

Vietnam and Japan
Joint ICT HRD Program

ITSS Software Development
Chapter 5. Use case analysis

Nguyen Thi Thu Trang
trangntt-fit@mail.hut.edu.vn

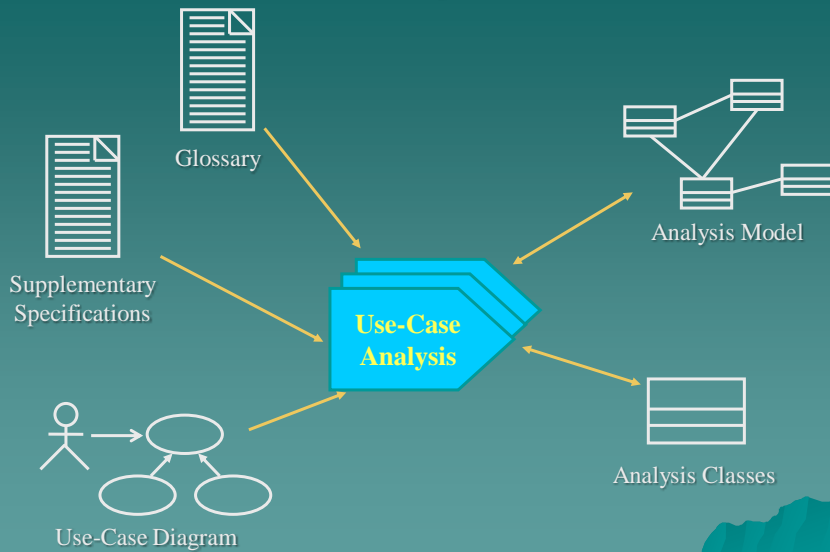
1

Content

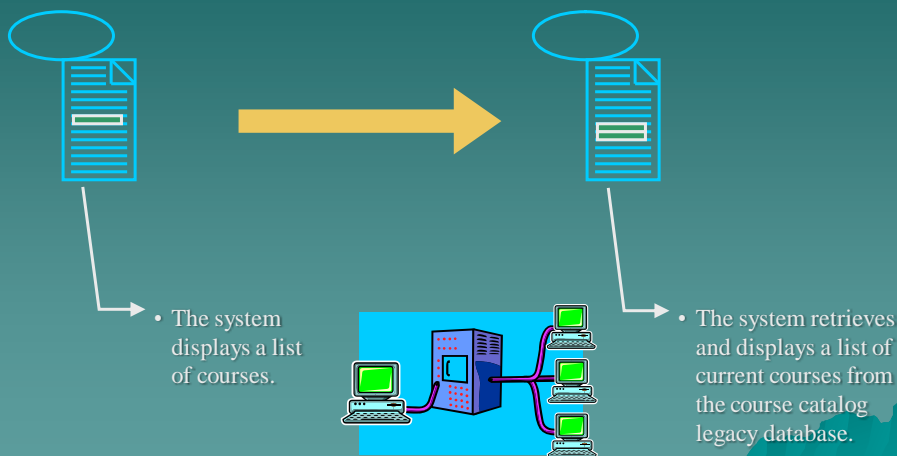
- ⇒ 1. Overview of Use case analysis
- 2. Analysis classes
- 3. Distribute Use-Case Behavior to Classes

