TRINITY COLLEGE FOR WOMEN NAMAKKAL

Department of Computer Science

OBJECT ORIENTED ANALYSES AND DESIGN

19PCSE07-ODD Semester

Presented by

S.BHUVANESWARI

Assistant Professor

Department of Computer Science

http://www.trinitycollegenkl.edu.in/

Thinking in Objects and UML - 1

Then too, there are sets of **proven design solutions** to problems that are considered 'best practices.'

- Certain 'groupings' of classes with specific responsibilities / interfaces.
- These provide specific solutions to specific problems.
- Called **Design Patterns**

We will discuss (much later) these standard patterns and how to **apply** them to develop solutions to common design problems.

Of course, design (solution to requirements) 'assume' a robust **requirements analysis** has taken place.

Use Cases are often used to capture **stories** of requirements and are often views as 'constituting' the **functional** requirements, but NOT the software quality factors (non-functional requirements).

Use Cases are **not specifically designed** to be object-oriented, but rather are meant to capture how an application will be used.

Many methods for capturing requirements.
We will concentrate on Use Cases (ahead).

Basic Terms: Iterative, Evolutionary, and Agile

1. Introduction

Iterative - the entire project will be composed of min-projects and will iterate the same activities again and again (but on different part of the project AND with different emphases) until completion.

Evolutionary (or incremental) - the software grows by increments (to be opposed to the traditional, and somewhat old-fashioned, Waterfall model of software development).

Agile - we will use a light approach to software development rather than a very rigid one (which may be needed for a safety-critical system for example)

This kind of approach seems better at treating software development as a **problem solving activity**; also the use of objects makes it amenable.

Why the Unified Process:

The Unified Process is a popular iterative software development process.

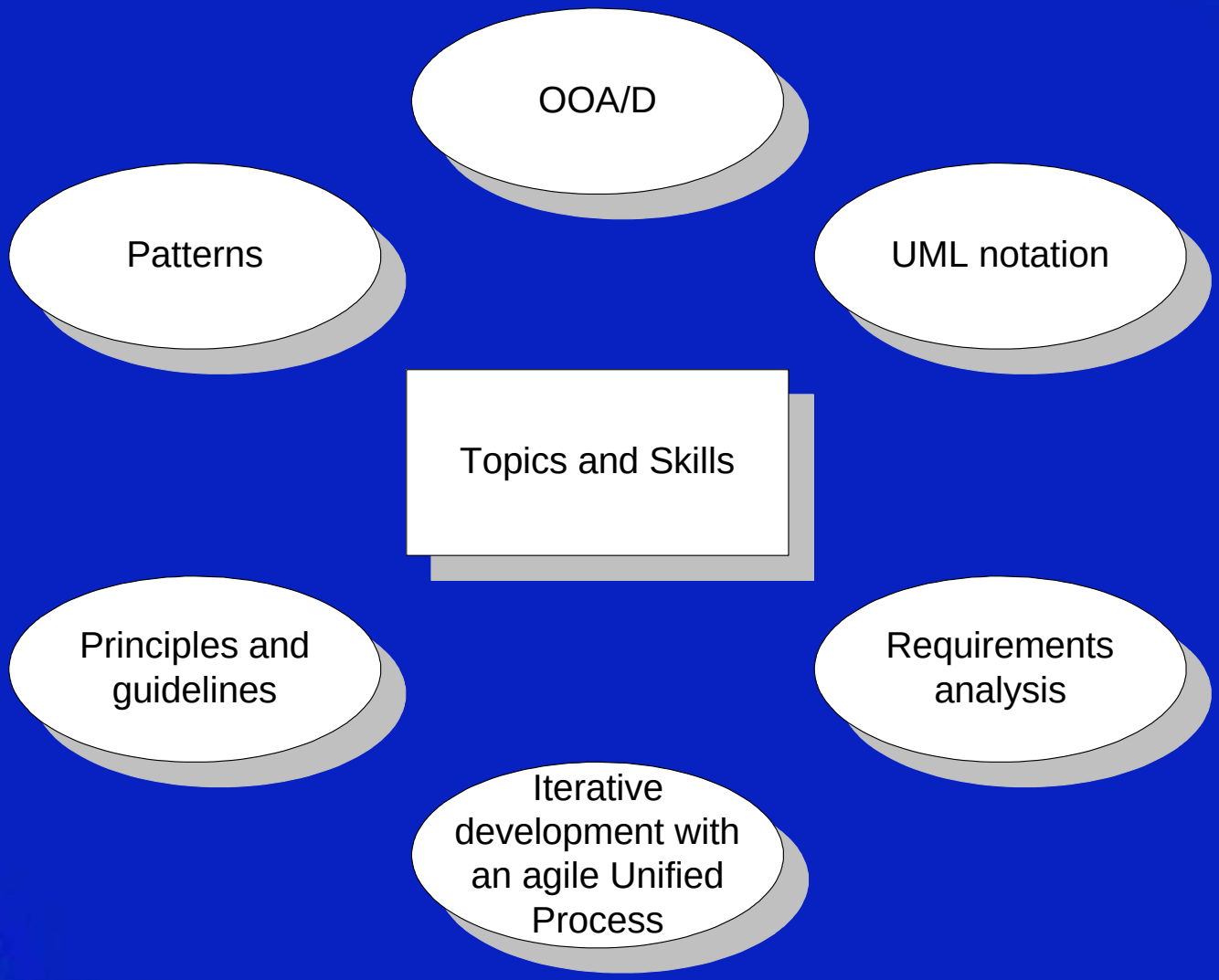
Iterative and evolutionary development involves relatively early programming and testing of a partial system, in repeated cycles.

