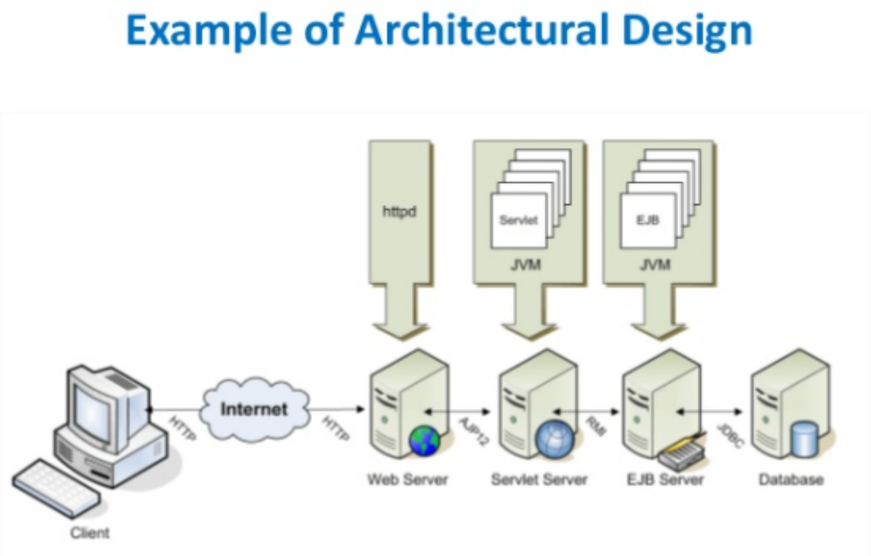
Watch video on YouTube



Architectural and Procedural Design

Architectural Design

- The process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system.
- The output of this design process is a description of the software architecture.
- Architectural design is an early stage of the system design process.
- It represents the link between specification and design processes and is often carried out in parallel with some specification activities.

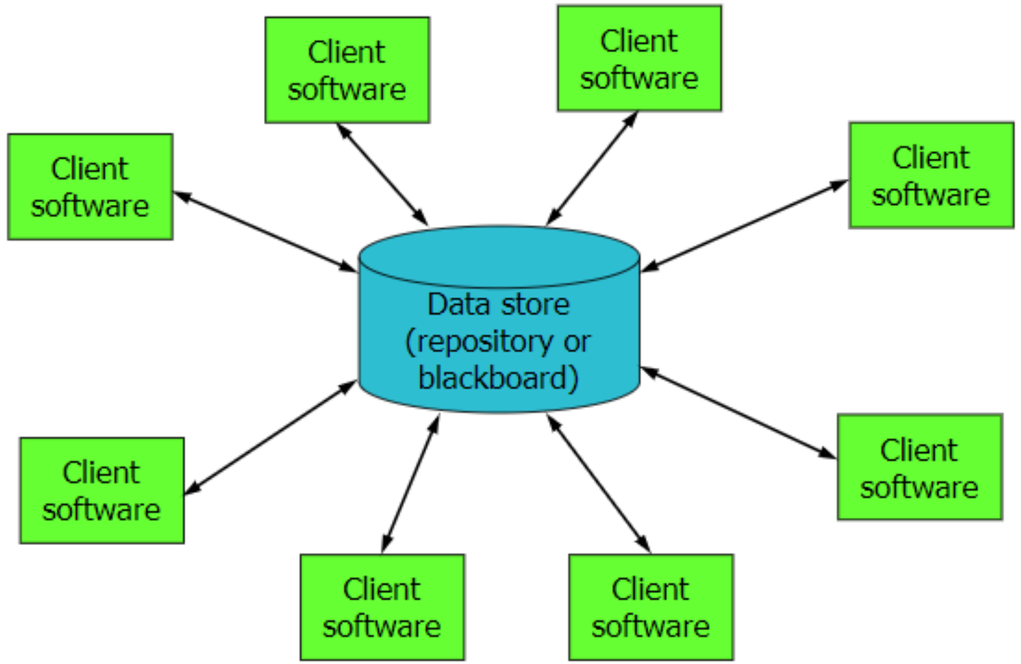


- A set of components(eg: a database, computational modules) that will perform a function required by the system.
- The set of connectors will help in coordination, communication, and cooperation between the components.
- Conditions that how components can be integrated to form the system.

Architectural Design

Data Centered Architecture:

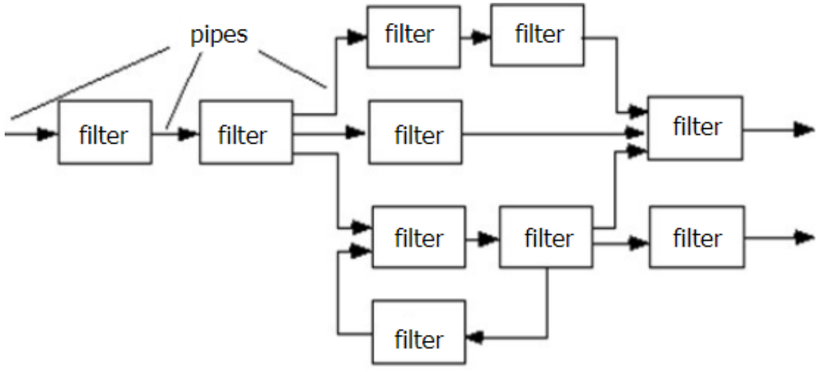
- Data Centered Architecture is a layered process which provides architectural guidelines in data center development.
- Data Centered Architecture is also known as Database Centric Architecture.
- This architecture is the physical and logical layout of the resources and equipment within a data center facility.



