SE
U-III

1. Discuss about the design concepts in a software development process.

1. Abstraction:-

which is used in all engineering disciplines.

It is a tool that permits a designer to consider a component at an abstract level independent of the details of its implementation.

Three widly used abstraction mechanism in software design are functional abstraction, de

1. Data abstraction
2. Control abstraction
3. Functional abstraction

2. Architecture :-

The architecture of the procedural and data elements of a design represents a software solution for the real world problem.

Software architecture implies two characteristics of computer program which are of primary importance.

i) The hierarchal structure of procedural components.
ii) The structure of data.

3. Patterns:

A design pattern describes a design structure that provides a reliable solution to a recurring problem.

The pattern forms an instance of the original design with includes certain specification through which one can determine weather the specification

4. Modularity:

A system consider as modular if it consists of discrete components so that each component can be implemented separately.

A software system can be made modular when each modules supports a well defined abstraction and have a clear interface, by which it can interact with other modules.

5. Information Hiding:

Information hidding plays an important role in designing software. It hides the internal details of the modules and its processing activities.

6. Functional Independence:

It refers to a features of software, where the developers limit the cohesion between various modules of software and make them to function Independently.

7. Refinment:

Stepwise refinment is a top-down approach for decomposing a system from high level specifications into more elementary levels.

It isolate the design aspects that are not actually dependent. It delays decision related to representation details.

8 Refactoring.

It is a process of improving the existing software without altering its internal mechanism.

Here once a given software is said to be refactored it means that, all the loopholes existing in such data structure

9. Intent:

The behaviour of a requirement model specification when the intent are divided into smaller manageable parts.

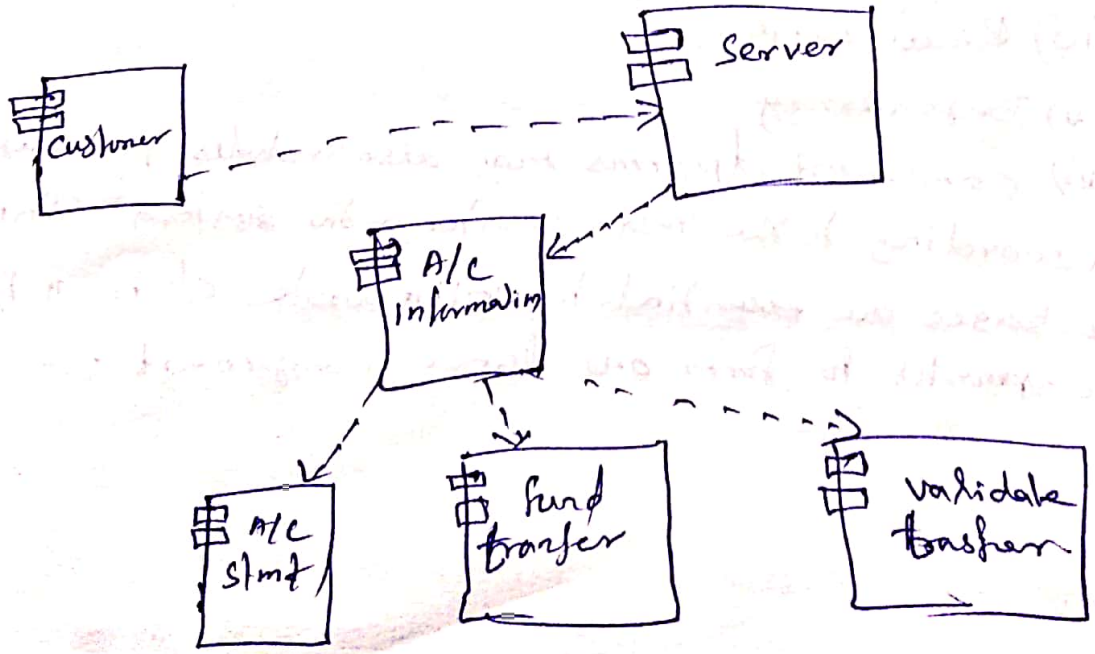
10. Aspect:

During the requirements analysis phase many aspects requirements related to data structure, patterns, requirements and QoS issues encounter.

2. Explain briefly about component level design.

The component is a physically existing part in a system. Exe file, obj file, etc... Can be represented as various components which can exist on a node. They are replaceable and supports openness and adaptability.

It helps in deciding the way in which the source code can be organized or the way the database is to be laid or the manner in which the executable files of a project are to be implemented.