2

Use-Case Analysis Overview



Supplement the Use-Case Description



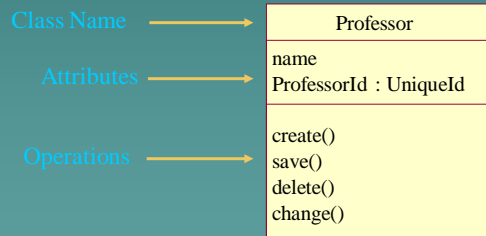
Content

1. Overview of Use case analysis
- ➡ 2. Analysis classes
3. Distribute Use-Case Behavior to Classes

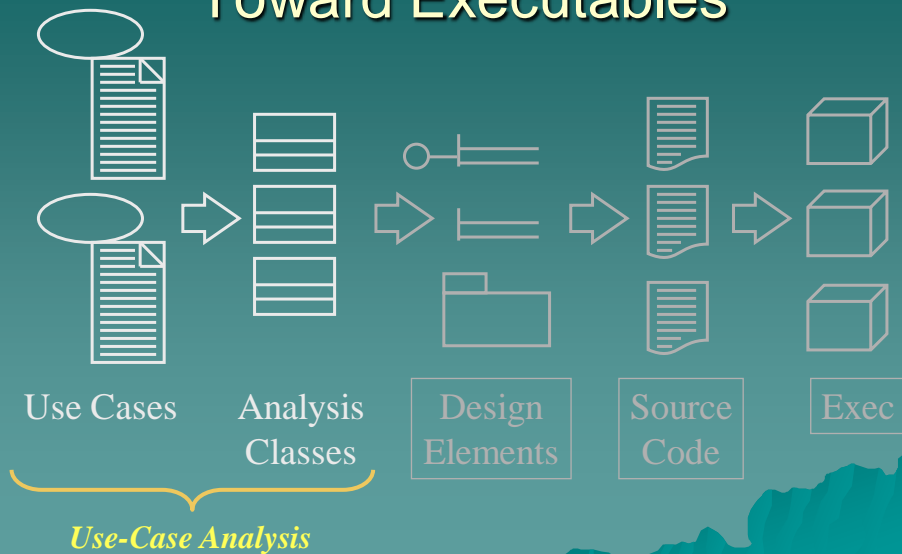
5

Review: Class

- ◆ An abstraction
- ◆ Describes a group of objects with common:
 - Properties (attributes)
 - Behavior (operations)
 - Relationships
 - Semantics

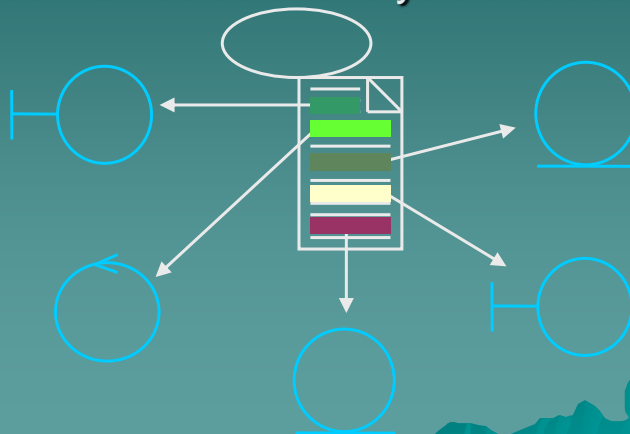


Analysis Classes: A First Step Toward Executables

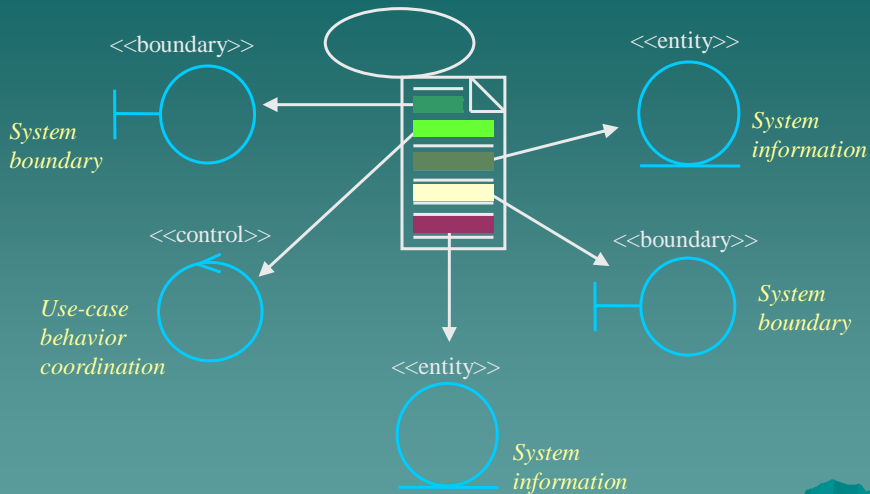


Find Classes from Use-Case Behavior

- ◆ The complete behavior of a use case has to be distributed to analysis classes