Architectural Design

Data flow architectures:

- This kind of architecture is used when input data to be transformed into output data through a series of computational manipulative components.
- The figure represents pipe-and-filter architecture since it uses both pipe and filter and it has a set of components called filters connected by pipes.
- Pipes are used to transmit data from one component to the next.



(a) pipes and filters

(b) batch sequential

Procedural Design

- Procedural design used to model programs that have an obvious flow of data from input to output.
- It represents the architecture of a program as a set of interacting processes that pass data from one to another.
- Procedural design is also called components design. It is completely based on process and control specification.
- The "state transition diagram" of the requirement analysis model is also used in components design.
- Components design is usually done after user interface design.

Engineering in One Video (EIOV)

Watch video on YouTube

Modularization

Modularization

- Modularity in design refers to the splitting of a large software system into smaller connected modules.
- Modules are interconnected through their interfaces.
- The interconnection should be simple to avoid costly maintenance.
- Two modules are directly connected if one module can call the other module.

Modularity has several key benefits:

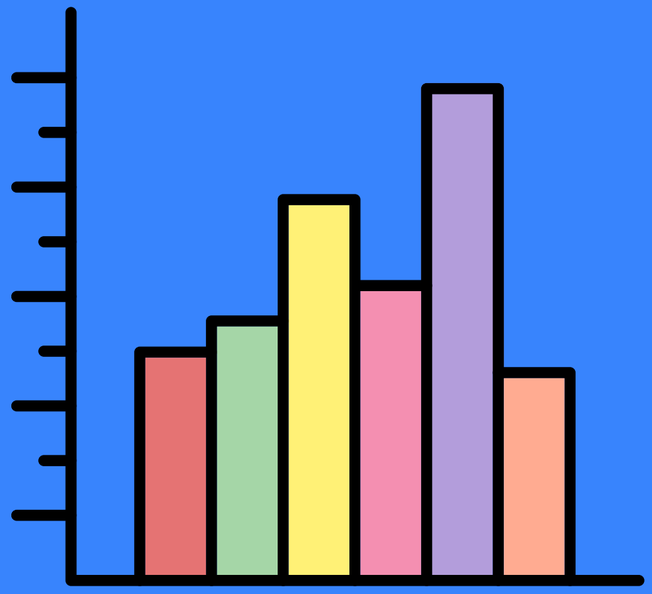
- **Testing & Debugging** Since each component is self-contained, you mitigate dependency issues. It becomes easy to test each component in isolation by using a mocking or isolation framework.
- **Reusability** If you discover you need the same functionality in a new project, you can package the existing functionality into something reusable by multiple projects without copying and pasting the code.
- **Extensibility** Your software now runs as a set of independent components connected by an abstraction layer.

Advantage of modularization

- Smaller components are easier to maintain
- Program can be divided based on functional aspects
- Easier because programmer only focus on one small simple problem rather than a large complex problem.
- Testing and Debugging easier.
- Easy to isolate bugs and easy to fix bugs.

Engineering in One Video (EIOV)

Watch video on YouTube



Design Structure Charts

Design Structure Charts

- A Structure Chart (SC) in software engineering and organizational theory is a chart which shows the breakdown of a system to its lowest manageable levels.

- They are used in structured programming to arrange program modules into a tree. Each module is represented by a box, which contains the module's name.