1) The way the source code can be organized
2) The way the database is to be laid
3) The way the executable files of a system are to be implemented.

Component diagram includes,

1. Components.

It refers to the part or elements of the system to be modeled. This can be executable, libraries, tables, files, etc...

2. Interfaces.

It consist of operations that belongs to classes or components residing in the models. It gives the external behaviour of a component to which it is attached.

3. Relationships:

There are four distinct types of relation between the components.

i) Dependency
ii) Association.
iii) Generalization
iv) Realization.

~~v) Dependency~~

vi) Component diagrams may also includes packages, according to the requirements of the system. This packages are essential to collaborate club all the components to form one large component.

2. What is Software architecture? Describe the different Software architectural styles with examples.

The Software architecture transforms the designs of an entire system with an intention of providing suitable structure to various components of that system.

It may sometimes happens that the existing architecture is to be reengineered.

Each architecture style consists of the following.

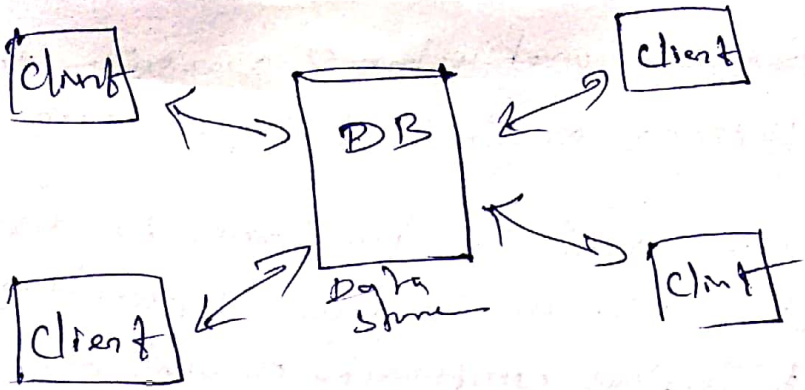
1. Certain number of connectors facilitating coordination Cooperation as well as communication among various components forming a system
2. Certain number of components capable of providing definite functionality
3. Semantic models.
4. Constraints which refers to integration of a system.

The categories of Software architecture are as follows,

1. Data Centered Architecture.

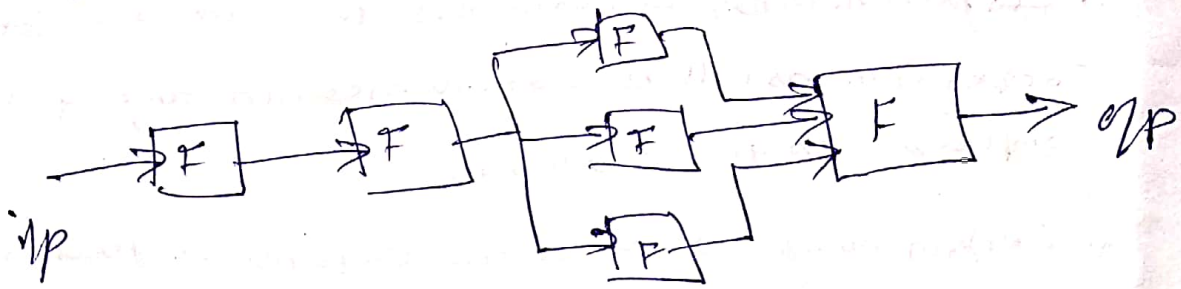
This architecture stores the essential data of the centre of architecture

* The client software are authorized to access this data.
* These client Software can easily manipulate the centred data.
* All the client Software are authorized to access several of their processes independently.



2. Data flow Architecture

When the given data is to be transformed into certain output by applying series of components.

