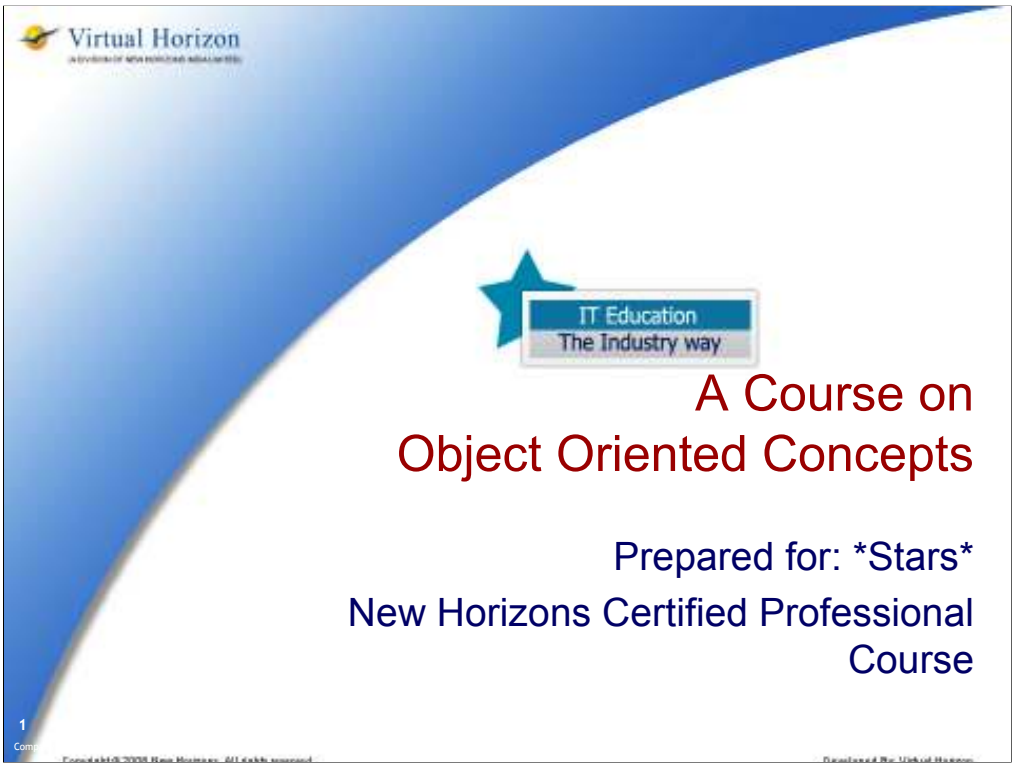
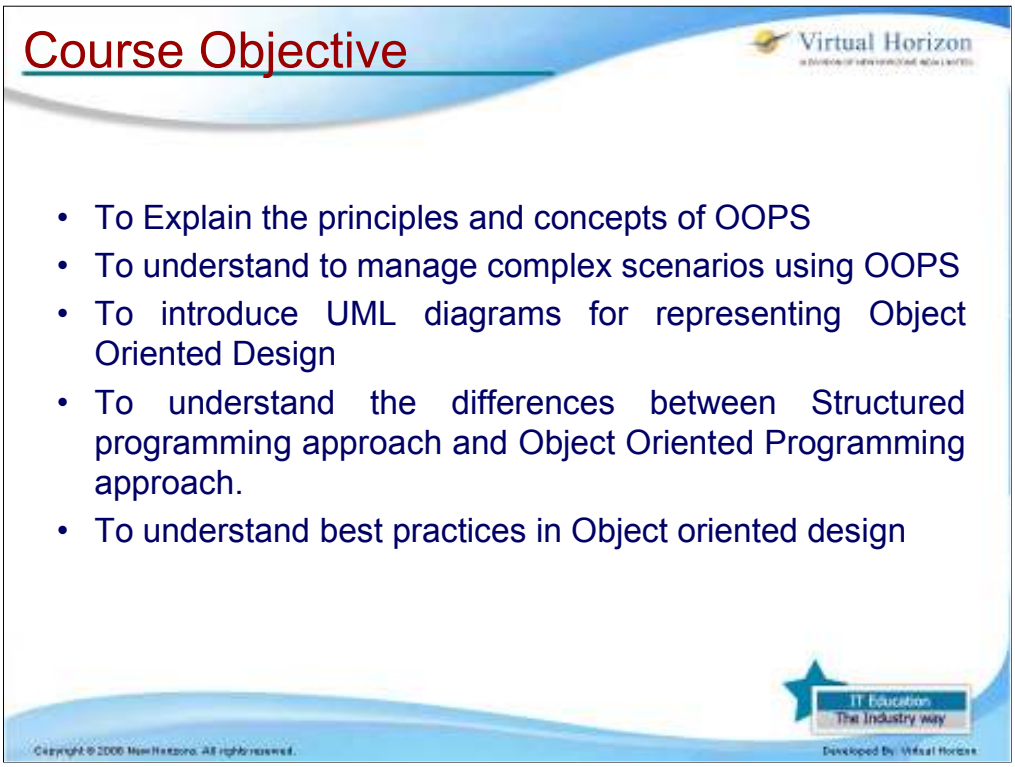

Virtual Horizon
IT Education
The Industry way
A Course on
Object Oriented Concepts
Prepared for: *Stars*
New Horizons Certified Professional
Course
1

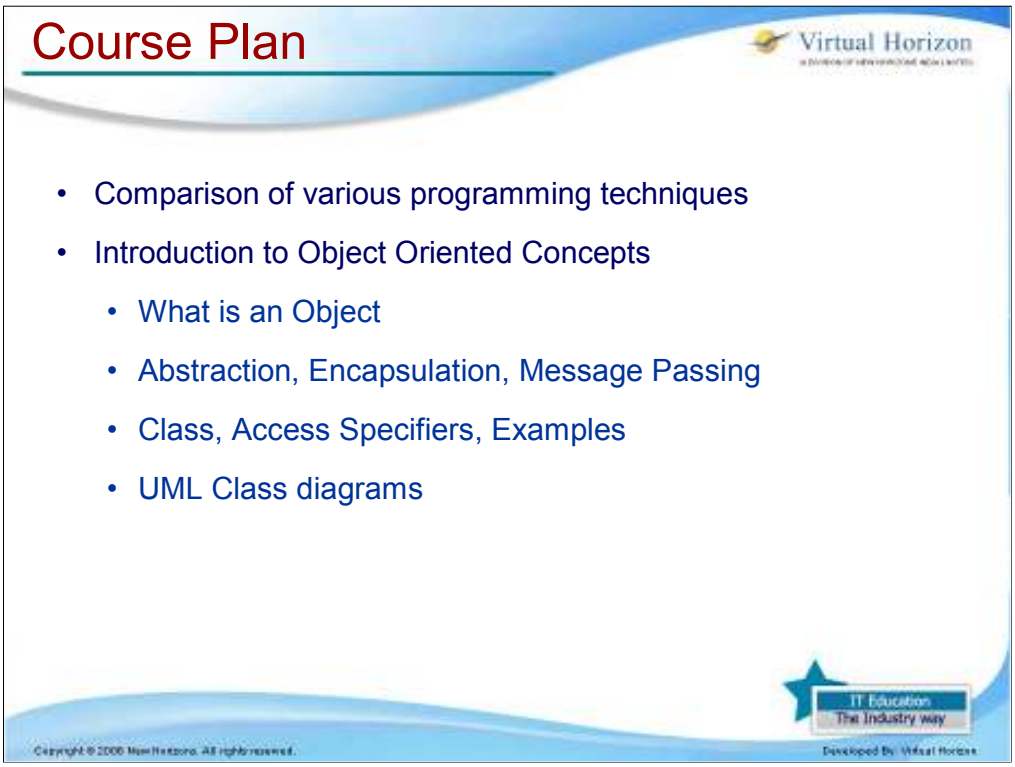


At the end of this course, you will

- understand the inherent complexity involved in software systems and will gain the knowledge of handling such complex software systems

- be able to differentiate between the two programming styles – Structured Programming and Object Oriented Programming

- learn the different features of Object Oriented Technology



Manage Software Complexity

- Different approaches for solving a problem which is complex in nature

Why Object Orientation to solve a complex problem ?

Differentiate between Structured programming and Object Oriented Programming approach

To discuss the different features of Object Oriented Technology

Course Plan

Virtual Horizon

- Advanced Object Oriented Concepts
  - Relationships
  - Inheritance
  - Abstract Classes
  - Polymorphism
- Object Oriented Design Methodology
- Trends in OO Technology
- Case Study and Solution

IT Education
The Industry way

Developed By: Virtual Horizon

William James' definition in the slide above points at handling complexity. We need to use a technique that will reduce the amount of facts we have to deal with simultaneously. Comprehension is not automatic. The time we need to comprehend something is inversely proportional to the number of things we are presented with and to the relevance of those items.

- **Example 1:**
  - Physician find the facts about the patient
    - Normally when a patient goes for a general check-up, the Physician on looking at the previous history of the patient, will understand the situation much faster than if he has to go for a full investigation.
    - The physician is trying to avoid the irrelevant data items so that he can come to the root cause as soon as possible.
- **Example 2:**
  - Take an entity as ***BOOK***. Let us try to find out the different characteristics of the same entity from the perspective of the viewer.
  - Let us take 2 cases where the same entity BOOK can be viewed differently:
    - Library System
      - In this case we will be focusing on Access Number, Book Name, Author Name
    - Shopkeeper
      - In this case we will be focusing on Item Number, Item Name, Price, Quantity On Hand.