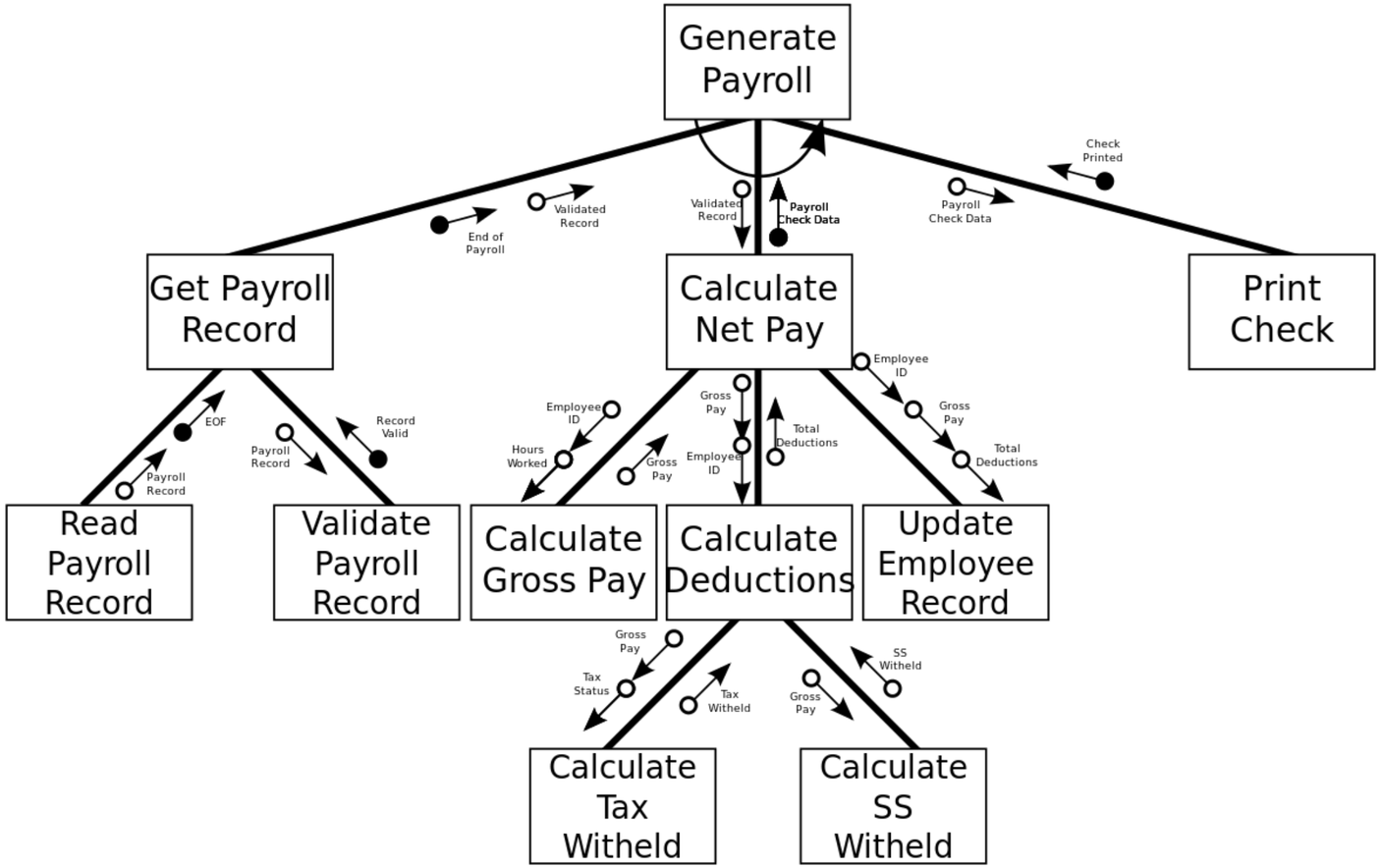
- The tree shows the relationship between modules, showing data transfer between the models. Data being passed from module to module that needs to be processed.

- Structure chart is a chart derived from a Data Flow Diagram. It represents the system in more detail than DFD. It breaks down the entire system into lowest functional modules, describes functions and sub-functions of each module of the system to a greater detail than DFD.

Engineering in One Video (EIOV)

Watch video on YouTube

Design Structure Charts



Symbol	Name
Module Name	Process
	Data Couple
	Flag

Engineering in One Video (EIOV)

Watch video on YouTube

Pseudo Codes

EIOV

Engineering

IN ONE VIDEO

CONNECT, LEARN AND GROW

Pseudo Codes

- Pseudo code uses the vocabulary of one language i.e, English and syntax of another i.e, any structured programming language.
- Pseudo code is a combination of algorithm written in a simple language and programming language statements.
- Using Pseudo code, the designer design describes system characteristics structured by keywords such as IF-THEN-ELSE, while-do and end.

Simple design tools

- Flow chart

- Pseudo-code
  - Wait for alarm
  - Count = 1
  - While (not ready to get up and count <= 3)
    - Hit snooze button
    - Increment count
  - Climb out of bed

Advantages of Pseudo Codes

- Converting a Pseudo code to a programming language is much easier as compared to converting to flowchart or decision table.
- It is easier to modify whenever programmer modification are necessary.
- Take less time to writing as compared to drawing flow chart.
- Done easily on Word Processor

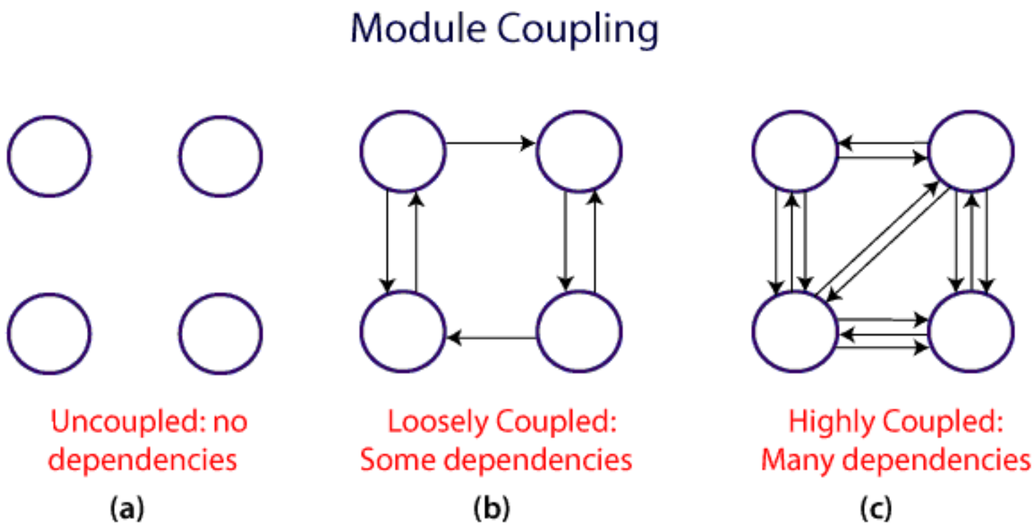
Engineering in One Video (EIOV)