Types of Analysis Classes



2.1. Boundary Classes

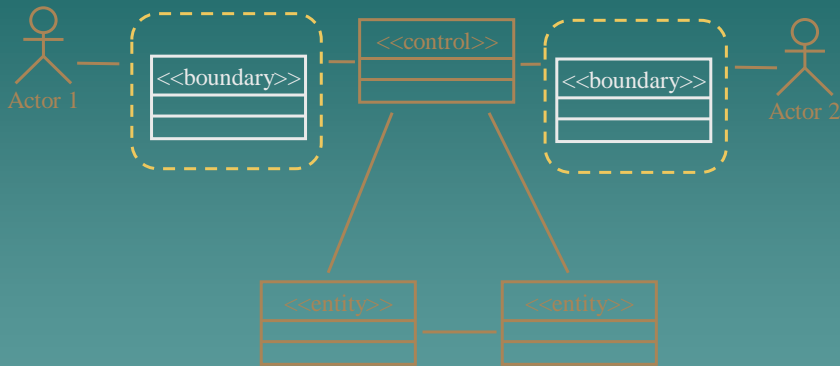
- ◆ Intermediate between the interface and something outside the system
- ◆ Several Types
 - User interface classes
 - System interface classes
 - Device interface classes
- ◆ *One boundary class per actor/use-case pair*

Analysis class stereotype



Environment dependent.

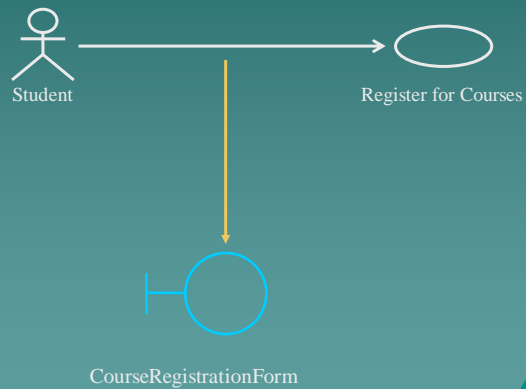
The Role of a Boundary Class



Model interaction between the system and its environment.

Example in Course Registration CS: Finding Boundary Classes

- ◆ One boundary class per actor/use case pair



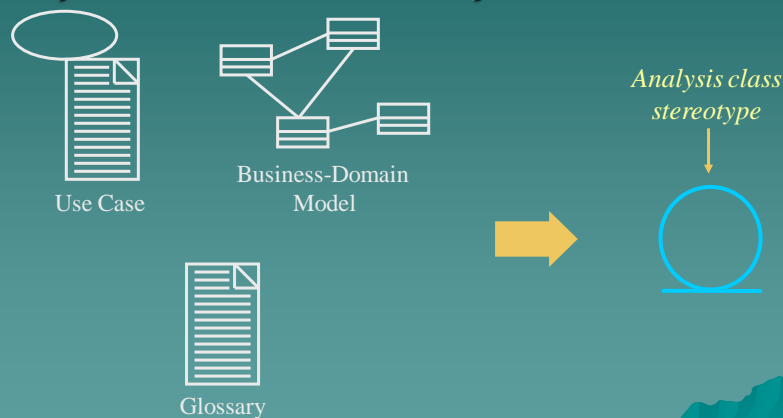
Guidelines: Boundary Classes

- ◆ User Interface Classes
 - Concentrate on what information is presented to the user
 - Do NOT concentrate on the UI details
- ◆ System and Device Interface Classes
 - Concentrate on what protocols must be defined
 - Do NOT concentrate on how the protocols will be implemented

Concentrate on the responsibilities, not the details!

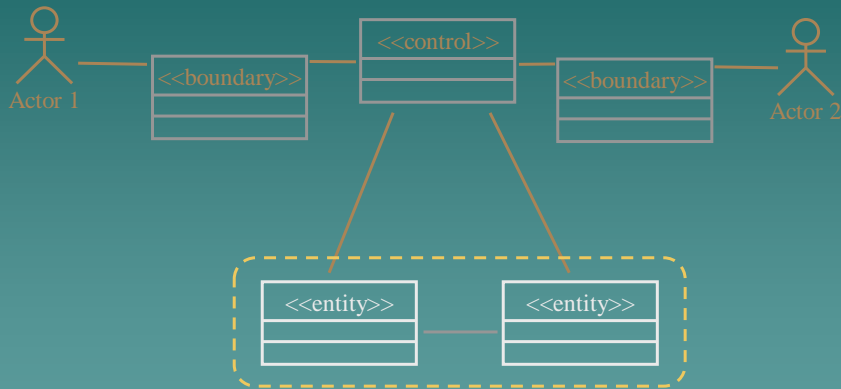
2.2. Entity Classes

- ◆ Key abstractions of the system



Environment independent.