Ways of handling Software Complexity

Virtual Horizon

- **Top Down**

"Divide and Rule"

"Algorithmic Decomposition

- **Bottom Up**

Emphasizing only on required details.

Ignoring unnecessary details

IT Education
The Industry way

Developed By: Virtual Horizon

Programming Techniques

Virtual Horizon

- Unstructured
  - Sequence of instructions, which manipulated global data
  - As size increases, code becomes more complex to maintain
- Procedural Programming
  - Brought in Modularity in coding, enhancing maintainability
  - Common functionalities grouped into separate modules
  - Complexity made manageable by hiding complexity inside functions (Concept of APIs)
  - Introduced the concept of structures (also known as data structures)

IT Education
The Industry way

Developed By: Virtual Horizon

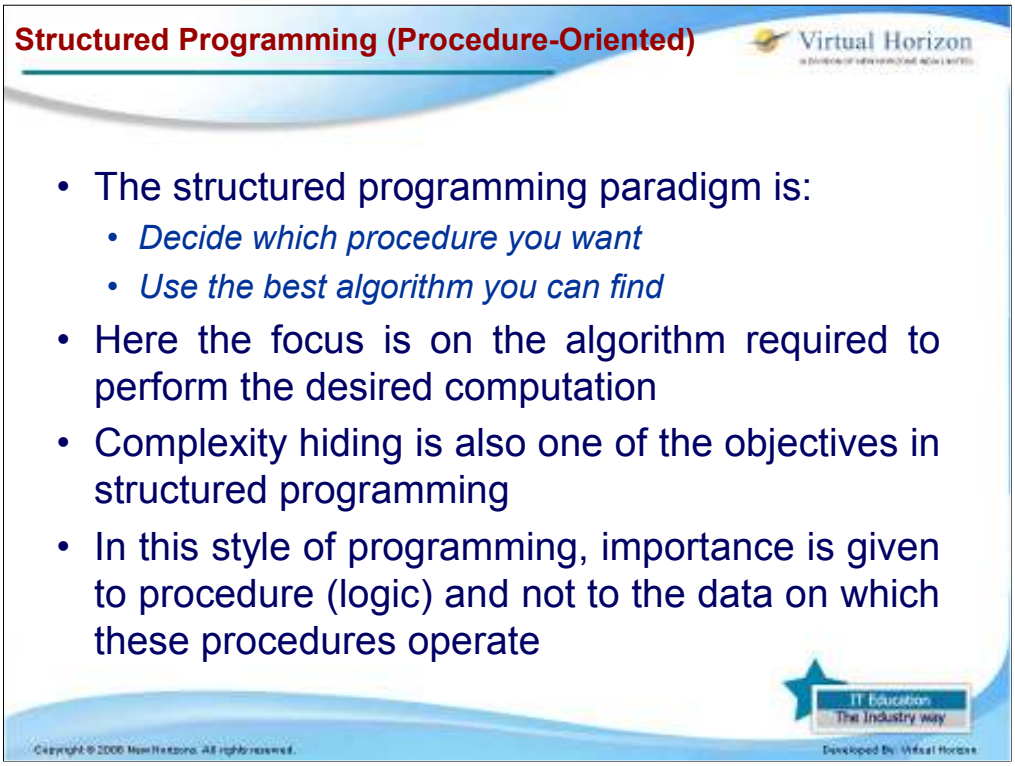
Programming Techniques

Virtual Horizon

- Object Oriented Programming
  - Data structures combined with relevant functions to create reusable objects
  - Focus of this course

IT Education
The Industry way

Developed By: Virtual Horizon



•**Key points in structured programming**

- Focus is on process rather than on data
- It is best suited for a simple solution
- Design approach is "***Top-Down***" where the entire solution is divided into smaller units (Functions and procedures)
- All these smaller units need to work on a data item to return the result
- For this reason the data items used are Global
- Modules are ***tightly coupled*** because of which the same module cannot be reused in another scenario.

- **Coupling :**
  - Coupling refers to the manner and degree of interdependence between software modules. ***(IEEE)***
  - Coupling applies to any relationship between software components.
  - Can be defined as mutual dependence of methods. Low coupling is good for design.

- **What is the problem if the modules are tightly coupled?**
  - If the modules are tightly coupled, it makes the system complex as the module is tough to understand. Also it is hard to change or correct such a module by itself if it is highly interrelated with another module.

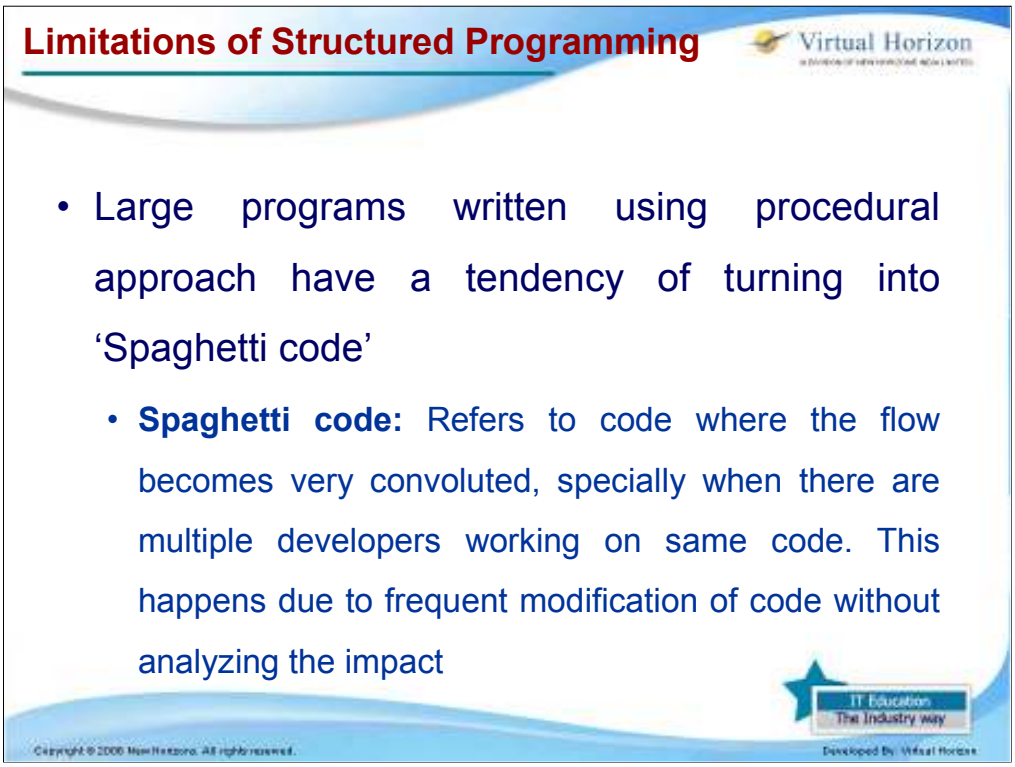
Limitations of Structured Programming

Virtual Horizon

- Modules are tightly coupled
- Scope of reusability is limited
- As the code size grows, maintaining code becomes difficult
- Any major changes required to functionality later may result in a lot of changes in code

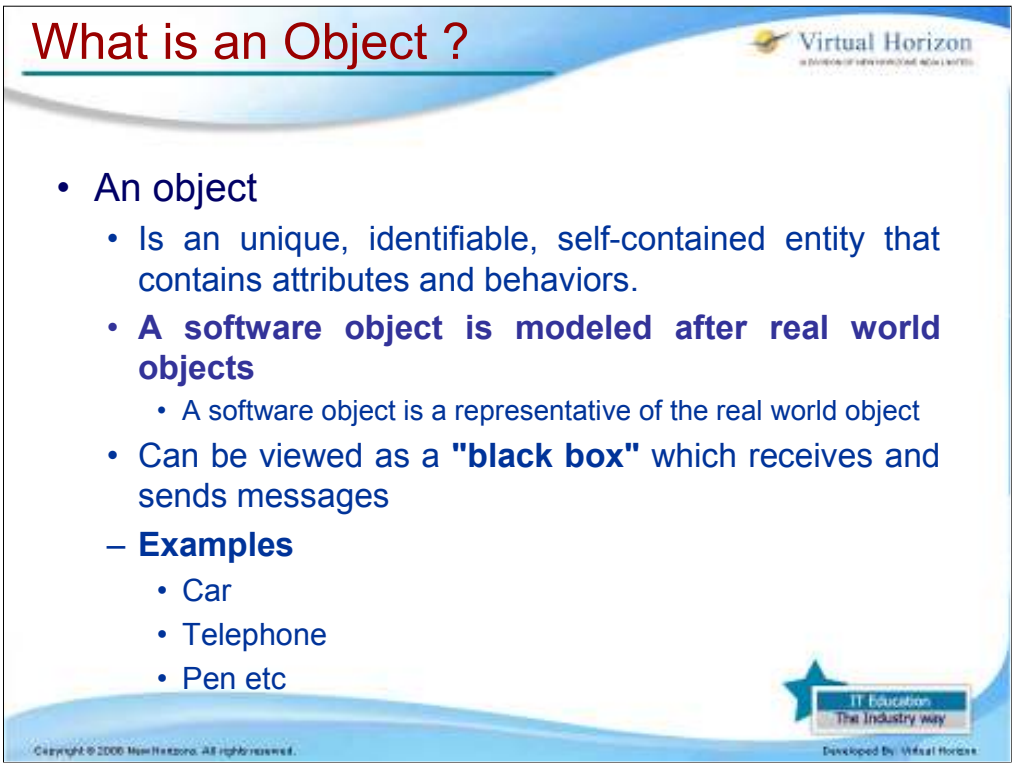
IT Education
The Industry way

Developed By: Virtual Horizon



Structured Programming becomes difficult to manage as the complexity increases the code paths become complex.