Watch video on YouTube

Coupling and Cohesion Measures

Coupling

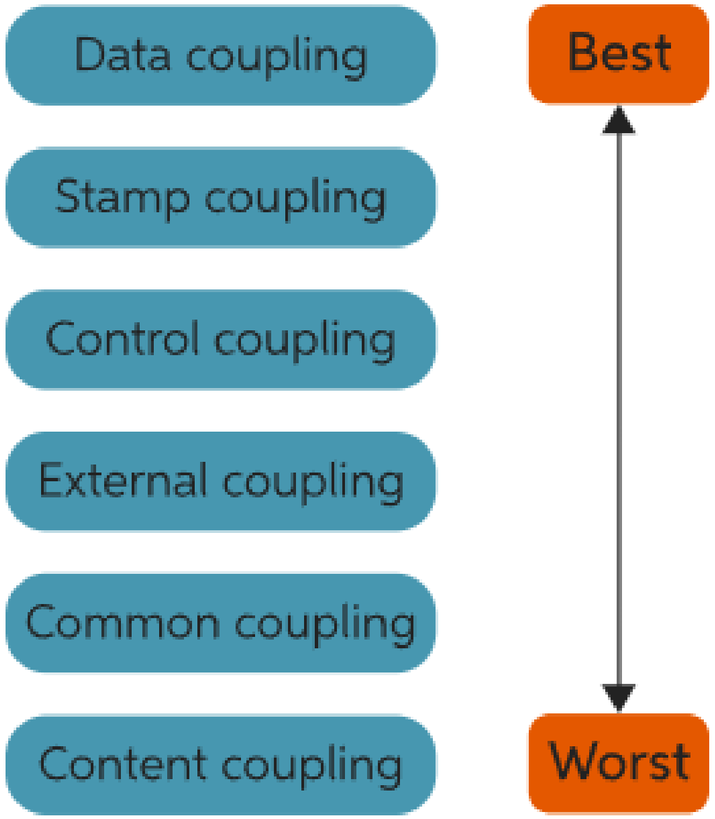
The coupling is the degree of interdependence between software modules. Two modules that are tightly coupled are strongly dependent on each other. However, two modules that are loosely coupled are not dependent on each other. Uncoupled modules have no interdependence at all within them.



A good design is the one that has low coupling. Coupling is measured by the number of relations between the modules. That is, the coupling increases as the number of calls between modules increase. Thus, it can be said that a design with high coupling will have more errors.

Coupling

Types of Module Coupling:



Coupling

Data Coupling:

When data of one module is passed to another module, this is called data coupling.

Stamp Coupling:

Two modules are stamp coupled if they communicate using composite data items such as structure, objects, etc.
When the module passes entire structure to another module, they are said to be stamp coupled.
For example, passing structure variable in C or object in C++ language to a module.

Coupling

Control Coupling:

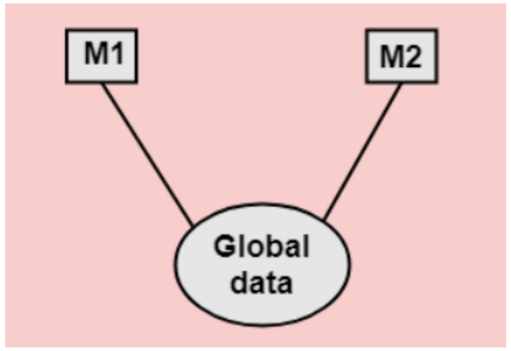
Control Coupling exists among two modules if data from one module is used to direct the structure of instruction execution in another.

External Coupling:

External Coupling arises when two modules share an externally imposed data format, communication protocols, or device interface. This is related to communication to external tools and devices.

Common Coupling:

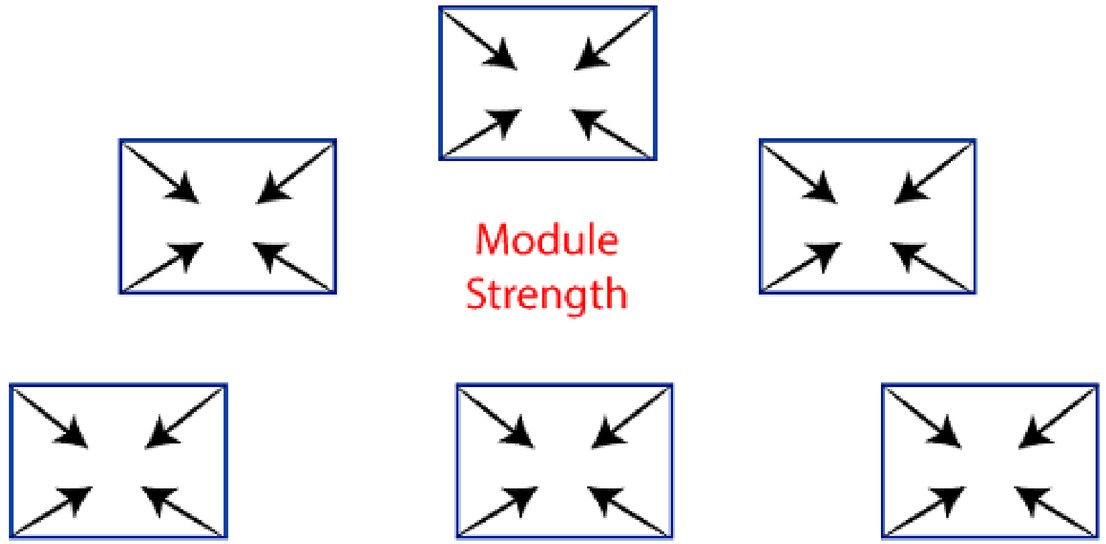
Two modules are common coupled if they share information through some global data items.