The Role of Entity Classes



Store and manage information in the system.

Guidelines: Entity Classes

- ◆ Use use-case flow of events as input
- ◆ Key abstractions of the use case
- ◆ Traditional, filtering nouns approach
 - Underline noun clauses in the use-case flow of events
 - Remove redundant candidates
 - Remove vague candidates
 - Remove actors (out of scope)
 - Remove implementation constructs
 - Remove attributes (save for later)
 - Remove operations

Example in Course Registration CS: Finding Entity Classes

- ◆ For “Register For Course” use case, there are some candidate entity classes:



3.3. Control Classes

- ◆ Provide coordinating behavior in the system
- ◆ model control behavior specific to one or more use cases



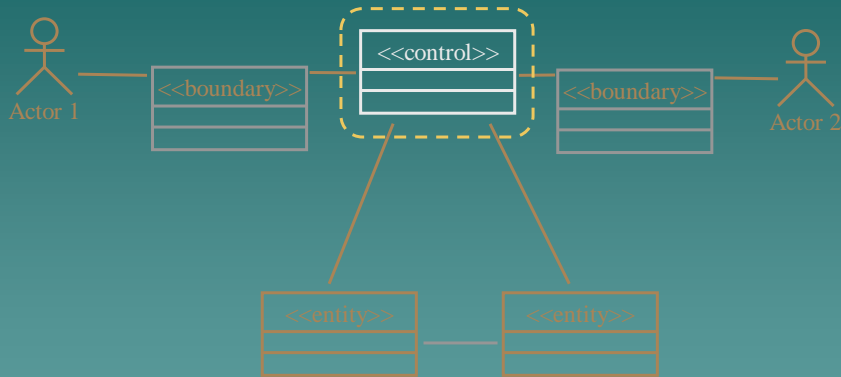
Use Case



*Analysis class
stereotype*

Use-case dependent. Environment independent.

The Role of Control Classes



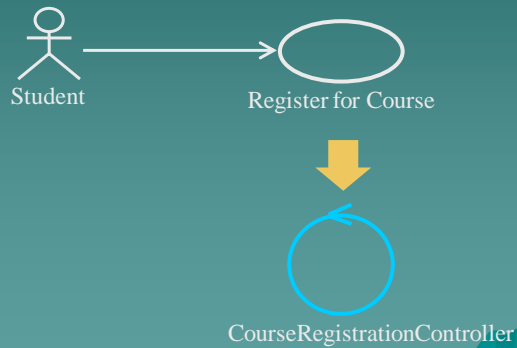
Coordinate the use-case behavior.

Guidelines: Control Classes

- ◆ In general, identify one control class per use case.
- ◆ The system can perform some use cases without control classes by using just entity and boundary classes.
 - This is particularly true for use cases that involve only the simple manipulation of stored information.
- ◆ More complex use cases generally require one or more control classes to coordinate the behavior of other objects in the system.
 - Examples of control classes include transaction managers, resource coordinators, and error handlers.

Example in Course Registration CS: Finding Control Classes

- ◆ For “Register for Course” use case:



Course Registration CS Summary: Analysis Classes



Use-Case Model

Design Model



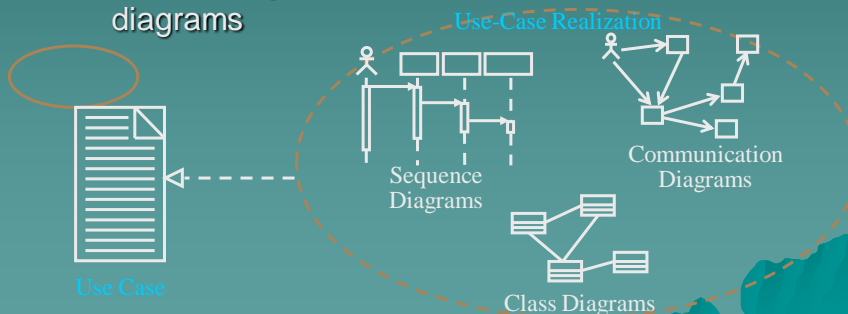
Content

1. Overview of Use case analysis
2. Analysis classes
- ➡ 3. Distribute Use-Case Behavior to Classes

23

3. Distribute Use-Case Behavior to Classes

- ◆ For each use-case flow of events:
 - Identify analysis classes
 - Allocate use-case responsibilities to analysis classes
 - Model analysis class interactions in Interaction diagrams