- Procedural method for developing information systems
  - works fine for automating routine processes like processing payroll checks.
  - works well in cases where data and applications are separate.
  - works well in cases where data comes in the start of the program, flows through a number of predefined procedures, and exits at the end.
- Structured Programming fails to address the complexities and needs of interactive environments where the flow control is not linear. The program flow dictates the flow of control to the user.
- Object Oriented Technology promises to ease the software complexity by providing a fundamental change to the way information systems are developed.
- Top-Down Approach:
  - Programmer should break larger pieces of code into shorter subroutines that are small enough to be understood easily.



As procedures are used to build structured program, objects are the building blocks of object oriented programs

•A primary rule of object-oriented programming is - as the user of an object, **you would never need to know what is there inside the object!**

•These characteristics represent a pure approach to object-oriented programming:

•Every object contain some member variables and member methods which work upon the member variable.

•A program is collection of objects, which needs to interact among them to do a process. The interaction of objects is also called as message passing.

•Every object has a type as objects instantiated from a class, Here class is considered as a type.

•Objects have ***state, behavior, and identity***

•**Every object:**

Contains data: The data stores information that describes the state of the object.

Has a set of defined behavior. This behavior consist of all the things that the object "knows" how to do. These are the methods present inside the object.

Has an individual identity. Each object is different from the other object even if they are instantiated from the same class.

State and Behavior

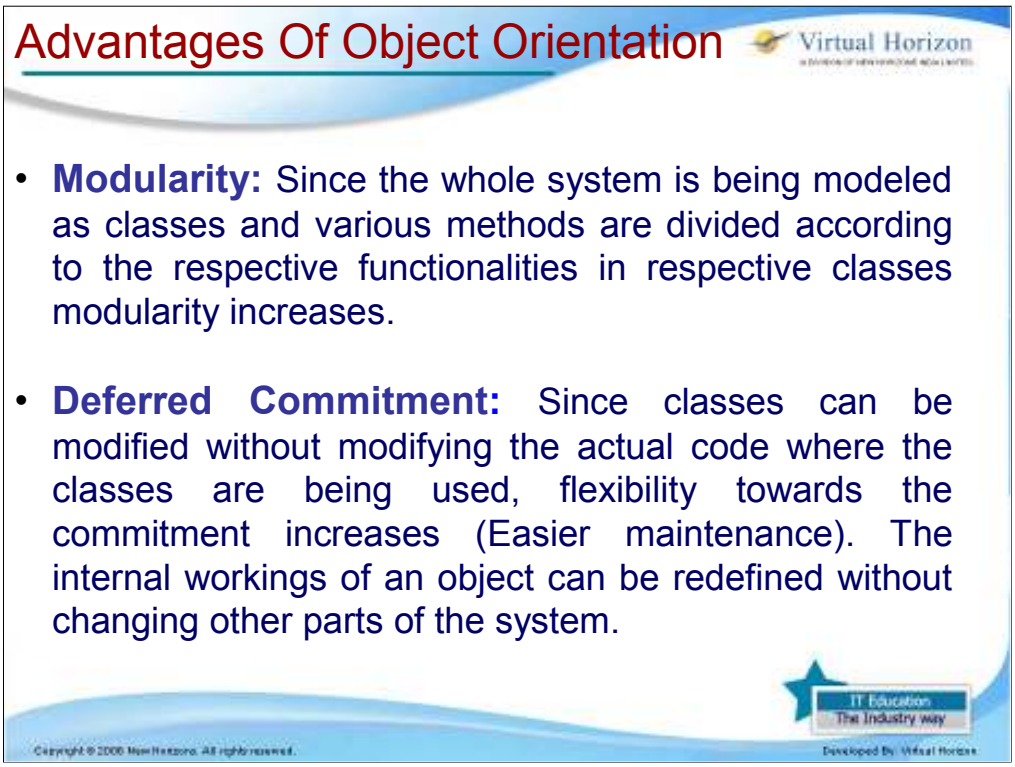
Virtual Horizon

- Example: Car object
  - State
    - Current Speed
    - Current Gear
    - Engine State (Running, Not Running)
  - Behavior **(Acts on the object and changes state)**
    - Slow down
    - Accelerate
    - Stop
    - Switch Off Engine
    - Start Engine

- Example: Dog Object
  - State
    - Color
    - Breed
    - Activity (Barking/Not barking)
    - Tail Activity (Wagging/Not Wagging)
  - Behavior
    - Bark
    - Wag Tail
    - Eat

IT Education The Industry way

Developed By: Virtual Horizon



The main advantages of object orientation are,

▪The main advantage of an OO system is that the class tree is dynamic and can grow.

The main Advantage of Object orientation is the enhancement that can be made without making changes in the previous written code.

I.e we can add new sub-system altogether without affecting already made system in place.

▪You function as a *developer* in an OO system is to foster the growth of the class tree by defining new, more specialized classes to perform the tasks your applications require.

what our role as a developer is that we have to build new classes using previously build classes thus reusing the system as much as possible which saves time, decreases cost and takes less time to build the software.

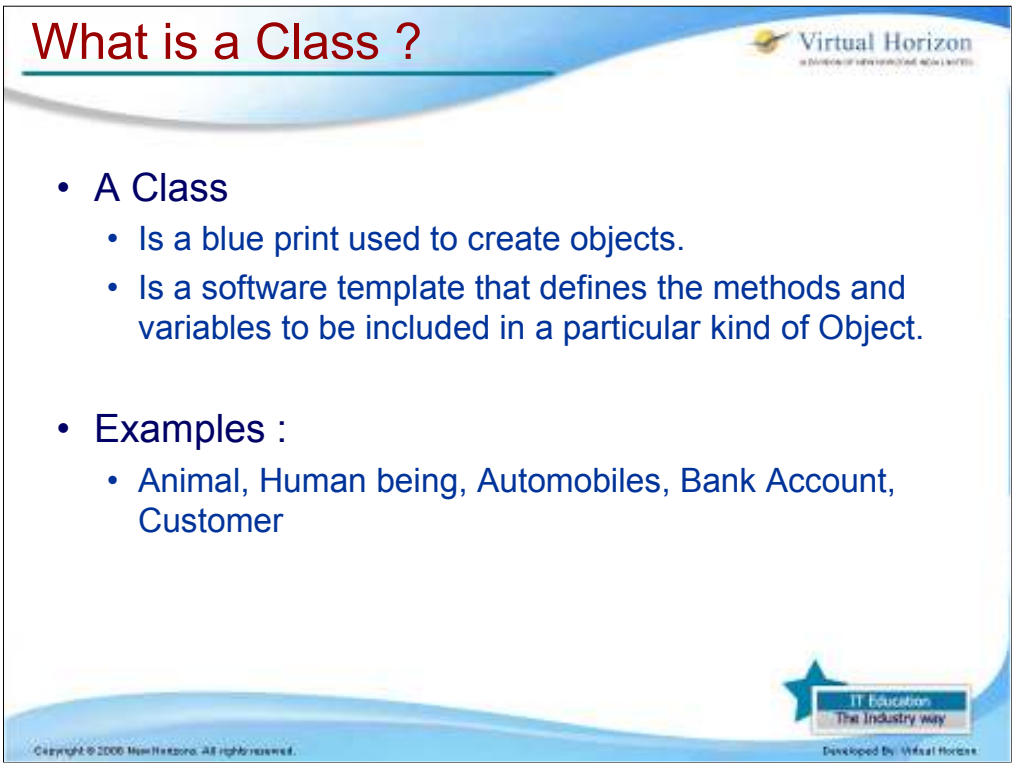
Advantages Of Object Orientation

Virtual Horizon

- **Reusability:** Since defining classes through inheritance helps in reusability thus faster production.
- Higher Quality: Since deriving classes from existing classes which are working successfully.
- Reduced cost: Because of reusability
- Increased Scalability: Easier to develop large systems from well tested smaller systems.

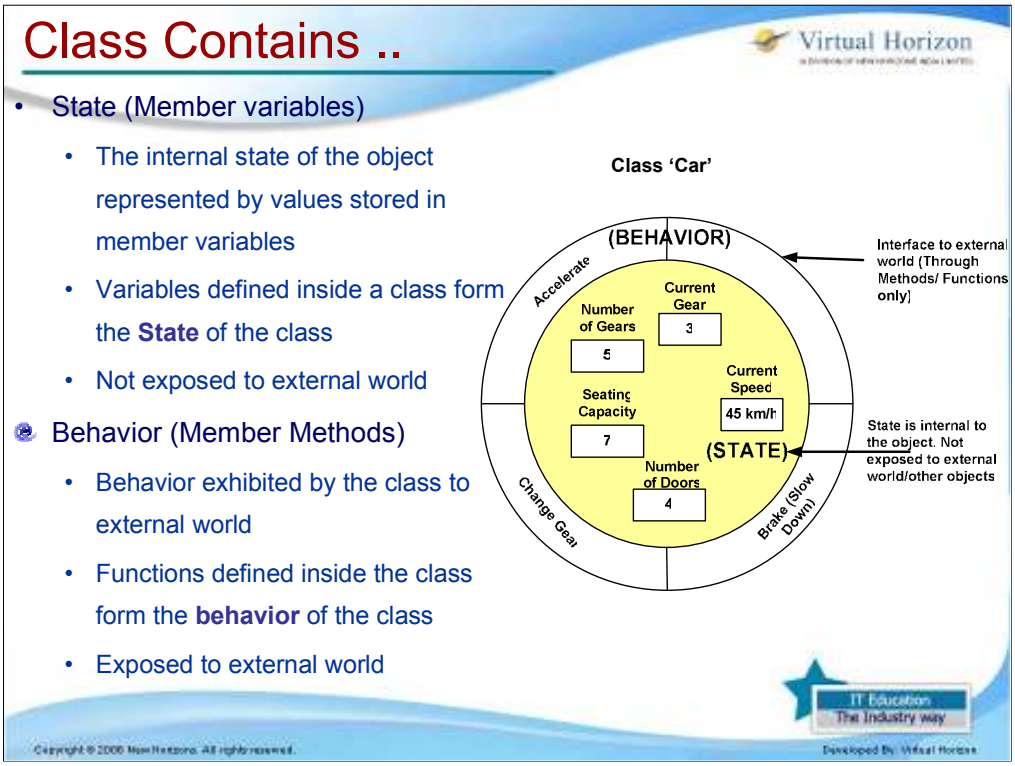
IT Education
The Industry way

Developed By: Virtual Horizon



- We never actually write the code for an object: what you write is the classes that is used to make objects..