Cohesion

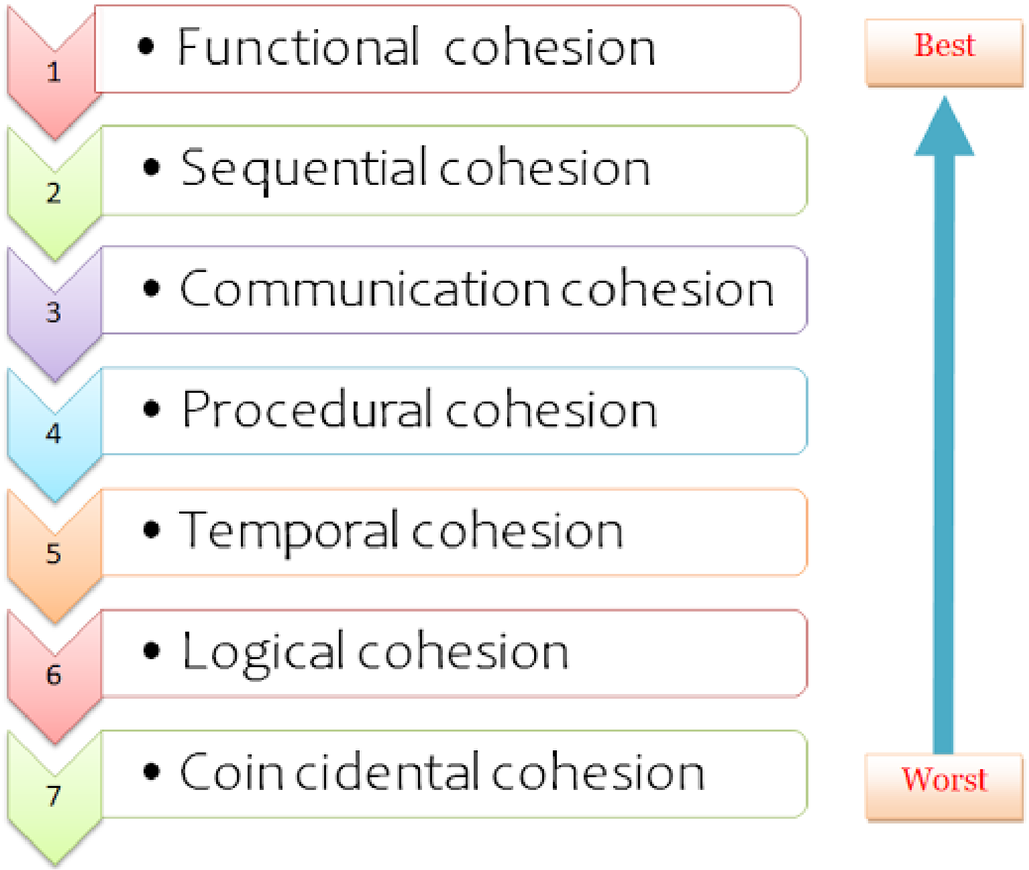
Cohesion shows the relationship within the module.

Cohesion is an ordinal type of measurement and is generally described as "high cohesion" or "low cohesion."



Cohesion= Strength of relations within Modules

Types of Cohesion



Cohesion

1. **Functional Cohesion:** Functional Cohesion is said to exist if the different elements of a module, cooperate to achieve a single function.
2. **Sequential Cohesion:** If the element of a module form the components of the sequence, where the output from one component of the sequence is input to the next.
3. **Communicational Cohesion:** If all tasks of the module refer to or update the same data structure, e.g., the set of functions defined on an array or a stack.
4. **Procedural Cohesion:** If the set of purpose of the module are all parts of a procedure in which particular sequence of steps has to be carried out for achieving a goal, e.g., the algorithm for decoding a message.
5. **Temporal Cohesion:** When a module includes functions that are associated by the fact that all the methods must be executed in the same time.
6. **Logical Cohesion:** If all the elements of the module perform a similar operation. For example Error handling, data input and data output, etc.
7. **Coincidental Cohesion:** If it performs a set of tasks that are associated with each other very loosely, if at all.

Engineering in One Video (EIOV)

Watch video on YouTube

Coupling

Coupling is also called Inter-Module Binding.

Coupling shows the relationships between modules.

In coupling, modules are linked to the other modules.

Cohesion

Cohesion is also called Intra-Module Binding.

Cohesion shows the relationship within the module.

In cohesion, the module focuses on a single thing.