3.1. Guidelines: Allocating Responsibilities to Classes

- ◆ Use analysis class stereotypes as a guide
 - Boundary Classes
 - ◆ Behavior that involves communication with an actor
 - Entity Classes
 - ◆ Behavior that involves the data encapsulated within the abstraction
 - Control Classes
 - ◆ Behavior specific to a use case or part of a very important flow of events

3.1. Guidelines: Allocating Responsibilities to Classes (2)

- ◆ Who has the data needed to perform the responsibility?
 - If one class has the data, put the responsibility with the data
 - If multiple classes have the data:
 - ◆ Put the responsibility with one class and add a relationship to the other
 - ◆ Create a new class, put the responsibility in the new class, and add relationships to classes needed to perform the responsibility
 - ◆ Put the responsibility in the control class, and add relationships to classes needed to perform the responsibility

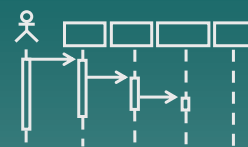
3.2. Interaction Diagrams

- ◆ Generic term that applies to several diagrams that emphasize object interactions
 - Sequence Diagram
 - Communication Diagram
- ◆ Specialized Variants
 - Timing Diagram
 - Interaction Overview Diagram

27

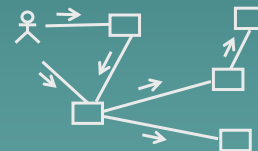
3.2. Interaction Diagrams (2)

- ◆ Sequence Diagram
 - Time oriented view of object interaction



Sequence Diagrams

- ◆ Communication Diagram
 - Structural view of messaging objects



Communication Diagrams

3.2. Interaction Diagrams (3)

◆ Timing Diagram

- Time constraint view of messages involved in an interaction



Timing Diagrams

◆ Interaction Overview Diagram

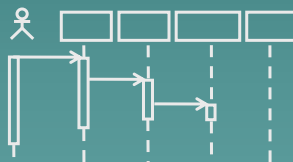
- High level view of interaction sets combined into logic sequence



Interaction Overview Diagrams

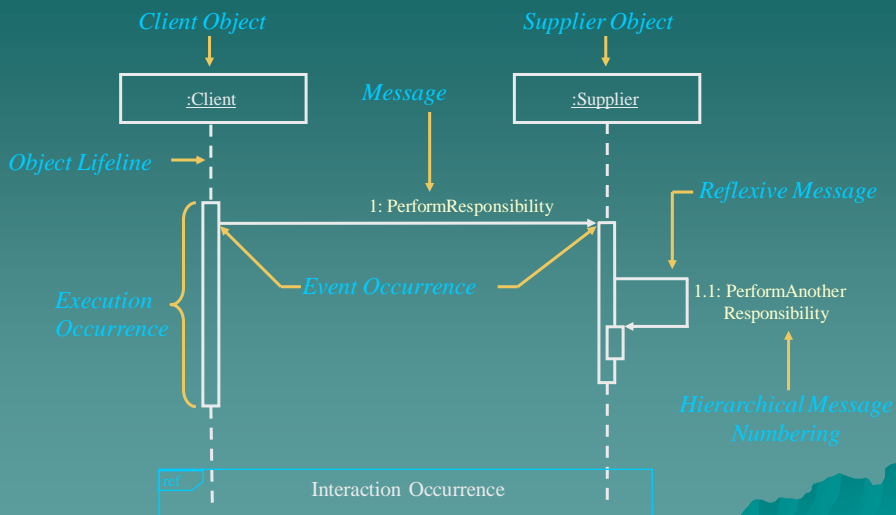
3.2.1. Sequence Diagram

- ◆ A sequence diagram is an interaction diagram that emphasizes the time ordering of messages.
- ◆ The diagram shows:
 - The objects participating in the interaction.
 - The sequence of messages exchanged.