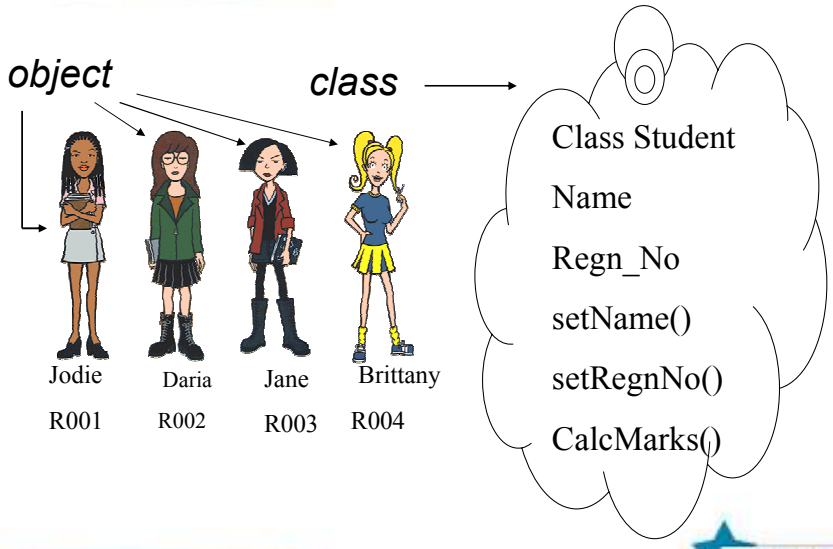
- Classes increase the efficiency and power of the object by:
    - Classifying objects
    - Relating objects to one another
    - Providing a mechanism to define and manage objects



- Every object belongs to (is an instance of) a class.

- An object may have fields, or variables
    - The class describes those fields with the help of member data.
- An object may have methods
    - The class describes those methods with the help of member methods.

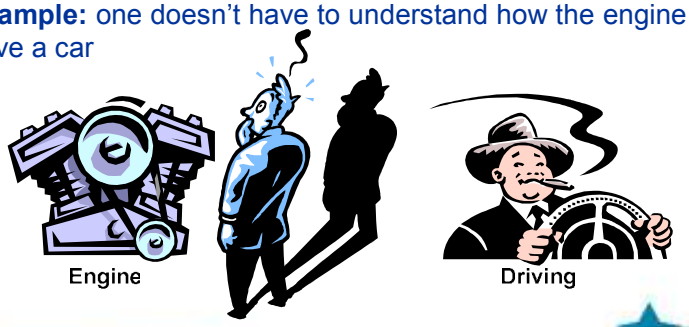
Example: Objects and Classes



Abstraction

Virtual Horizon

- The process of forming general and relevant concepts from a more complex scenario
  - Helps simplify the understanding and using of any complex system
  - Hide information that is not relevant
  - Simplifies by comparing to something similar in real world
  - **Example:** one doesn't have to understand how the engine works to drive a car

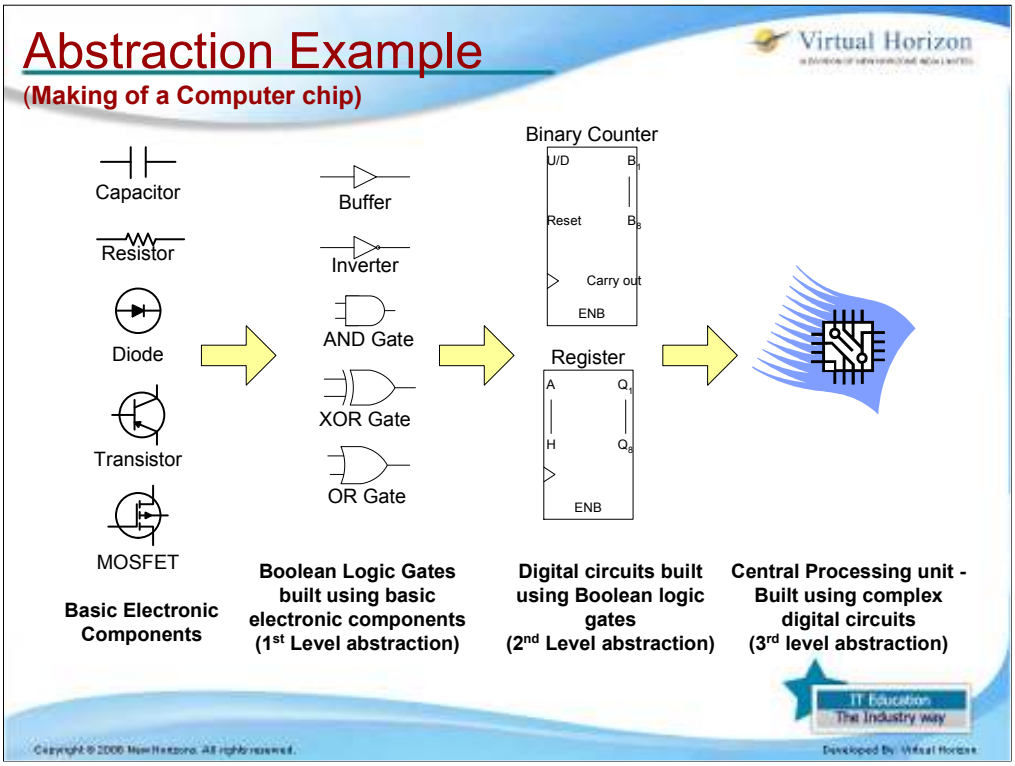


IT Education
The Industry way

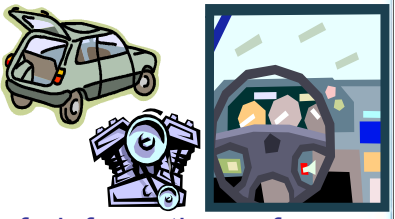
Developed By: Virtual Horizon

Abstraction Example

(Making of a Computer chip)

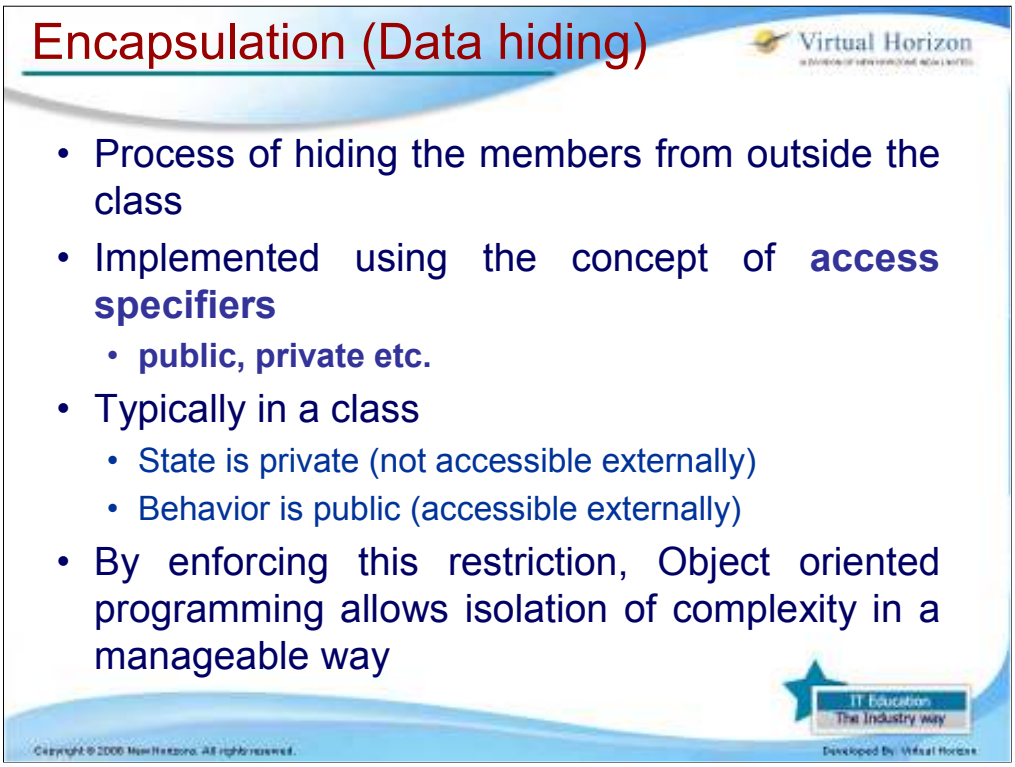


Encapsulation



- Encapsulate = “En” + “Capsulate”
  - “En” = “In a”
  - Encapsulate = “In a Capsule”
- Encapsulation means localization of information of knowledge within an object.
- Encapsulation also means “**Information hiding**”
  - "The process of hiding all the details of an object that do not contribute to its essential characteristics; typically, the structure of an object is hidden, as well as the implementation of its methods. The terms information hiding and encapsulation are usually interchangeable.“

Example: A car’s dashboard hides the complexity and internal workings of its engine.



