UNIT -3

OBJECT ORIENTED DESIGN

Objects, Object classes:

* An object is an entity that has a state and a defined set of operations which operate on that state

* The state is represented as a set of object attributes

* The operations associated with the object provide services to other objects, which request these services when some computation is required

Object Classes:

* Objects are created according to some object class definition

* An object class definition serves as a template for objects

* It includes declarations of all the attributes and services which should be associated with an object of that class

* Object communicate by requesting services from other objects, and if necessary exchange information required for service provision

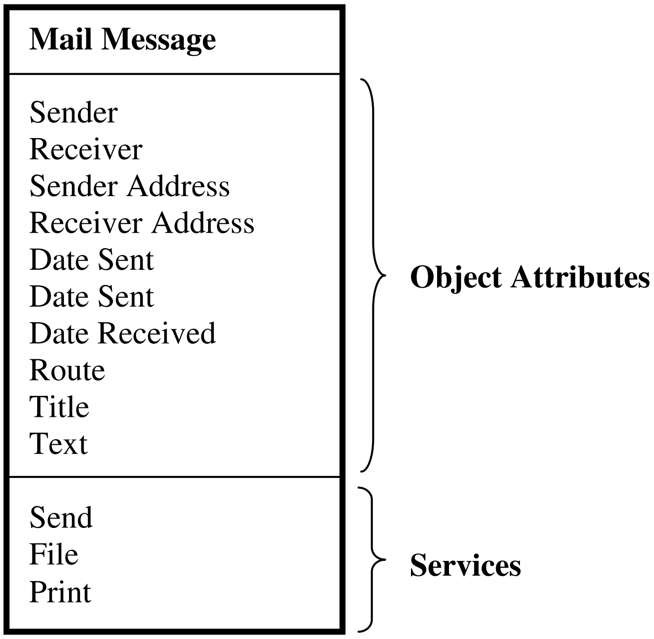
* In some distributed systems, object communications are implemented directly as text messages which are exchanged by objects

* Object communication is implemented as procedure (Or) Function calls

* An object class is represented as a named round edged rectangle with two sections

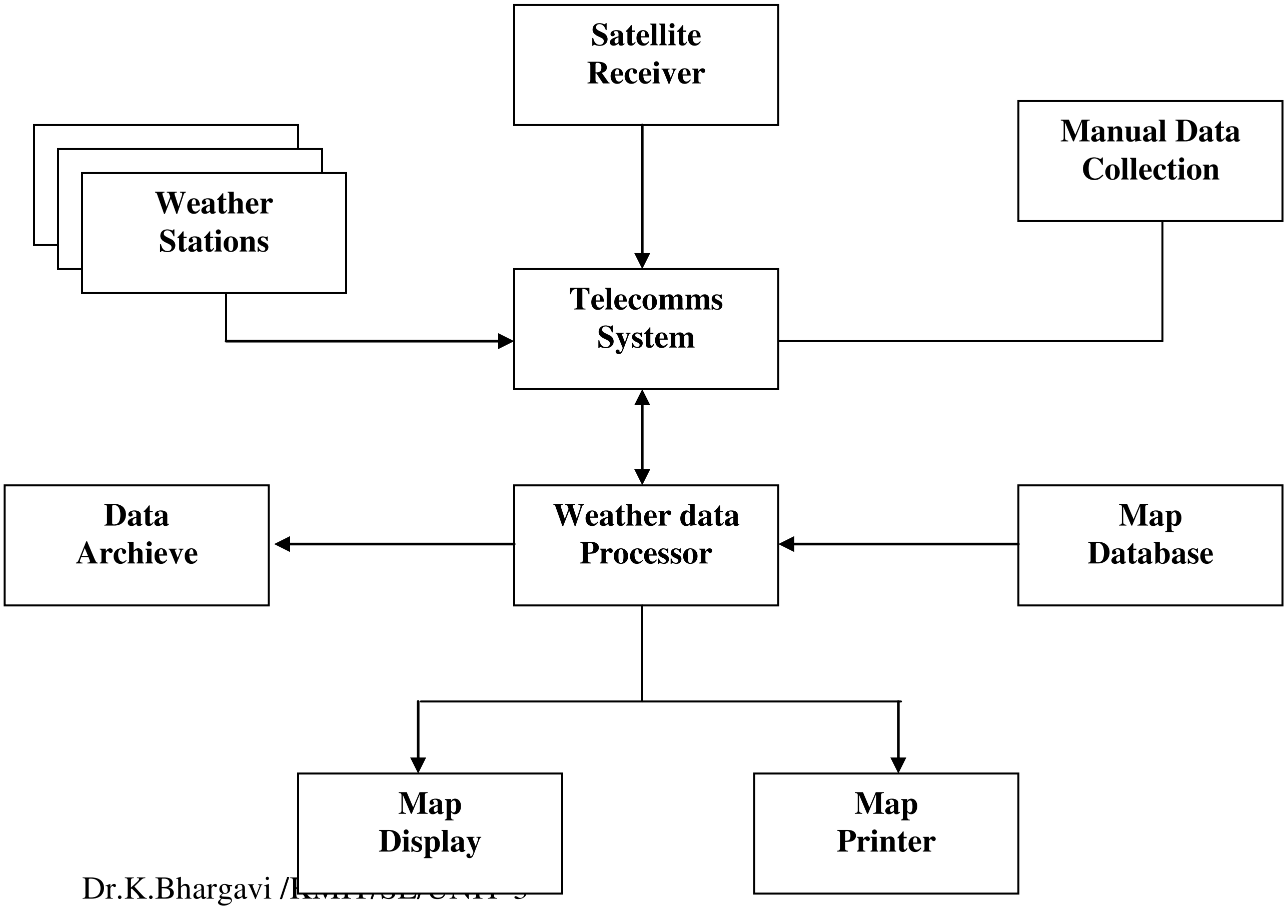
* The object attributes are listed at the top section