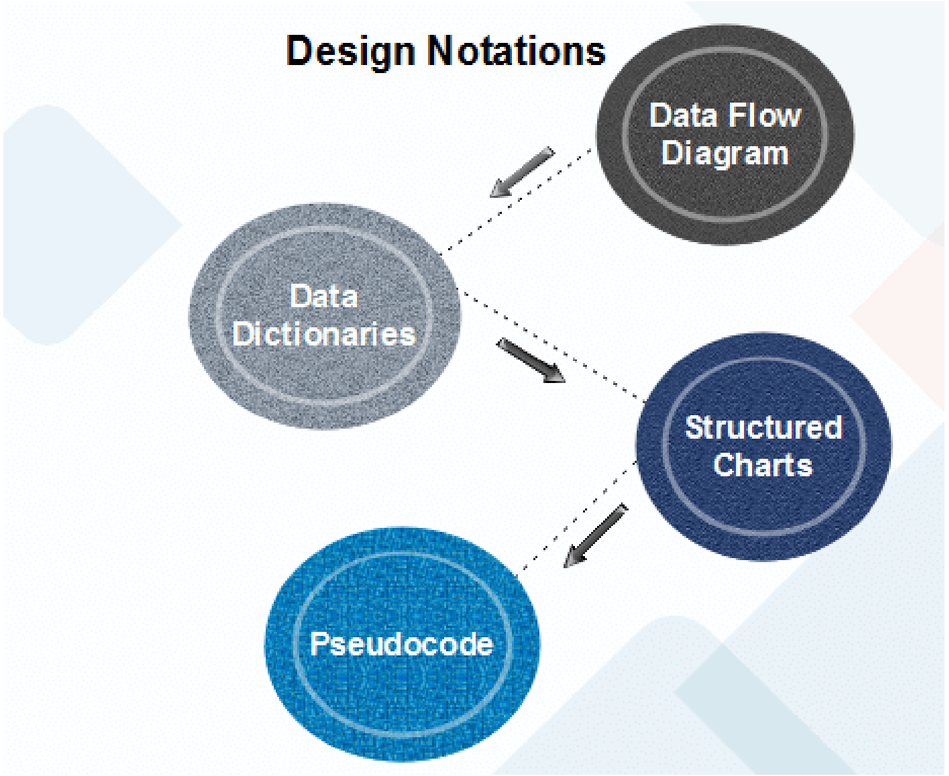
Engineering in One Video (EIOV)

Watch video on YouTube

Function Oriented Design

Function Oriented Design

- Function Oriented Design is an approach to software design where the design is decomposed into a set of interacting units where each unit has a clearly defined function.



Function Oriented Design

Techniques used by Function Oriented Design are:

- **Data Flow Diagram**(A data flow diagram (DFD) maps out the flow of information for any process or system.)
- **Data Dictionaries**(Data dictionaries are simply repositories to store information about all data items defined in DFDs. At the requirement stage, data dictionaries contain data items.)
- **Structure Charts**(Components are read from top to bottom and left to right. When a module calls another, it views the called module as black box, passing required parameters and receiving results)
- **Pseudo Codes**(It uses keywords and indentation. Pseudo codes are used as replacement for flow charts. It decreases the amount of documentation required.)

Engineering in One Video (EIOV)

Watch video on YouTube

Object Oriented Design

EIOV

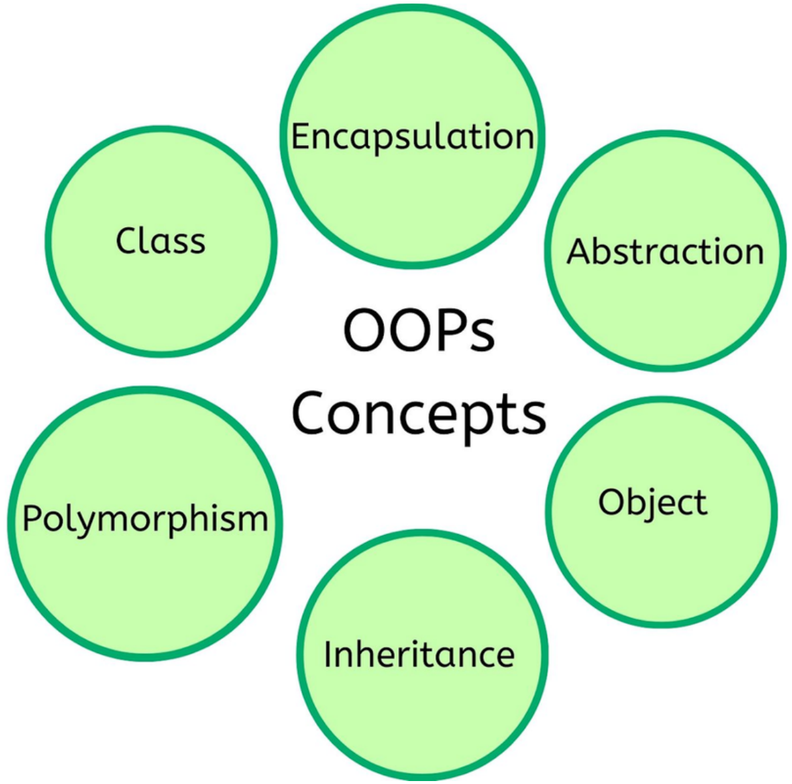
Engineering

IN ONE VIDEO

CONNECT, LEARN AND GROW

Object Oriented Design

- Object-oriented design is the process of planning a system of interacting objects for the purpose of solving a software problem. It is one approach to software design.
- In the object-oriented design method, the system is viewed as a collection of objects (i.e., entities).