- This is also called as **information hiding.**

- **IEEE defines Information Hiding as :**

  - *A software development technique in which each module's interfaces reveal as little as possible about the module's inner working and other modules are prevented from using information about the module that is not in the module's interface specification.*

Message Passing

Virtual Horizon

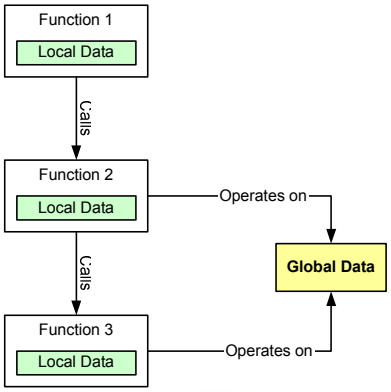
- An object by itself may not be very useful
- Useful work happens when one object invokes methods on other objects
  - Accessing data members of another object directly is not a good programming practice
- **Example:**
  - A car by itself is not capable of any activity
  - A person interacts with the car using steering wheel, gauges on dashboard and various pedals
  - This interaction between objects result in 'change of state' achieving something useful

IT Education
The Industry way

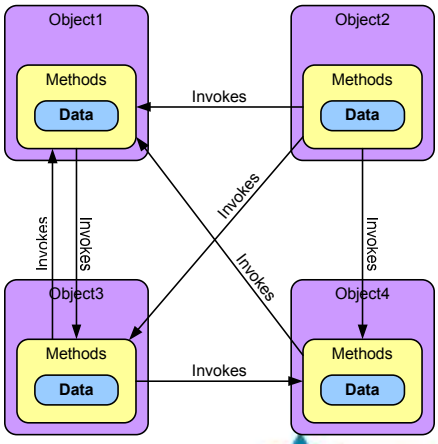
Developed By: Virtual Horizon

Procedural versus Object Oriented Programming

- In procedural programming, functions operate based on either local or global data
- In Object oriented programming, Objects exist and objects interact with other objects (message passing)



Calls in Procedural Language



Message passing between Objects

Access specifiers in a class

Virtual Horizon

- Access specifiers specify the accessibility of member variables and member methods
- **Private:** Accessible only within the class
- **Public:** Accessible externally (and also within the class)
- **Protected:** Similar to private under normal circumstances.
- **Note:** Some OO language uses more access specifiers also.
  - Example: Java supports ‘package’

IT Education
The Industry way

Developed By: Virtual Horizon

UML and UML Class Diagrams

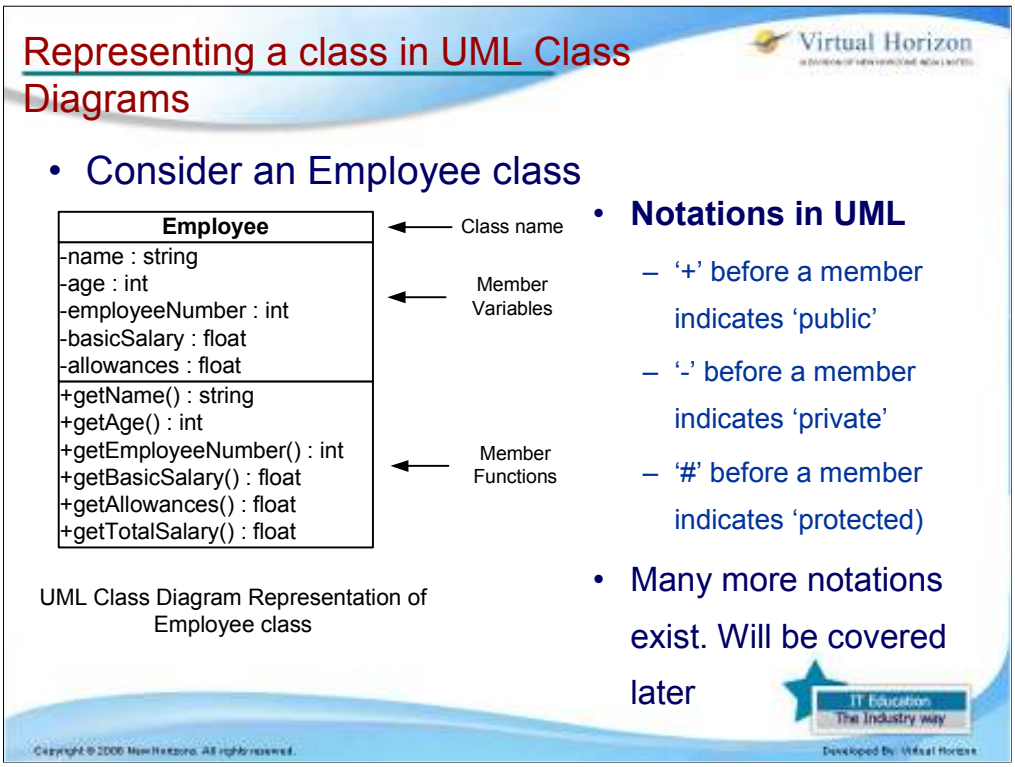
Virtual Horizon

- Unified Modeling language (UML) is a set of diagrams which pictorially represent object oriented design
- UML is extensively used by software engineers to communicate design
- In OO Design
  - Pictures are easier to understand than textual explanation

- UML Class diagram is a technique to represent classes and their relationships pictorially

IT Education
The Industry way

Developed By: Virtual Horizon



Here, you need to understand the various visibility labels (Access specifier) supported by different OO Languages.

The purpose of **protected** access specifier is discussed later.

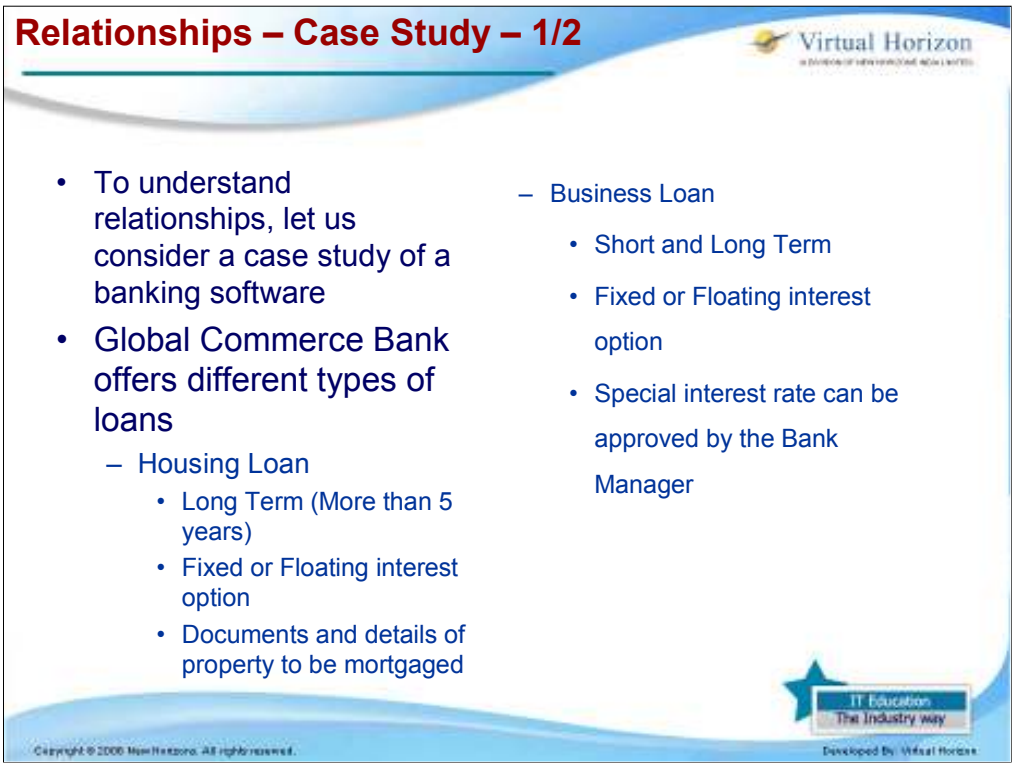
Relationships

Virtual Horizon

- Different types of relationships can exist between classes
- Identifying relationships helps design the objects better
  - Analogous to relations between entities in RDBMS design
  - (Entity Relationship Diagram)
- There are 3 types of relationships
  - **Is-A (or Kind-Of)**
  - **Has-A (or Part-Of)**
  - **Uses-A**

IT Education
The Industry way

Developed By: Virtual Horizon



Floating Interest Rate: For some types of loans, banks offer interest rate which keeps changing with time based on the economic situation.

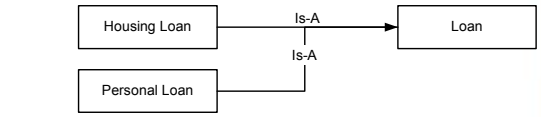
Moratorium Period: Lead time after which the repayment of loan starts in case of large loans.

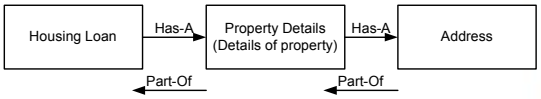
Floating Interest Rate: For some types of loans, banks offer interest rate which keeps changing with time based on the economic situation.