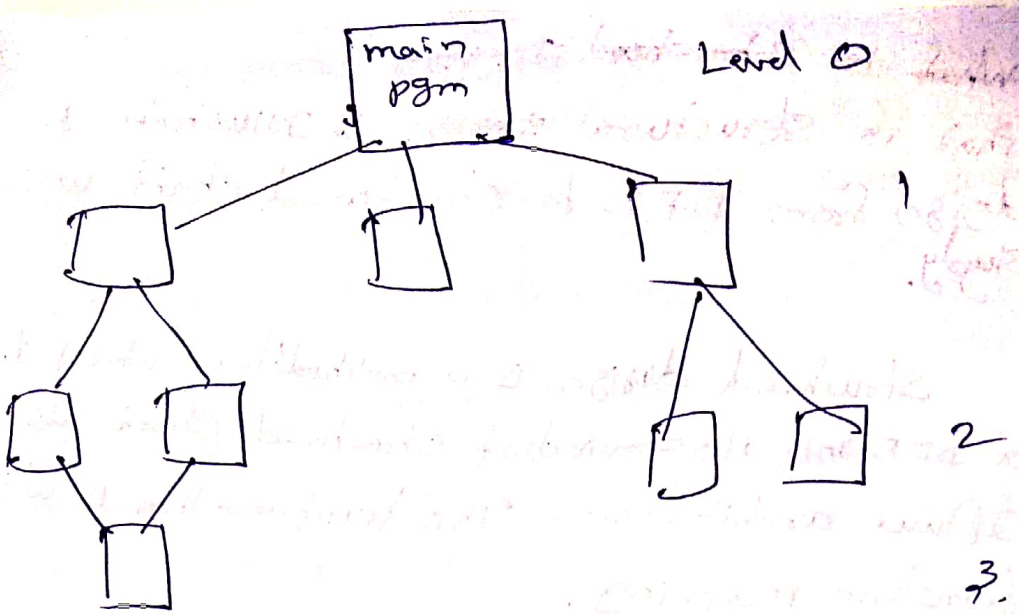


3. The call and return architecture

This architecture has got two sub architecture refferred as main pgm / sub pgm or the requirements.

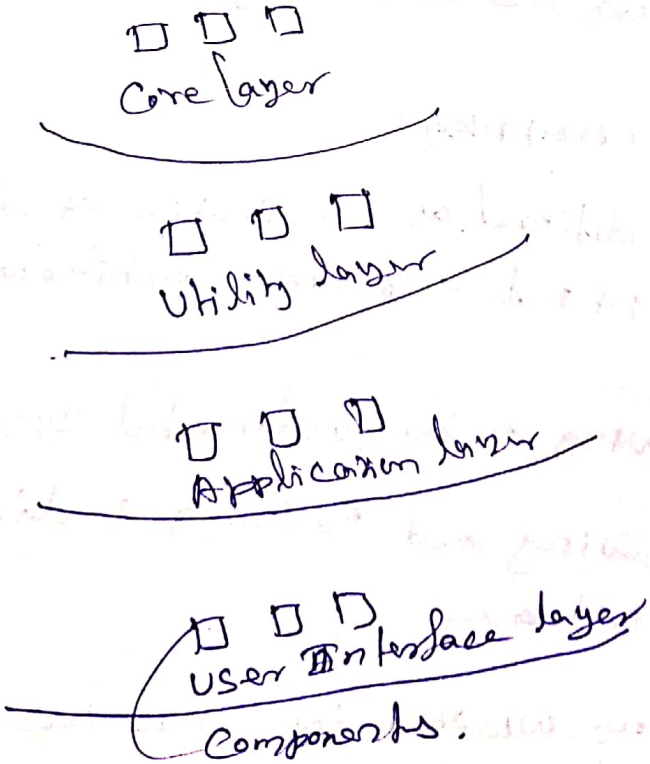
The main program and sub program decomposes of number of constituent program components.

The levels consists of main program, the level 1 consists of controlled sub program and level 2 and 3 consists of application subprogram.



4. Layered Architecture :-

The layered architecture basically has four layers. The rectangular small boxes represented in each layer are the components associated with that layer. Each of these layers defines a definite set of operations.



~~4. What is Structural Design~~

4. What is Structural Design ? Illustrate the Structured design from DFD to structured chart with a case study.

Structured design is a methodology which transforms a DFD into its equivalent structured chart also known as Software architecture. This transformation is also known as transform mapping.

1. Establishing the type of data flow
2. Representing the flow boundries
3. Mapping DFD to the structure of program.
4. Defining the hierarchy of control flow.
5. Refining the derived structure by making use of design heuristics and design measures.
6. Refining and elaborating the architectural description.

Transform mapping :

It defined as a collection of design steps that maps a DFD to a specific architectural style.

1) Reviewing of the Fundamental System model.
2) Reviewing and Refining of data flow diagrams for software
3) Finding whether the DFD has transaction or Transform Characteristics.

3.i) Choose a global flow characteristics depending on the current nature of the DFD

3.ii) Separate the local regions of the transform.