Engineering in One Video (EIOV)

Watch video on YouTube

Top-Down and Bottom-Up Design

Engineering in One Video (EIOV) Watch video on YouTube

Top-Down and Bottom-Up Design

S.No.	TOP DOWN APPROACH	BOTTOM UP APPROACH
1.	In this approach We focus on breaking up the problem into smaller parts.	In bottom up approach, we solve smaller problems and integrate it as whole and complete the solution.
2.	Mainly used by structured programming language such as COBOL, Fortran, C, etc.	Mainly used by object oriented programming language such as C++, C#, Python.
3.	Each part is programmed separately therefore contain redundancy.	Redundancy is minimized by using data encapsulation and data hiding.
4.	In this the communications is less among modules.	In this module must have communication.
5.	It is used in debugging, module documentation, etc.	It is basically used in testing.
6.	In top down approach, decomposition takes place.	In bottom up approach composition takes place.
7.	In this top function of system might be hard to identify.	In this sometimes we can not build a program from the piece we have started.
8.	In this implementation details may differ.	This is not natural for people to assemble.

Engineering in One Video (EIOV)

Watch video on YouTube

Measurement and Metrics

Measurement

- A software metric is a measure of software characteristics which are measurable or countable. Software metrics are valuable for many reasons, including measuring software performance, planning work items, measuring productivity, and many other uses.
- Within the software development process, many metrics are that are all connected. Software metrics are similar to the four functions of management: Planning, Organization, Control, or Improvement.

Need of Software Measurement:

- Software is measured to:
- Create the quality of the current product or process.
- Anticipate future qualities of the product or process.
- Enhance the quality of a product or process.
- Regulate the state of the project in relation to budget and schedule.

Metrics

A metric is a measurement of the level that any impute belongs to a system product or process.

There are 4 functions related to software metrics:

1. Planning
2. Organizing
3. Controlling
4. Improving

Metrics

Classification of Software Metrics:

There are 3 types of software metrics:

Product Metrics:

- Product metrics are used to evaluate the state of the product, tracing risks and undercovering prospective problem areas. The ability of team to control quality is evaluated.

Process Metrics:

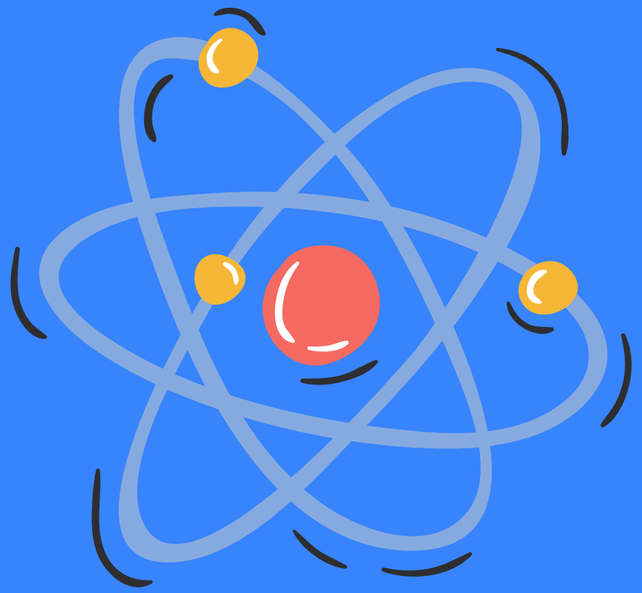
- Process metrics pay particular attention on enhancing the long term process of the team or organization.

Project Metrics:

- The project matrix describes the project characteristic and execution process.
- Number of software developer
- Staffing pattern over the life cycle of software
- Cost and schedule
- Productivity

Engineering in One Video (EIOV)

Watch video on YouTube



Halestead’s Software Science

Engineering in One Video (EIOV) Watch video on YouTube

Halestead's Software Science

Statement:

A computer program is an implementation of an algorithm considered to be a collection of tokens which can be classified as either operators or operand.

- All software science metrics can be defined in terms of these basic symbols. These symbols are called as a token.
- The basic measures are
  - n1 = count of unique operators.
  - n2 = count of unique operands.
  - N1 = count of total occurrences of operators.
  - N2 = count of total occurrence of operands.
  - Size of the program can be expressed as N = N1 + N2.
- **Program Volume (V)**

  The unit of measurement of volume is the standard unit for size "bits." It is the actual size of a program if a uniform binary encoding for the vocabulary is used. $V=N*\log_2 n$
- **Program Level (L)**

  The value of L ranges between zero and one, with L=1 representing a program written at the highest possible level (i.e., with minimum size).