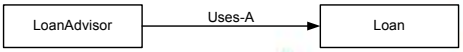
Moratorium Period: Lead time after which the repayment of loan starts in case of large loans.

Relationships identified in the case study

- **Is-A** Relationship (**Inheritance**)
  - **A class is similar to another class**
  - **Class is a different type of another class**
- **Has-A** Relationship (**Aggregation**)
  - **Class contains another class (as member)**
  - **Another class is part of the class**
- **Uses-A** Relationship (**Association**)
  - **Loosely coupled relationship**
  - **A class interacts with another class**

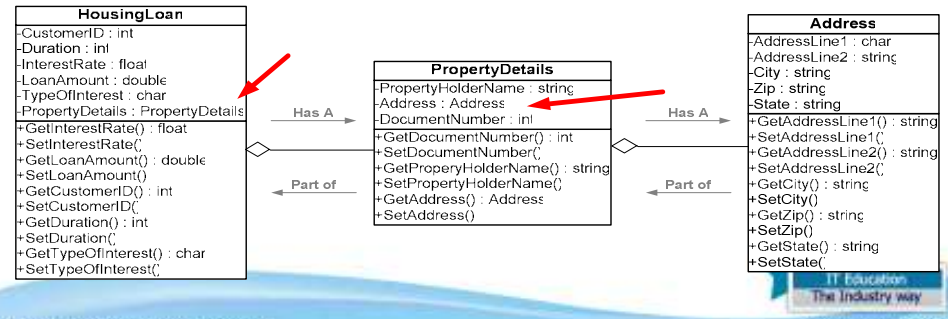






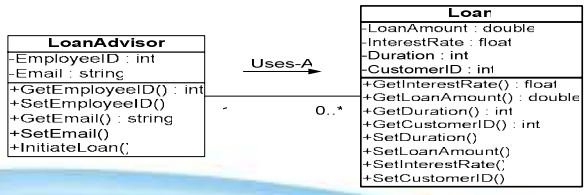
Has-A Relationship - Aggregation

- class HousingLoan has 'PropertyDetails' as a member variable
- class PropertyDetails has 'Address' as a member variable
  - Address is a generic class which can store any address (address of a property or address of a person etc)
  - **Has-A relationship is represented with a diamond headed line in UML**



Uses-A Relationship - Association

- Objects interacting with other objects. It may include
  - Creation of another type of object
  - Method invocation (Message passing) on already existing object
- Examples:
  - LoanAdvisor creates a Loan object for a new loan
  - LoanAdvisor invokes a method on Loan object

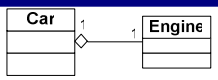
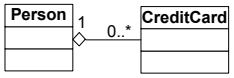
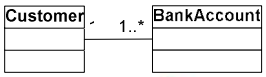


Relationships – Multiplicity of Relationships

Virtual Horizon

Notation	Meaning
1	One only
*	Many (More than one always)
0..1	Zero or One
0..*	Zero or Many
1..*	One or Many

*** Applies only to Has-A and Uses-A Relationships**

Multiplicity	Representation
One to One Aggregation *A car can have only one engine*	
One to Many Aggregation (Many = zero or more) *A person can have zero or more credit cards*	
One to Many Association (Many = one or more) *In a bank, a customer can use one or more accounts.*	

IT Education
The Industry way

Developed By: Virtual Horizon

Inheritance

Virtual Horizon

Types of Inheritance

- Inheritance are of the following types
    - Simple or Single Inheritance
    - Multi level or Varied Inheritance
    - Multiple Inheritance
    - Hierarchical Inheritance
    - Hybrid Inheritance
    - Virtual Inheritance

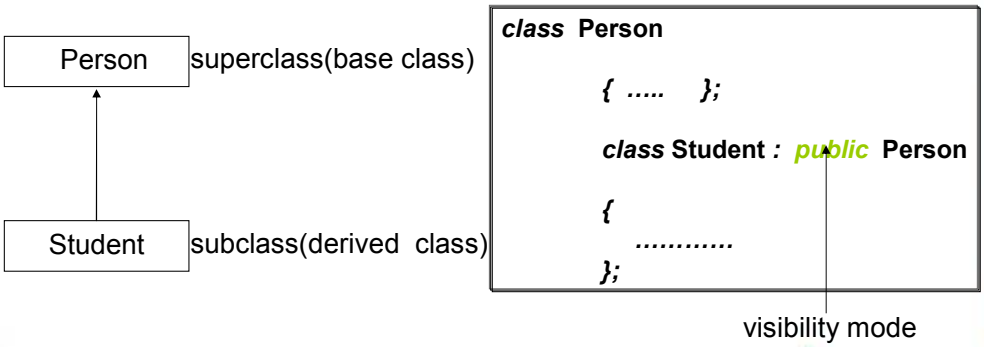
IT Education
The Industry way

Developed By: Virtual Horizon

Simple or Single Inheritance

Virtual Horizon

- This a process in which a sub class is derived from only one superclass.
- a Class Student is derived from a Class Person



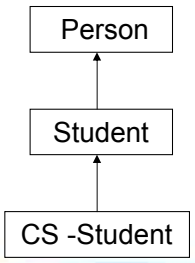
IT Education
The Industry way

Developed By: Virtual Horizon

Multilevel or Varied Inheritance

Virtual Horizon

- The method of deriving a class from another derived class is known as Multiple or Varied Inheritance.
- A derived class CS-Student is derived from another derived class Student.




```
Class Person
        { ……};
Class Student : public Person
        { ……};
Class CS -Student : public Student
        {  …….};
```

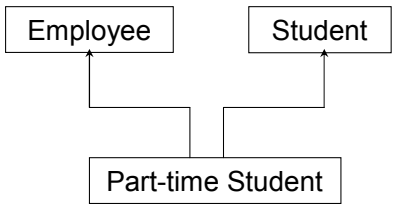
IT Education
The Industry way

Developed By: Virtual Horizon

Multiple Inheritance

Virtual Horizon

- A class is inheriting features from more than one super class
- Class Part-time Student is derived from two base classes, Employee and Student




```
Class Employee
        {……..};
Class Student
        {……..};
Class Part-time Student : public Employee,
                          public Student
        {…….};
```

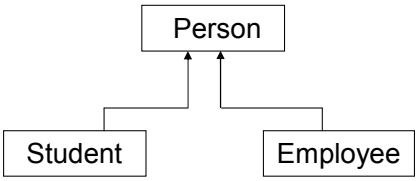
IT Education
The Industry way

Developed By: Virtual Horizon

Hierarchical Inheritance

Virtual Horizon

- **Many sub classes are derived from a single base class**
- **The two derived classes namely Student and Employee are derived from a base class Person.**




```
Class Person
        {……};
Class Student : public  Person
        {……};
Class Employee : public  Person
        {……};
```

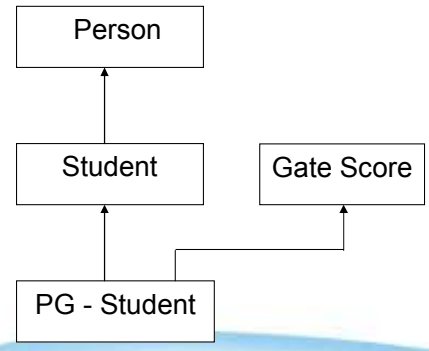
IT Education
The Industry way

Developed By: Virtual Horizon

Hybrid Inheritance

Virtual Horizon

- In this type, more than one type of inheritance are used to derive a new sub class
- Multiple and multilevel type of inheritances are used to derive a class PG-Student




```
Class Person
        { ……};
Class Student : public Person
        { ……};
Class Gate Score
        {…….};
Class PG - Student : public Student
                     public Gate Score
        {……..};
```

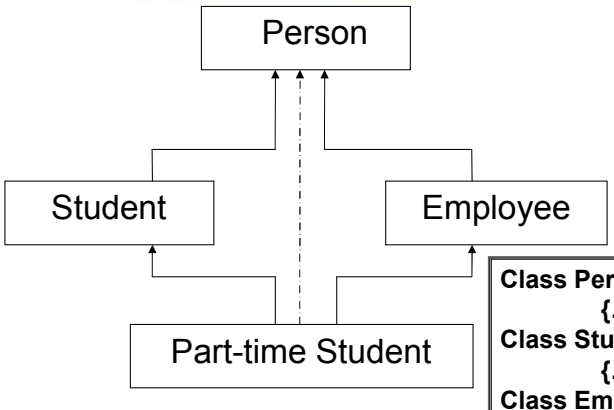
IT Education
The Industry way

Developed By: Virtual Horizon

Virtual Inheritance

- A sub class is derived from two super classes which in-turn have been derived from another class.
- The class Part-Time Student is derived from two super classes namely, Student and Employee.
- These classes in-turn derived from a common super class Person.
- The class Part-time Student inherits, the features of Person Class via two separate paths

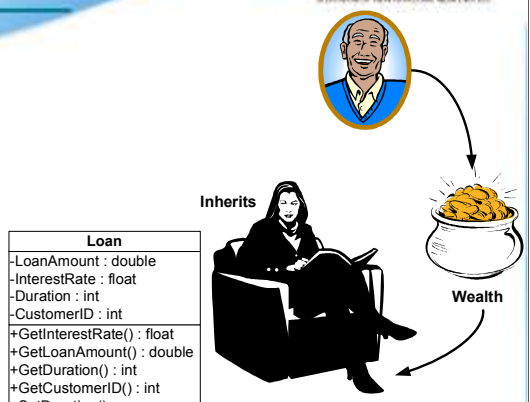
Virtual Inheritance

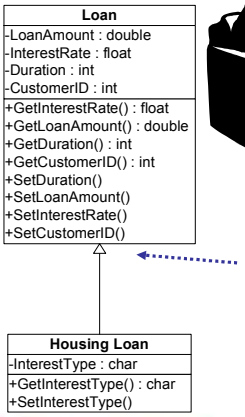



```
Class Person
        {……};
Class Student : public Person
        {……};
Class Employee : public Person
        {……};
Class Part-time Student : public Student,
                          public Employee
        {…….};
```