4) Separating the transform center

5) Conducting first level factoring.

i) Top level component, decision making

ii) Middle level component, handle small amount of work

iii) Low level component, handle I/P, O/P, and computation work

The controller coordinates the following subordinate functions,

1. The sensor input controller which process the incoming information.
2. The alarm conditions controller which handles transform flow, manages all the operations.
3. The alarm output controller which process the outgoing information coordinates the generation of output information.

6) Conducting second level factoring.

7) Refining first iteration Architecture through design Heuristics for Improved Software quality

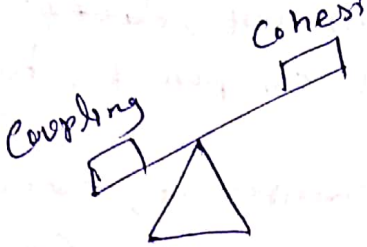
- Effective fan factoring
- High Cohesion
- low coupling

The main reason of refactoring is to generate a structure that can be understood easily.

Q. Explain in detail types of cohesion and coupling with example

Cohesion:

A measure which identifies the dependency of elements with in a module. Thus, Coupling gives the intra module connections. Intra module connection means, interaction needed between elements of module.



This does not mean that Coupling becomes 'zero' or 'will' for the highest cohesion.

Level of cohesion,

The amount of co-cohesive between elements gives rise to the levels of cohesion.

* Functional Cohesive
* Sequentially cohesive
* Communicationally Cohesive
* Procedurally Cohesive
* Temporally Cohesive
* Logically cohesive
* Coincidentally Cohesive

Coupling:

A qualitative measures that identifies the dependency of one module on another module. A module is said to be tightly coupled. If they

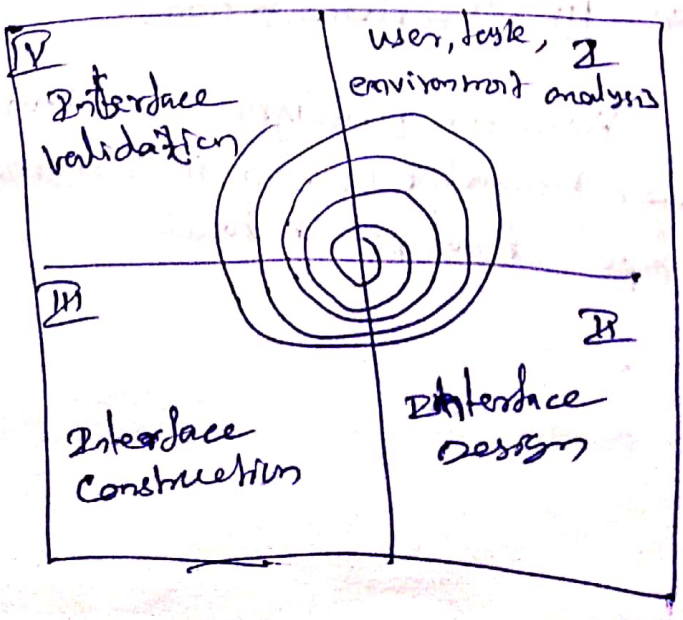
are strongly connected and are said to be loosely coupled, If they are loosely connected. on the other hand. If the modules are independent of one another then no coupling exist between these modules.

There are

* Content Coupling
* Common Coupling
* Control Coupling
* Type use Coupling
* Inclusion or Import coupling
* External coupling.

6. Explain user Interface design process in detail.

The design process is a sequential procedure which is divided into four phases. These phases are represented using a spiral model and each phase can be repeated more than once before the completion of entire design process.



Phase - I :-

User, Task and Environment Analysis.

The user requirements are acquired depending on the user type and its category.

i) Does the system hardware is exposed to certain calamities like light, noise, heat etc - -
ii) The location of interface
iii) Is the interface surrounded by other co-systems which can be operated by the user.

Phase - II Interface Design.

Interface Design deals with the designing of the system. Here, different objects, which are active part of the system are determined and implemented.

i) Defining interface objects and operations by using the information collected during interface analysis
ii) modeling the behaviour of the user action which may change the user interface
iii) prototype each interface as it will be given to the end user.

Phase - III Implementation.

Here the prototype is initially constructed, which is later brought up into its originality by using the interface development tools.

phase - V Interface validation:

It is the last black testing phase where the correctness of the system is tested against the user requirements.