* The services provided by the object are set out in the bottom section



5.2 Object Oriented Design Process:

Example:



* It is a means of designing with information hiding

* Information hiding allow the information representation to be changed without other extensive system modifications

* The main problem in object-oriented design is identifying the objects that make up the system, their attributes and associated operations

* **Weather Stations** – Which collects information and communicates it for processing

* **Map database** – Which provides templates of maps for weather data to be added

* **Map** – Which is displayed and printed

* **Weather Data** – This is used to produce the Map, objects, attributes and operations of a weather station

(1) Objects:

=> Air

=> Ground Thermometer

=> Anemometer

=> Wind Vane

=> Barometer and Rain Gauge

(2) Operations:

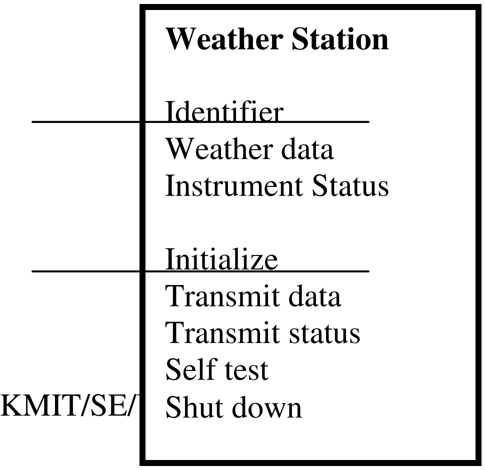
=> Collect Data

=> Perform Data Processing

=> Transmit data

(3) Attributes:

=> Summarized Data



Weather Station

Identifier
Weather data
Instrument Status

Initialize
Transmit data
Transmit status
Self test
Shut down

(4) Data Collected by Weather Station:

=> Air Temperature

=> Ground Temperature

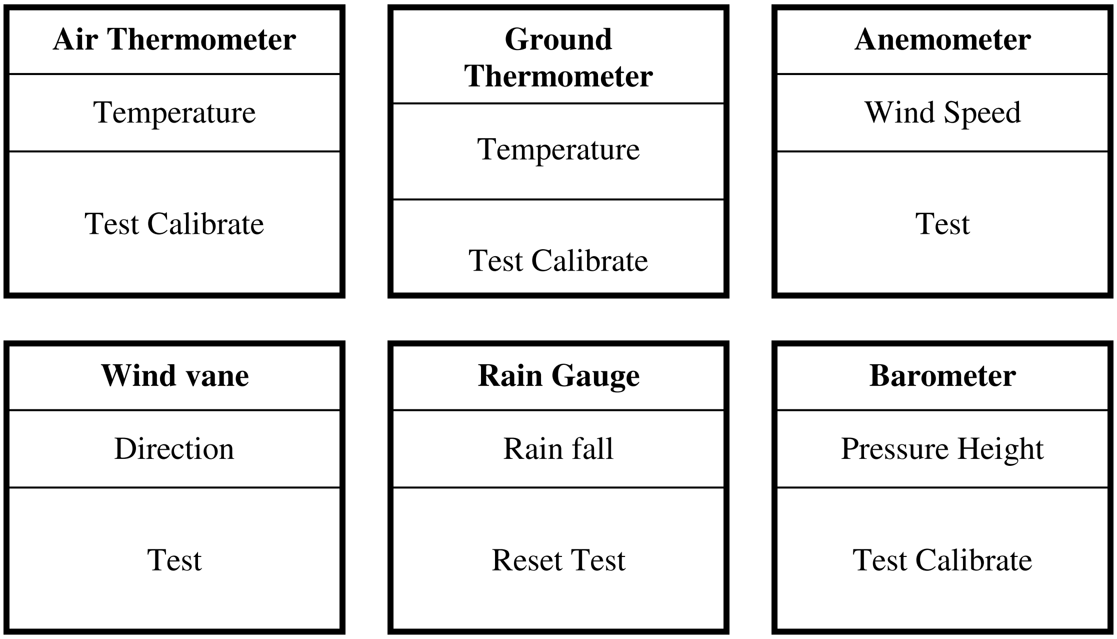
=> Wind Speed

=> Pressure

=> Rainfall

=> Wind Direction

Objects in weather System:



Air Thermometer
Temperature
Test Calibrate

Ground Thermometer
Temperature
Test Calibrate

Anemometer
Wind Speed
Test

Wind vane
Direction
Test

Rain Gauge
Rain fall
Reset Test

Barometer
Pressure Height
Test Calibrate

Object Evolution:

* An important advantage of an object oriented approach to design is that it simplifies the problem of making changes to the design

* The reason for this is that object state representation does not influence the design

* Changing the internal details of an object is unlikely to affect any other system objects, because objects are loosely coupled

* It is usually straight forward to introduce new objects without significant effects on the rest of the systems

PERFORMING USER INTERFACE DESIGN

User Interface Design:

* It creates an effective communication medium between a human and a computer

* Following a set of interface design principles, design identifies interface objects and actions that creates a screen layout, that forms the basis for a user interface prototype

* A software engineer designs the user interface by applying an iterative process that drawn on widely accepted design principles

Steps Followed:

* User interface begins with the identification of user task and environment requirements

* Once user tasks have been identified, user scenarios are created and analyzed to define a set of interface objects and actions

* Create screen layouts that depict graphical design and placement of icons, definition of descriptive screen text, specification and tilting for windows

* Tools are used to prototype and implement the design model

The Golden Rules:

(1) Place the user in control

(2) Reduce the user's memory load

(3) Making the interface Consistent

(1) Place the user in control

* During the requirement gathering session, user was asked about the attributes of the widows oriented graphical interface

* The User wanted a system that reacted to his needs and helped him to get things done

* He wanted to control the computer not have the computer control her

Design Principles – that allow the user to maintain control:

(i) Define interaction modes in a way that does not force a user into unnecessary (Or) Undesired actions

* An interaction mode is the current state of the interface

Example:

* If a spell check is selected in a word processor menu, the software moves to a spell – checking mode

* No need to force the user to remain in spell checking mode, if the user desires to make a small text edit along the way

* The user should be able to enter and exit the mode without any effort

(ii) Provide for flexible interaction:

* Different users have different interaction preferences choice should be provided

Example:

* Software might;

=> allow a user to interact via keyboard commands

=> Mouse Movements

=> a digitizer pen (Or) Voice recognition commands

* But every action is not amenable to every interaction mechanism

* For example

=> The difficult of using keyboard commands (Or) Voice input to draw complex shape