Is-A Relationship - Inheritance

- Inheritance refers to a class replicating some features or properties from another class
- Inheritance allows definition of new classes on similar lines of a **base class** (Also called **parent or Super class**)
- The class which inherits from another class is called as '**derived class**'

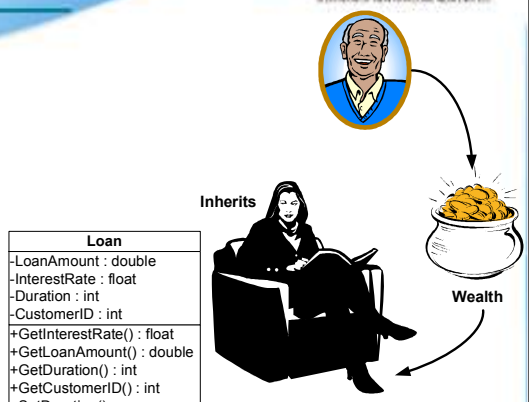




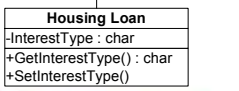
Note: Inheritance is represented by a triangle head arrow in UML Class diagrams

Is-A Relationship - Inheritance

- **Example:** The HousingLoan class inherits from Loan class
  - Need not redefine member variables and methods defined in parent class 'Loan
  - Loan → Base Class
  - HousingLoan → Derived Class



Note: Inheritance is represented by a triangle head arrow in UML Class diagrams



protected access specifier

Virtual Horizon

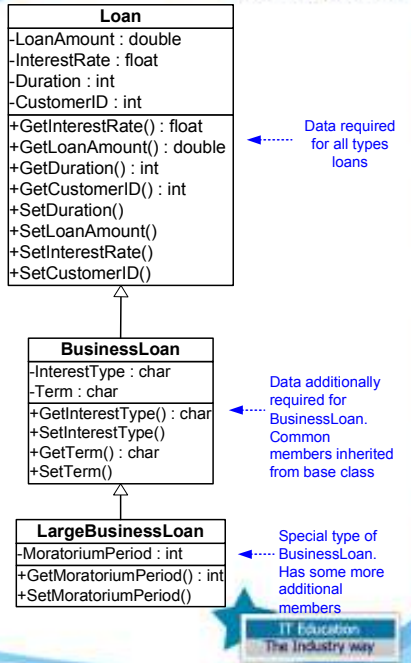
- Protected is similar to private in normal circumstances
  - Access is restricted to only within the class
- Derived classes can also access protected members

Developed By: Virtual Horizon

Multi-Level Inheritance

Virtual Horizon

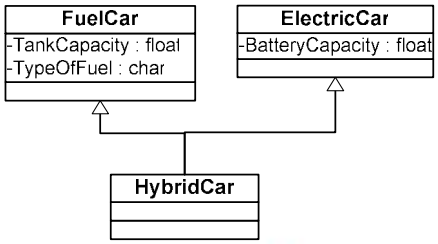
- A class can inherit from another class
  - Derived class inherits all the members of base class
- Another class can inherit from the derived class
  - The new class inherits all the member of all its ancestor classes
- **Example:**
  - BusinessLoan Inherits from Loan class
  - LargeBusinessLoan Inherits from BusinessLoan class



Multiple Inheritance (Rarely used)

Virtual Horizon

- Concept of a class inheriting from more than one base class
- **Example:** A Hybrid car can inherit from FuelCar and BatteryCar



- **Note:**
  - Multiple inheritance is rarely used because of the complexities it brings in
  - Modern OO languages like Java and C# don't support Multiple Inheritance

IT Education
The industry way

Developed By: Virtual Horizon

Advantages and Disadvantages of inheritance

Virtual Horizon

- Advantages
  - Promotes reusability
  - Helps in better abstraction, thereby resulting in better design
  - Eliminates duplication of code
- **Disadvantages**
  - Overuse of this concept (in cases where not necessary) leads to bad design
  - Wrong usage of inheritance can lead to code and design complexity

IT Education
The Industry way

Developed By: Virtual Horizon

Generalization and Specialization

Virtual Horizon

- To use inheritance
  - One has to identify similarities in among different classes
  - Move common data and methods to base class

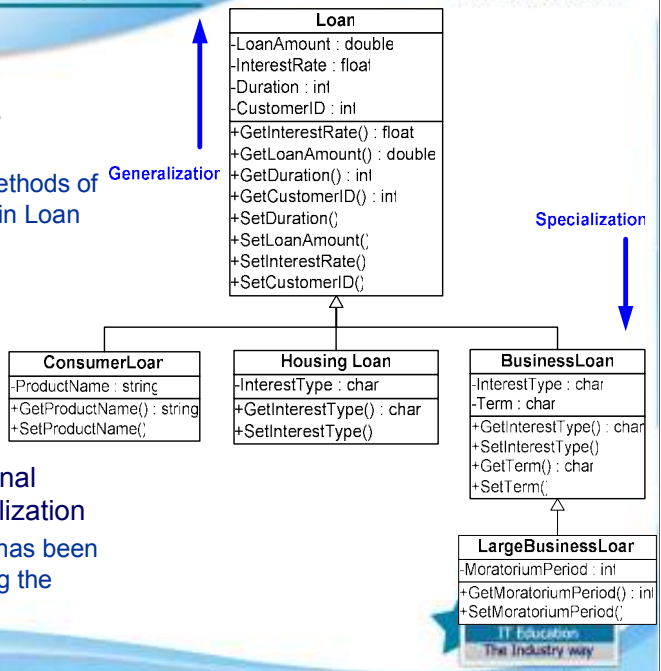
- **Generalization:** Process of identifying the similarities among different classes
- **Specialization:** Process of creating classes for specific need from a common base class

IT Education
The Industry way

Developed By: Virtual Horizon

Generalization and Specialization - Example

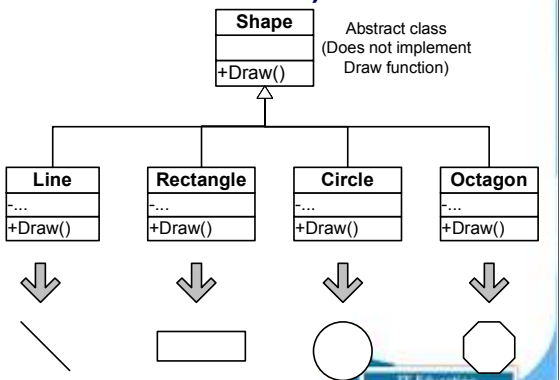
- 'Loan' class represents generalization
  - Common data and methods of all types of loans are in Loan class
- Derived classes 'HousingLoan',
- 'BusinessLoan', 'Personal Loan' represent specialization
  - Specific functionality has been achieved by extending the Loan class



Abstract class

- Outlines (sets a blueprint) for behavior of a class
- But does not implement the behavior fully
- Provides method signatures without implementation
- **(May or may not implement some methods)**

- Also called **Abstract Base Class**
- Abstract class cannot be instantiated
- The derived class must implement all the methods that are not implemented



Polymorphism

Virtual Horizon

- Refers to an object's ability to behave differently depending on its type
  - Poly = 'many'
  - morph = 'form'
- This characteristic enables making extensions to a class's functionality
- Two features of an object which achieve polymorphism
  - **Method Overloading** (or Function overloading)
  - **Method Overriding** (or Function overriding)

IT Education
The Industry way

Developed By: Virtual Horizon

Method Overloading

Virtual Horizon

- Practice of using same method name to denote several different operations
  - Some OO languages allow overloading of both functions and Operators (like +, - etc)
- **Example:**
  - Consider a String class which is a utility class designed to abstract and simplify string operations
  - 'Append' functions are overloaded to accept different types of data
    - One Append function appends an integer value to string, another Append function appends a float value

Method Overloading - Example

Virtual Horizon

```
class String {
   private:
      char* m_str;

   public:
      String (char* str);
      String (int size);
      ...
      ...
      void Append (int value);
      void Append (float value);
      void Append (char value);
      void Append (String* str);
      void Append (char* str);
}
```

```
// Object strRate created with "INR"
String* strRate = new String ("INR");

// Object strItems created with " Only"
String* strItems = new String (" Only");

// Appending a space character to String.
strRate->Append (' ');

// Appending a float to String
Str->Append (199.95f);

// Appending another String to strRate!
strRate->Append (strItems)

// Final contents is "INR 199.95 Only"
```

IT Education
The Industry way

Developed By: Virtual Horizon


Virtual Horizon
IT Education
The Industry way
Object Oriented Design and Best Practices
55

Case Study of Global Commerce Bank

Virtual Horizon

- **Let us consider a portion of the requirement from the case study**
- **We have seen the class design for Loans already in earlier slides**
- **Excerpts from Case Study (Course Material: Page 25)**

There are three types of bank employees who deal with Loans in the bank

Loan Advisors: Advises customers on various loans and can also initiate loans. The final approval of loans can be done only by the Bank Manager.

Bank Manager: Primarily responsible for approving all loans and fixing of special interest rates for Business Loans. The bank manager can also initiate loans

Teller: Can accept EMIs.