  $L=V^*/V$
- **Program Difficulty**

  The difficulty level or error-proneness (D) of the program is proportional to the number of the unique operator in the program.D= (n1/2) * (N2/n2)

```
void sort ( int *a, int n ) {
int i, j, t;

  if ( n < 2 ) return;
  for ( i=0 ; i < n-1; i++ )  {
        for ( j=i+1 ; j < n ; j++ ) {
              if ( a[i] > a[j] ) {
                    t = a[i];
                    a[i] = a[j];
                    a[j] = t;
              }
        }
  }
}
```

$V = 80 \log_2(24) \approx 392$

- Ignore the function definition
- Count operators and operands

3	<	3	{
5	=	3	}
1	>	1	+
1	-	2	++
2	,	2	for
9	;	2	if
4	(	1	int
4	)	1	return
6	[]		

1	0
2	1
1	2
6	a
8	i
7	j
3	n
3	t

	Total	Unique
Operators	N1 = 50	n1 = 17
Operands	N2 = 30	n2 = 7

Watch video on YouTube

Measurement

Advantages:

- Predicts error rate.
- Predicts maintenance effort
- Simple to calculate
- Measure overall quality
- Used for any language

Disadvantages:

- Depends on complete code
- Complexity increases as program level decreases
- Difficult to compute

Engineering in One Video (EIOV)

Watch video on YouTube

Function Point (FP) Based Measures

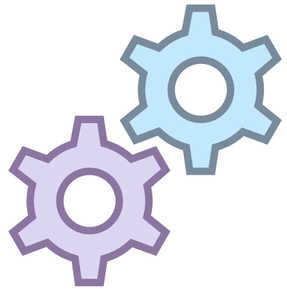
Function Point (FP) Based Measures

Function point is used in the estimation of software development cost which is the most important potential use of function point data.

Function point may be used to compute the following importance metrics:

Productivity = FP/persons-months
Quality = Defects/FP
Cost = Rupees/FP

Organizations that use function point method develop criteria for determining whether a particular entry is low, average or high.



Function Point (FP) Based Measures

There are three types of transaction functions.

- External Inputs
- External Outputs
- External Inquiries

Advantages:

- Size oriented metrics
- Language dependent
- Understood by the non technical user
- To estimate cost and resources required for software development

Disadvantages:

- Manually Counting Process
- Difficult to Understand
- Requires Experience

Function Point (FP) Based Measures

Types of FP Attributes

Measurements Parameters	Examples
1.Number of External Inputs(EI)	Input screen and tables
2. Number of External Output (EO)	Output screens and reports
3. Number of external inquiries (EQ)	Prompts and interrupts.
4. Number of internal files (ILF)	Databases and directories
5. Number of external interfaces (EIF)	Shared databases and shared routines.

Engineering in One Video (EIOV)

Watch video on YouTube

Cyclomatic Complexity Measures

Cyclomatic Complexity Measures

Cyclomatic complexity is a software metric used to measure the complexity of a program. It gives the quantitative measure of logical complexity of the program.

How to Calculate Cyclomatic Complexity?

McCabe proposed the cyclomatic number, V (G) of graph theory as an indicator of software complexity. The cyclomatic number is equal to the number of linearly independent paths through a program in its graphs representation. For a program control graph G, cyclomatic number, V (G), is given as:

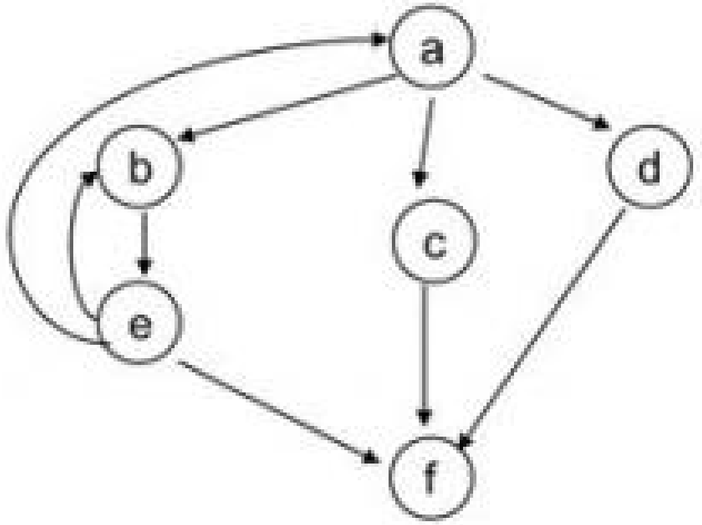
$$V(G) = E - N + 2 * P$$

E = The number of edges in graphs G
N = The number of nodes in graphs G
P = The number of connected components in graph G.

Cyclomatic Complexity Measures

Source code

```
a:      switch ...
        case b
b:              ...
                do e while ...
c:              ...
        case d
d:              ...
e:      if ...
                jump a
        case c
f:      end
```



Cyclomatic complexity

v (G) = e - n + 2 p

= 9 - 6 + 2 = 5

= Number of different sections of the control flow graph

= Number of binary decisions + 1

Cyclomatic Complexity Measures

Advantages:

- Gives complexity of various designs
- Computed early in life cycle
- Easy to apply
- Measures minimum effort

Disadvantages:

- Measures program's control complexity and not the data complexity
- Nested conditional structures are harder to understand
- Ignore the size of the program

EIOV

Engineering

IN ONE VIDEO

CONNECT, LEARN AND GROW

Happy Ending!

Congratulations!