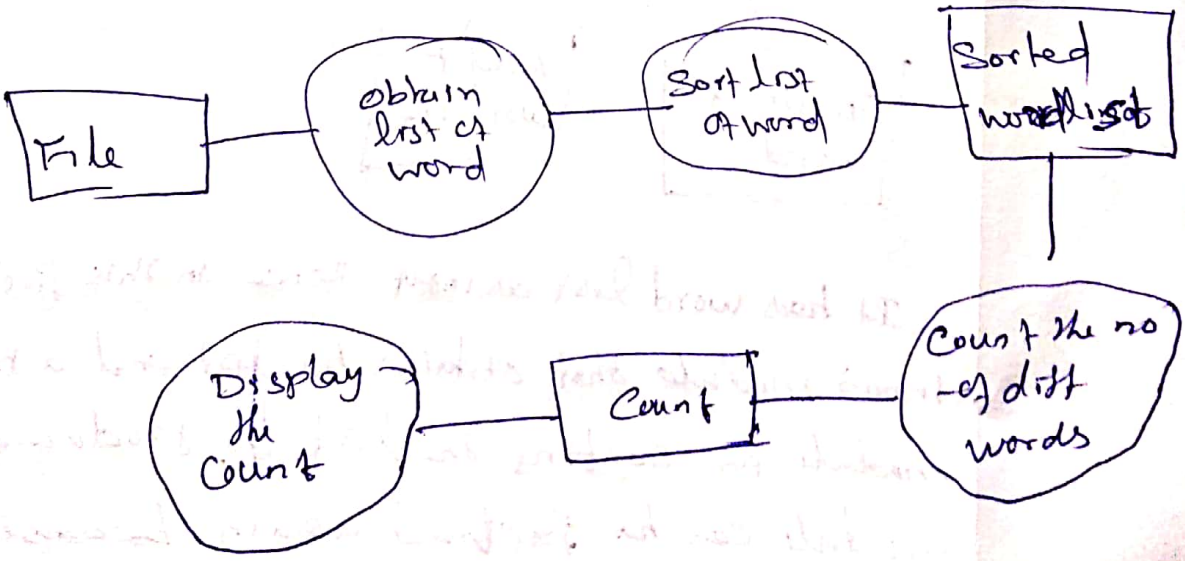
i) Is the system completely satisfying the user.

ii) Does the system possesses easy to be learned and adopted features.

iii) Is the system satisfying the variations and the details specified by the user in user requirements gathering phase etc..

7- Consider the problem of determining the number of different words in an input file. Carryout structured design by performing transform and transaction analysis.

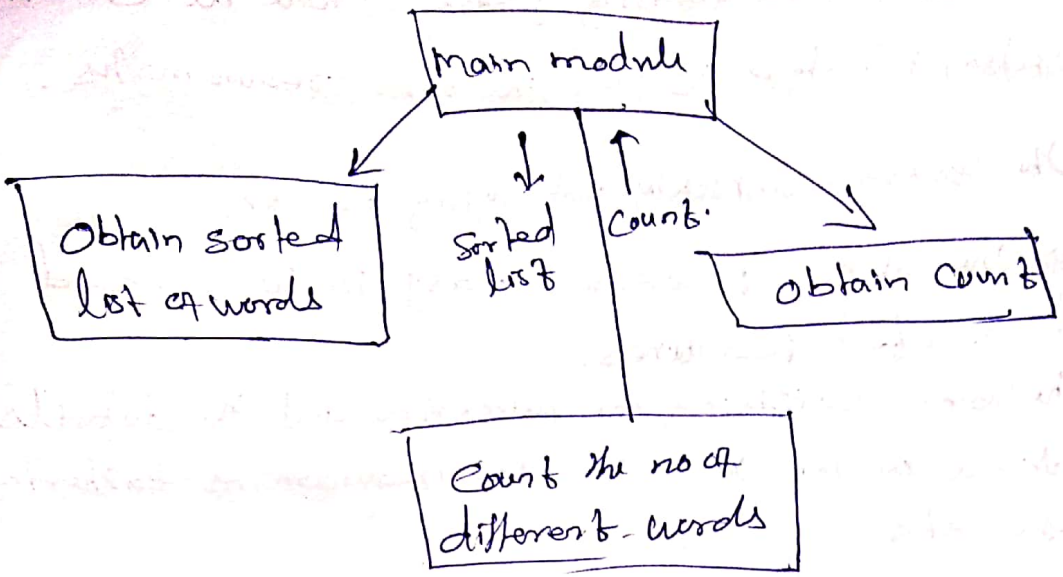
Given problem determining the number of distinct words in an input file. The first step to performing a structured design methodology to any system is to design a Level 0 DFD.