It typically also means that development starts before the exact software requirements have been specified in detail;

Feedback (based on measurement) is used to clarify, correct and improve the evolving specification:

This is in complete contrast to what we usually mean by engineering!

We will be studying all of the topics Using



The Rush to Code

Analysis: - investigate the **problem** and the **requirements.**

What is needed? Required functions? Investigate domain objects.

Problem Domain

The **Whats** of a system.

Do the right thing (analysis)

Design:

Conceptual solution that meets requirements.

Not an implementation

E.g. Describe a database schema and software objects.

Avoid the CRUD activities and commonly understood functionality.

The Solution Domain

The '**Hows**' of the system

Do the thing right (design)

What is Object-Oriented Analysis and Design

OOA: we find and describe **business objects** or concepts in the **problem domain**

OOD: we define how these **software objects <u>collaborate</u>** to meet the requirements.

Attributes and methods.

OOP: Implementation: we implement the design objects in, say, Java, C++, C#, etc.

<u>Object:</u>

An object is a real-world element in an object–oriented environment that may have a physical or a conceptual existence. Each object has −

Identity that distinguishes it from other objects in the system.

State that determines the characteristic properties of an object as well as the values of the properties that the object holds.

Behavior that represents externally visible activities performed by an object in terms of changes in its state.

<u>Class:</u>

A class represents a collection of objects having same characteristic properties that exhibit common behavior

A set of attributes for the objects that are to be instantiated from the class.

A set of operations that portray the behavior of the objects of the class. Operations are also referred as functions or methods.

Inheritance

Inheritance is the mechanism that permits new classes to be created out of existing classes by extending and refining its capabilities. The existing classes are called the base classes/parent classes/super-classes,

Types of Inheritance

Single Inheritance − A subclass derives from a single super-class.

Multiple Inheritance − A subclass derives from more than one super-classes.

Multilevel Inheritance − A subclass derives from a super-class which in turn is derived from another class and so on.

Hierarchical Inheritance − A class has a number of subclasses each of which may have subsequent subclasses, continuing for a number of levels, so as to form a tree structure.

Hybrid Inheritance − A combination of multiple and multilevel inheritance so as to form a lattice structure.

Polymorphism

Polymorphism is originally a Greek word that means the ability to take multiple forms.

Encapsulation:Encapsulation is the process of binding both attributes and methods together within a class.

Data Hiding:Typically, a class is designed such that its data (attributes) can be accessed only by its class methods and insulated from direct outside access.

THANK YOU

http://www.trinitycollegenkl.edu.in/