Abstraction in oo design(rough idea)

Through the process of abstraction, a programmer hides all but the relevant data about an object in order to reduce complexity and increase efficiency.

In the process of abstraction, the programmer tries to ensure that the entity is named in a manner that will make sense and that it will have all the relevant aspects included and none of the extraneous ones.

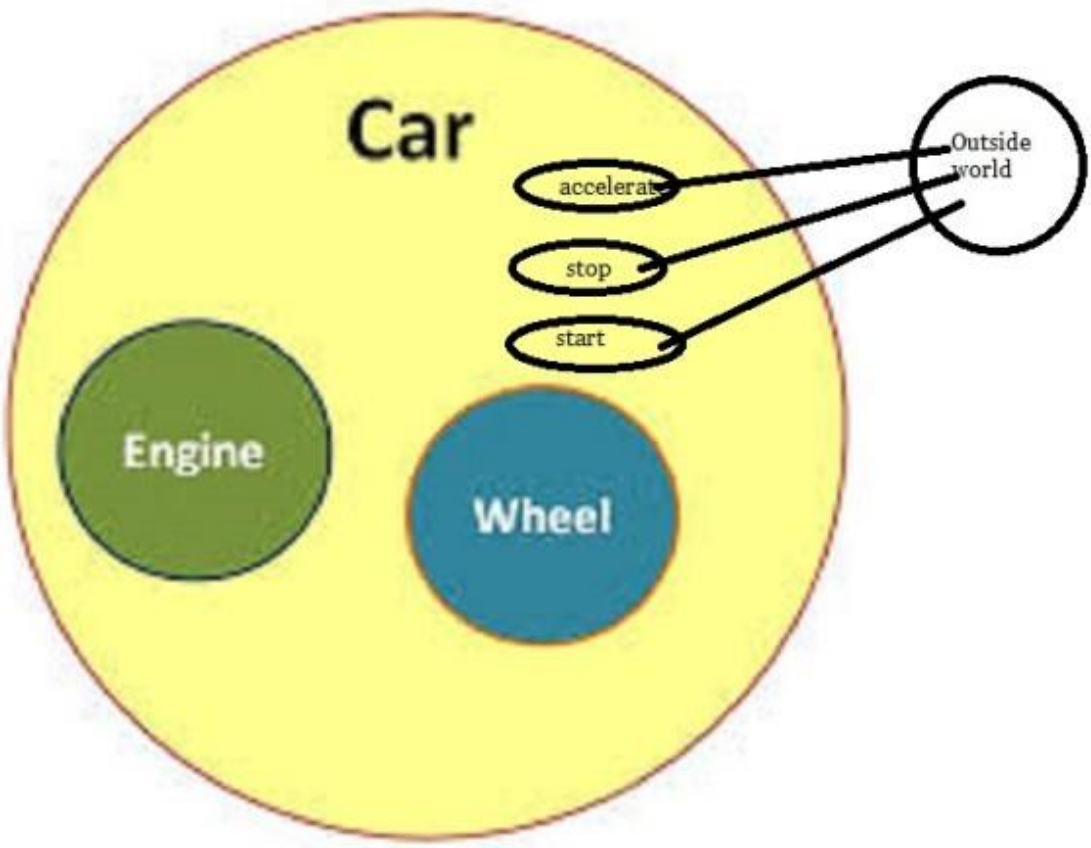
Abstraction is that simply separates implementation details and interface details.

> The interface view is the face seen by other programmers. It describes what a software component can perform.
>
> The implementation view,on the other hand, is the face seen by the programmer working on a particular component.It describes how a component goes about completing a task.
>
> (Can also mention Parna's principle too)

Scenario explained:

It is possible for one programmer to know how to use a component developed by another programmer, without needing to know how the component is implemented. For example, suppose each of the six components in the IIKH is assigned to a different programmer. The programmer developing the Meal component needs to be able to allow the IIKH user to browse the database of recipes, and select a single recipe for inclusion in the meal. To do this, the Meal component can simply invoke the browse behavior associated with the Recipe Database component, which is defined so as to return an individual Recipe. This description is valid regardless of the particular implementation used by the Recipe Database component to perform the actual browsing action.

The purposeful omission of implementation details behind a simple interface is known as **information hiding**. We say the component encapsulates the behavior, showing only how the component can be used, not the detailed actions it performs.



Abstraction mechanism.

1. Procedures and Functions

Libraries of procedures and functions (such as mathematical or input/output libraries) provided the first hints of *information hiding*.

They permit the programmer to think about operations in high level terms, concentrating on *what* is being done, not *how* it is being performed.

But they are not an entirely effective mechanism of information hiding.

2.Block Scoping

The block scoping mechanism of Algol and its successors, such as Pascal, o OFFERS
es slightly more control over name visibility than does a simple distinction between local and global names.

It also doesn’t solve the problem of information hiding.

3.Modules

Modules basically provide collections of procedures and data with import and export statements.

Solves the problem of encapsulation.

Modules by themselves provide an effective method of information hiding, but do not allow us to perform instantiation, which is the ability to make multiple copies of the data areas.

4.Abstract Data Types

An Abstract Data Type is a programmer-defined data type that can be manipulated in a manner similar to system-provided data types.

Must have the ability to instantiate many different copies of the data type.

Data type can be implemented using provided operations, without knowledge of internal representation.

5.Objects - ADT's with Message Passing

Characteristics of Objects

Encapsulation -- similar to modules

Instantiation -- similar to ADT's

Messages -- dynamic binding of procedure names to behavior

Classes -- a way of organization that permits sharing and reuse

Polymorphism -- A new form of software reuse using dynamic binding