Steps in Object Oriented Design – 1/6

Virtual Horizon

- Step 1: Identify all the 'nouns' (type of objects or classes) in the requirement (Loan part has been done already)
  - **Loan Advisor**
  - **Bank Manager**
  - **Teller**

IT Education
The Industry way

Developed By: Virtual Horizon

Steps in Object Oriented Design – 1/6

Virtual Horizon

- Step 2: Identify the commonalities between classes (Generalization) if it is obvious.
  - **Do not force fit generalization where it doesn't make sense**
  - **Common factor is that all of them are employees of bank**
  - **Employees have some common attributes**

IT Education
The Industry way

Developed By: Virtual Horizon

Steps in Object Oriented Design – 2/6

Virtual Horizon

- Step 3: In any given situation, start with the simplest object which can be abstracted into an individual class
  - In the classes identified above, Employee is the simplest class
- Step 4a: Identify all the member variables and methods the class should have
  - Class Employee
    - **Member Variables:** EmployeeID, Email
    - **Methods:** Get/Set Employee ID, Get/Set Email

IT Education
The Industry way

Developed By: Virtual Horizon

Steps in Object Oriented Design – 3/6

Virtual Horizon

- Step 4b: Let us go ahead and identify the member variables and methods for the rest of the classes
  - **Class LoanAdvisor**
    - **Member Variables: EmployeeID, Email**
    - **Methods: Get/Set EmployeeID, Get/Set Email, Initiate Loan**

IT Education The Industry way

Steps in Object Oriented Design – 3/6

Virtual Horizon

- **Class BankManager**
  - **Member Variables: EmployeeID, Email**
  - **Methods: Get/Set EmployeeID, Get/Set Email, InitiateLoan, ApproveLoan**
- **Class Teller**
  - **Member Variables: EmployeeID, Email**
  - **Methods: Get/Set EmployeeID, Get/Set Email, AcceptEMI**

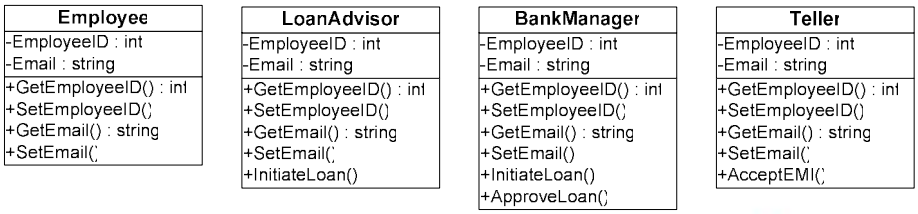
IT Education
The Industry way

Developed By: Virtual Horizon

Steps in Object Oriented Design – 4/6

Virtual Horizon

- **Step 5:** Ensure that the class is fully independent of other classes and contains all the attributes and methods necessary.
- **Step 6:** Keep All data members private or protected.

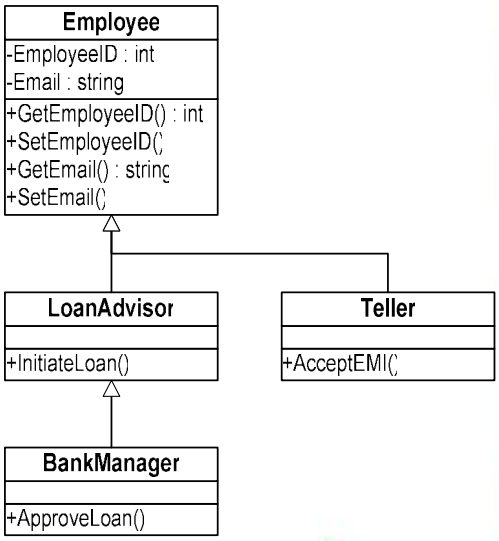


IT Education
The Industry way

Developed By: Virtual Horizon

Steps in Object Oriented Design – 5/6

- **Step 7 and 8:** The methods in class should completely abstract the functionality. The methods in the class should not be a force fit of procedural code into a class
- **Step 9:** Inherit and extend classes from the base class ONLY IF situation has scope for it.
  - Shift the commonalities among classes to the base class



Steps in Object Oriented Design – 6/6

Virtual Horizon

- **Step 10:** Define the "Has-A" and "Uses-A" relationship among classes
  - **(Is-A has been defined already in previous step)**
  - **Uses-A and Has-A could've been defined if we had considered the entire case study**

IT Education
The Industry way

Developed By: Virtual Horizon

Steps in Object Oriented Design – 6/6

Virtual Horizon

- **Step 11:** Keep the number of classes in your application under check (Do not create any unnecessary classes)
  - **(Design in previous slide doesn't have any unnecessary classes)**
- **Step 12:** Always REVIEW your design
  - To eliminate any redundancy
  - To assess whether the relationships identified make sense

IT Education
The Industry way

Developed By: Virtual Horizon

Best Practices in OO Design – 1/3

Virtual Horizon

- Defining a class
  - Keep data members private or protected
  - Only data members which can be made public should be constants
  - Methods should completely abstract functionality (behavior) of the class
  - For an existing class, if any new functionality (behavior) is required, expose it only as methods
  - If your class has only ONE DATA member check and see if this class is really required or it can be made as data member in another class

IT Education
The Industry way

Developed By: Virtual Horizon

Best Practices in OO Design – 2/3

Virtual Horizon

- Relationships, Complexity management
  - Class should be always self-sufficient in functionality
    - (If required it should be used in another project as is)
  - If two classes are inter-dependent so much that one cannot function without the other, there may be a possible defect in design
    - (Do not partition functionality across different classes)

IT Education
The Industry way

Developed By: Virtual Horizon

Best Practices in OO Design – 2/3

Virtual Horizon

- Do not create unnecessary relationships between classes
- Do not try to force fit all OO concepts into design
  - It is not mandatory that your design should have inheritance and polymorphism etc.
  - Purely, the situation decides what needs to be in the class

IT Education
The Industry way

Developed By: Virtual Horizon

Best Practices in OO Design – 3/3

Virtual Horizon

- Common Design practices
  - Avoid writing too many classes for every single functionality in the class
  - When using frameworks (or Third Party APIs or Built in Libraries) reuse as many classes from them as much as possible. Do not reinvent the wheel
    - **Example:** Java provides a Calendar and GregorianCalendar class for date related functions. Many programmers tend to write their own date logic instead of simply using this class

IT Education
The Industry way

Best Practices in OO Design – 3/3

Virtual Horizon

- Always REVIEW Your design before implementations
  - To eliminate any redundancy
  - To assess whether the relationships identified make sense

Developed By: Virtual Horizon

Procedural versus Object Oriented Programming – 1/2

Virtual Horizon

Procedural Programming	Object Oriented Programming
Emphasis on algorithms, procedures	**Emphasis on Data;** binding of data structures with methods that operate on data
Real world is represented by logical entities and control flow. Tries to fit real life problem into procedural language	Real world is represented by objects mimicking external entities. Allows modeling of real life problem into objects with state and behavior
In a given module, data and procedures are separate	Data (State) is encapsulated effectively by methods (Behavior)
Program modules are linked through parameter passing mechanism	Program modules are integrated parts of overall program. Objects interact with each other by Message passing

IT Education
The Industry way

Developed By: Virtual Horizon

Procedural versus Object Oriented Programming – 2/2

Procedural Programming	Object Oriented Programming
Uses abstraction at procedure level	Uses abstraction at class and object level
Algorithmic decomposition tends to focus on the sequence of events	Object Oriented decomposition focuses on abstracted objects and their interaction
Passive and dumb data structures used by active methods	Active and intelligent data structures (object) encapsulates all passive procedures
Procedural languages: C, COBOL, PASCAL	OO Languages: C++, Small Talk, Java, C#


Virtual Horizon
IT Education
The Industry way
Appendix – Case Studies
73

Case Study 1: SpedFast Courier Company

- Create a UML Class diagram with the design for the requirements of SpedFast Courier Company

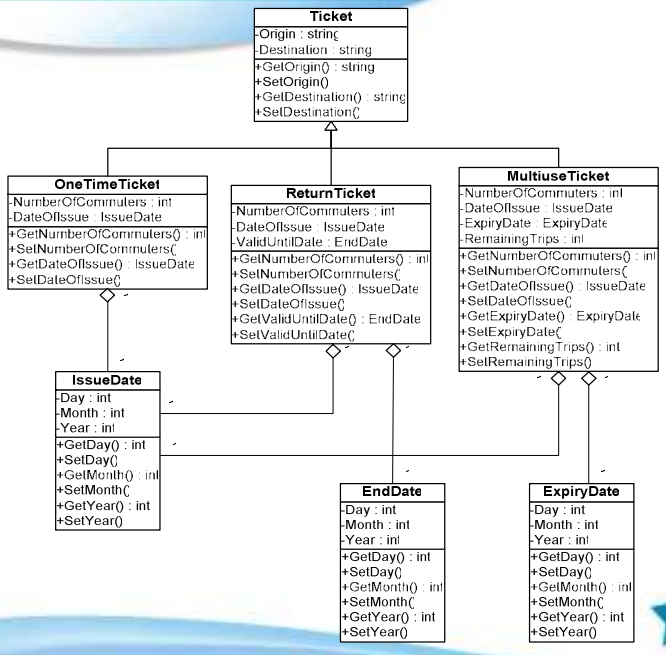
Microsoft Word Document

Case Study 2: Review the OO Design of D-Rail Inc.

- Read and understand the requirements of D-Rail Inc
- Review the OO design of D-Rail Inc and suggest a better design
- Refer to the “Best Practices”

Microsoft Word Document

Case Study 2: D-Rail Inc design




Virtual Horizon
Thank You
IT Education
The Industry way
Developed By: Virtual Horizon