Sequence Diagram

The Anatomy of Sequence Diagrams

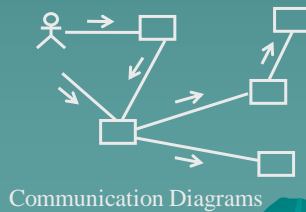


Exercise: Course Registration CS

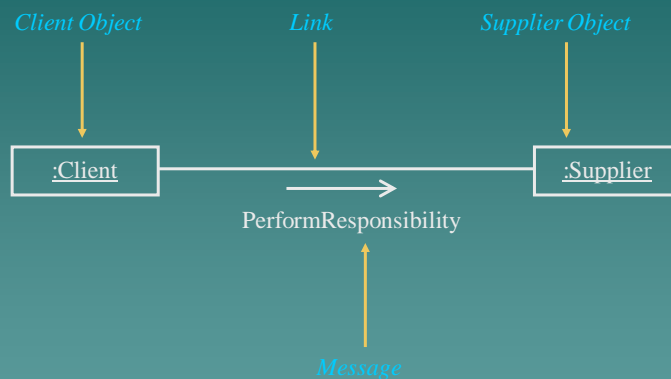
- ◆ Draw a sequence diagram for “Register for course” use case

3.2.2. Communication Diagram

- ◆ A communication diagram emphasizes the organization of the objects that participate in an interaction.
- ◆ The communication diagram shows:
 - The objects participating in the interaction.
 - Links between the objects.
 - Messages passed between the objects.



The Anatomy of Communication Diagrams

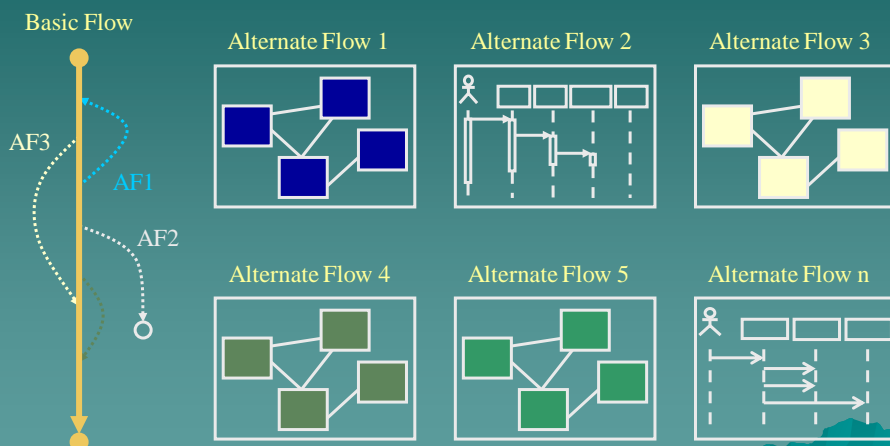


Exercise: Course Registration CS

- ◆ Draw a communication diagram for “Register for course” use case

35

One Interaction Diagram Is Not Good Enough



3.2.3. Sequence and Communication Diagram Comparison

◆ Similarities

- Semantically equivalent
 - ◆ Can convert one diagram to the other without losing any information
- Model the dynamic aspects of a system
- Model a use-case scenario

3.2.3. Sequence and Communication Diagram Comparison (2)

Sequence diagrams	Communication diagrams
<ul style="list-style-type: none">▪ Show the explicit sequence of messages▪ Show execution occurrence▪ Better for visualizing overall flow▪ Better for real-time specifications and for complex scenarios	<ul style="list-style-type: none">▪ Show relationships in addition to interactions▪ Better for visualizing patterns of communication▪ Better for visualizing all of the effects on a given object▪ Easier to use for brainstorming sessions

Checkpoints: Analysis Classes

- ◆ Are the classes reasonable?
- ◆ Does the name of each class clearly reflect the role it plays?
- ◆ Does the class represent a single well-defined abstraction?
- ◆ Are all attributes and responsibilities functionally coupled?
- ◆ Does the class offer the required behavior?
- ◆ Are all specific requirements on the class addressed?



Checkpoints: Message Design

- ◆ Have all the main and/or sub-flows been handled, including exceptional cases?
- ◆ Have all the required objects been found?
- ◆ Have all behaviors been unambiguously distributed to the participating objects?
- ◆ Have behaviors been distributed to the right objects?
- ◆ Where there are several Interaction diagrams, are their relationships clear and consistent?



Question?