(iii) Allow user interaction to be interruptible and undoable

* When user involved in a sequence of actions, he should be able to interrupt the sequence without losing the work that had been done

* The user should also be able to undo action

(iv) Streamline interaction as skill levels advance and allow the interaction to be customized

* User should find that they perform the same sequence of interactions repeatedly

* Design a "macro" mechanism that enables an advanced user to customize the interface to facilitate the interaction

(v) Hide technical internals from the casual users:

* The user should not aware of the operating system, file management functions and other computing technology

* A user should never be required to type operating systems commands from within application software

(vi) Design for direct interaction with objects that appear on the screen

* The user feels a sense of control, when able to manipulate the objects that are necessary to perform a task

Example:

* An application interface that allows a user to "Stretch" an object [scale it in size] is an implementation of direct manipulation

(2) Reduce the user's memory load:

* Design principles enable an interface to reduce the user's memory load

* If the user is forced to remember more than the interaction with the system will be more error - prone

* The system should remember pertinent information and assist the user during interaction

(i) Reduce demand on short-term memory:

* When user involved in complex tasks the demand on short-term memory can be significant

* The interface should be deigned to reduce the requirements to remember past actions and results

* This can be achieved by providing visual cues that enable a user to recognize past actions rather having to recall them

(ii) Establish Meaningful defaults:

* The initial set of defaults should make sense for average user

* But user should be able to specify individual preferences

* However, reset option should be available enabling the reduction of original default values

(iii) Define Shortcuts that are intuitive:

* When Mnemonics are used to accomplish a system function [e.g. ALT + P to invoke the print function]

* The mnemonics should be tied to the action in a way that is easy to remember

Example:

* First letter of the task to be invoked

(iv) The visual layout of the interface should be based on a real world metaphor

* For example a bill payment system should use a checkbook and check register metaphor to guide the user through the bill paying process

* This enables the rely on well – understood visual cues, rather than memorizing an interaction sequence

(v) Disclose information in a progressive fashion:

* The interface should be organized hierarchically

* i.e. The information about the tasks an object (Or) some behavior should be presented first at high level of abstraction

* More detail should be presented after the user indicate s with a mouse pick

Example:

* In word processing applications the underline function, here every underlining capability is not listed

* The user must pick underlining and then all underlining options [e.g. Single Underline, Double Underline, Dashed Underline] are presented

(3) Make the interface consistent:

* The interface should present and acquire information in a consistent fashion. i.e

=> all visual information is organized according to a design standard, that is maintained throughout all screen displays

=> Input mechanisms are constrained to a limited set, that is used consistently throughout the application

=> Mechanisms for navigating from task to task are consistently defined and implemented

Design Principles that make interface consistent:

(a) Allow the user to put the current task, into a meaningful context:

* Many interface implement complex layers of interactions with dozens of screen images

* It is important to provide indicators

Example:

=> Window Tiles

=> Graphical Icons

=> Consistent Color coding that enables user to know the context of the work

* In addition the user should be able to determine,

=> Where he has come from what alternatives exist for a transition to a new task

(b) Maintain Consistency across a family of applications:

* A set of applications should all implements the same design rules to maintain consistency

(c) If past interactive models have created, user expectations do not make changes, unless there is a compelling reason to do so

* Once a particular interactive sequence has become a standard, the user expects this in every application

* A change will cause confusion

Example:

* The use of **CLT + S** to save a file

* But invoke **ALT + S** to scaling results in confusion

User Interface Analysis and Design:

(1) Interface Analysis and Design models:

* When a user interface is to be analyzed and designed four different modules come into play

(i) User Model:

* A software engineer establishes a user model

* It establishes the profile of end users of the system

* The profiles may be the users age, education, motivation etc..