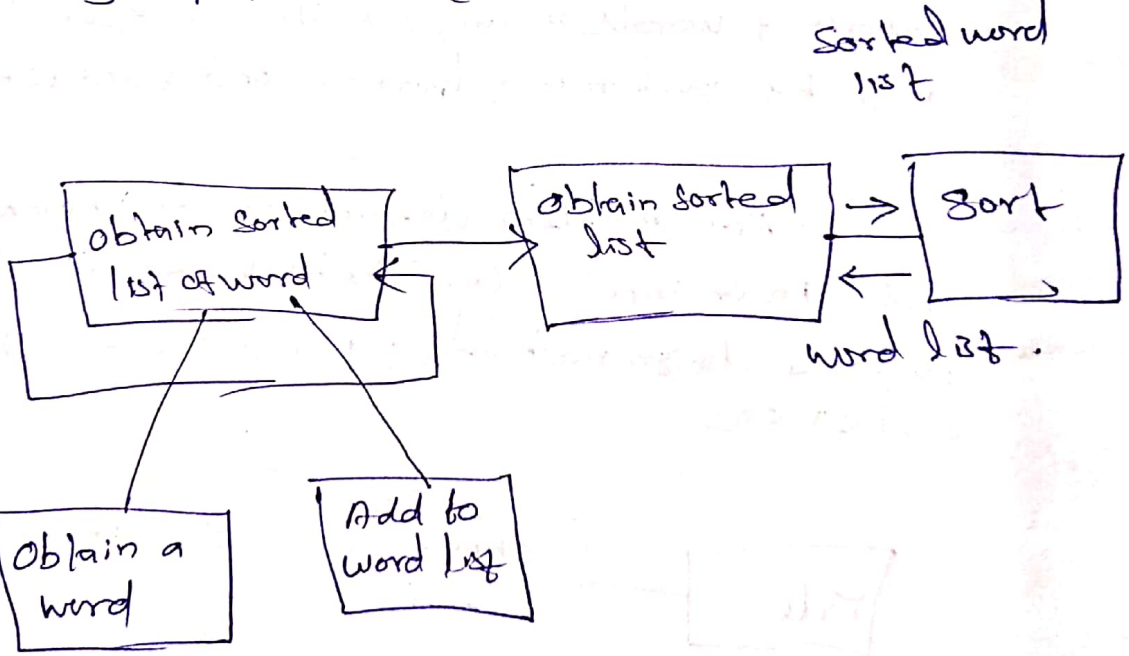
The data flow diagram for problem of word Counting.

(r)

First level Factoring:

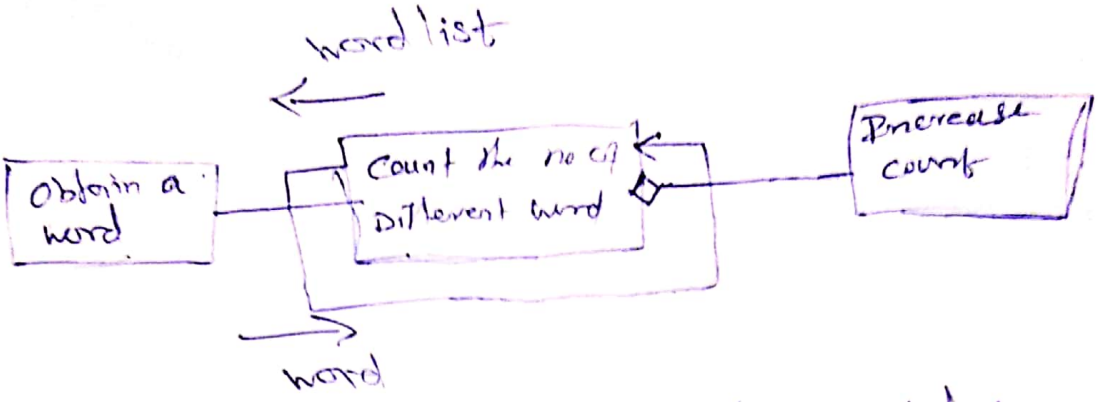


Factoring input module (obtain sorted list of words).



It has word list as input. Hence in this factoring, an input module that obtains the list and a transform module for sorting the list is produced. This input module can be factored again because it has two functionalities.

Since there is only one transform after the most abstract output, there is no need to factoring it. The looping arrow in the above figure represents the iteration. The factoring of the central transform may



It determins the number of words by obtaining the word again and again and chuw whether it is similar to the previous word.