The Users can be categorized as

(i) Novices:

* No synthetic knowledge of the system

* Little semantic knowledge of the application (Or) Computer usage in general

(ii) Knowledgeable, intermittent users:

=> Reasonable semantic knowledge of the application

=> But relatively low recall of syntactic information necessary to use interface

(iii) Knowledgeable Frequent Users:

=> Good Semantic and Syntactic knowledge that often leads to "Power – Users - Syndrome"

=> i.e. Individual who look for shortcuts and abbreviated modes of interaction

(2) Design model:

* It incorporates data, architectural, interface and procedural representations of the software

* The requirements specification may establish certain constraints that help define the user of the system

(3) Mental Model:

* It is the image of the system that end users carry in their heads

(4) Implementation Model:

* It must accurately reflect syntactic and semantic information about the interface

* When the implementation model and users mental model are coincident,

=> Users feel comfortable with the software and use it effectively

The Process:

* The user interface analysis and design process encompasses four distinct framework activities

(1) User tasks, environmental analysis and modeling

(2) Interface design

(3) Interface Construction [Implementation]

(4) Interface validation

* **Interfaces analysis focuses on**

=> the profile of the users, who will interact with the system

=> Skill level

=> Business understanding

=> General receptiveness to the new system are recorded

=> Different user categories are defined

* The goal of interface design is to be define a set of interface objects and actions [and their screen representation] that enable a user to perform all defined tasks in a manner that meets every usability goal

* The construction activity normally begins with the creation of prototype

* To complete the construction of the interface development tools may be used

* **The validation focuses on**

=> the ability of the interface to implement every user task correctly and to achieve all general user requirements

=> the degree to which the interface is easy to use and easy to learn

=> the users acceptance of the interface

Interface Analysis:

* A key tenant of all software engineering process models is "Better Understand the problem, before attempt to design a solution"

* In user interface design understanding the problem means, understanding

=> The people [End – Users], who interact with the system

=> Tasks that end – Users must perform to do their work

=> The content that is presented as part of interface

=> The environment in which these tasks will be conducted

(1) User Analysis:

* As we noted that each user has a mental image of the software that may different from mental image developed by other users

* The user's mental image may be different from software engineer's design model

* The only way that a designer can get the mental image is by accomplish this

(i) User Interviews:

* The representatives from the software team meet end users to better understand their needs, motivations, work culture

(ii) Sale Input:

* Sales people meet customers and users on a regular basis and gather information that will help software team to categorize users and better under stand their requirements

(iii) Marketing Inputs:

* Market analysis can be invaluable in the definition of market segments, while providing an understanding of how each segment might use the software in different ways

(iv) Support Input:

* Support Staff talk with users on a daily basis,

* Making them the most likely source of information on

=> What works and what doesn't

=> What users like and what they dislike

=> What features generates questions

=> What features are ease to use

(2) Task Analysis and Modeling:

* The goal of task analysis is to answer the following questions:

=> What work will the user perform in specific circumstances?

=> What tasks and subtasks will be performed as the user does the work?

=> What specific problem domain object will the user manipulate as work is performed?

=> What is the sequence of work tasks the work flow?

=> What is the hierarchy of tasks?

(i) Task Elaboration:

* It is also called as functional decomposition (Or) stepwise refinement

* It is a mechanism for refining the processing tasks that are required for software to accomplish some desired function

(ii) Object Elaboration:

* Here the software engineer examines the Use case and other information obtained from the user and extracts the physical objects that are used by the interior designer

* The objects are categorized into classes and attributes of each class are defined

(iii) Work Flow analysis:

* This technique allow a software engineer to understand

=> how work process is completed when several people [and roles] are involved

(iv) Hierarchical Representation:

* Once work flow has been established a task hierarchy can be defined for each user type

* This hierarchy is derived by a stepwise elaboration of each task, identified for the user

(3) Analysis of the Display Content:

* During this analysis step, the format and aesthetics of the content are considered

* The question that are asked here are

=> Are different types of data assigned to consistent geographic locations on the screen [e.g. photos always appear on upper right hand corner?]

=> Can the user customize the screen location for content?

=> Is proper on screen identification assigned to all content?

=> how will error messages and warnings be presented to the user?

=> how will color be used to enhance understanding?

(4) Analysis of the work environment?

* In some applications the user interface for a computer based system is placed in a “User – Friendly location”

Example:

* Proper Lightening, good Display height, easy keyboard access