

OO Modeling and Class Diagrams

13 January 2010
Lecture 13

Topics for Today

- Object Oriented Modeling and Class Diagrams
- Building Class Diagrams
- Mapping Class Diagrams to Relational Database Schemas
- Merging Objects and Functions: Introduction to FOOM

- Source: PS04 1.2, 1.3, 1.5, 2.2,
 - Ramsin and Kabeli 48-52 (on Telem)

Objects and ERD

- Objects and Entities have similarities and differences

	Objects	Entities
Properties	Variable name Type Length/Size Access/Visibility	Name Type
Actions	Names Parameters Logic	
Connections	References/Pointers To other Objects	Relationship symbols To other Entities

Object Oriented Modeling

- Objects
 - Properties, Variables
 - Methods, Messages

- Objects can be connected to other objects
 - Is A – inheritance
 - Has A – contains
 - References A – has a pointer to

- Similar to Entities in ERD

ERD to Objects

- ERD can be the starting point for objects
 - Need to add actions (methods)
 - ERD is a static picture of items while objects include dynamic behavior

- We may design classes for a number of reasons:
 - Support or instantiate data stores
 - Aid in data processing

- When designing classes, use a **language independent manner**
 - Worry about the logic, not syntax
 - Later you can choose Java, C++, C#, etc.

Classes and Class Diagrams

- Classes are much more flexible than ERD
 - Can show **behavior**
 - Can more clearly define actions and procedural steps
 - That also means it's harder to do a good job at it...

- Good OO Design includes:
 - **Minimality** – don't have extra classes
 - **Structure** – Classes should give a logical structure to the solution, allocating information into conceptual blocks
 - **Interfaces and information hiding** – show only as much information as is necessary
 - **Inheritance and Reuse** – Take existing classes or code from different sources to speed up development
 - **Polymorphism** – Polymorphic typing lets you better leverage inheritance

Class Diagrams

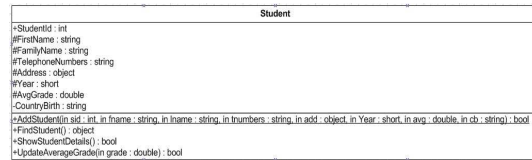
- Aside from the main programming language, we can
- Unified Modeling Language (UML)
 - Offers tools for modeling OO systems
 - Many different kinds of graphs, charts, diagrams
- Class Diagrams
 - Abstract representation of a class

January 13, 2010

ISE 323: Information Systems Engineering 1

7

Class Diagrams



January 13, 2010

ISE 323: Information Systems Engineering 1

8

Class Diagrams

- Example in Visio
- Example in MS Visual Studio

January 13, 2010

ISE 323: Information Systems Engineering 1

9

Relationships

- Can be expressed in UML
- Inheritance
 - Is A
 - Overlap
 - Coverage
 - One-to-one
- Contains
 - Has A
 - One-to-one, one-to-many, many-to-many
- References
 - Points To, References A
 - One-to-one, one-to-many, many-to-many

January 13, 2010

ISE 323: Information Systems Engineering 1

10

Building Class Diagrams

- Developing classes is like developing ERD
 - Classes which address the requirements
- More flexibility
 - Lots of ways
 - Makes it harder

January 13, 2010

ISE 323: Information Systems Engineering 1

11

Building Class Diagrams

- Minimality
- Structure
- Interfaces and Information Hiding
- Inheritance and Reuse
- Polymorphism

January 13, 2010

ISE 323: Information Systems Engineering 1

12

So Far

- Object Oriented Modeling and Class Diagrams
- Building Class Diagrams
- Mapping Class Diagrams to Relational Database Schemas
- Merging Objects and Functions: Introduction to FOOM

January 13, 2010

ISE 323: Information Systems Engineering 1

13

Class Diagrams to Relations

- Why map?
 - OO has won over the programming world
 - But not the DB world
- Object Oriented Database Management Systems (OODBMS)
 - Exist, but not so common
- Object-Relational Database Management Systems (ORDBMS)
 - Much more common

January 13, 2010

ISE 323: Information Systems Engineering 1

14

Mapping OO to Relations

- Challenges:
 - Objects are not "normalized"
 - Not always a clear identifier
 - Reference, containment
 - Virtual, derived properties

January 13, 2010

ISE 323: Information Systems Engineering 1

15

Mapping OO to Relations

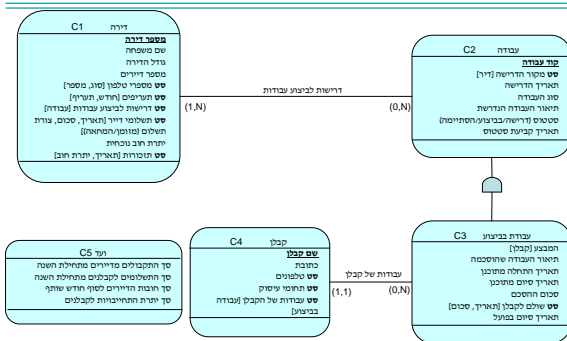
- Strategy
 - First classes with obvious key parameters
 - Then other ones without key parameters, find key parameters
 - Choose the one with the least connections first
 - Then map the key parameters to fields
 - Then go over the non-key fields and map them
 - Then connections
 - Then inherited and secondary classes

January 13, 2010

ISE 323: Information Systems Engineering 1

16

Mapping Example



January 13, 2010

ISE 323: Information Systems Engineering 1

17

Mapping Classes

- One relation per class (or so)
- Similar name

January 13, 2010

ISE 323: Information Systems Engineering 1

18

Mapping Keys

- Key attributes go to fields
- Single attribute
 - It's the primary key
- Multiple attributes
 - They are the primary key together
- Key with a relationship attribute and simple attributes
 - Together plus it's a foreign key
- Key with multiple relationship attributes and one or more simple attributes
 - Similar to the previous case
 - Each relationship is a foreign key

January 13, 2010

ISE 323: Information Systems Engineering 1

19

Mapping Other Attributes

- Simple Attributes
 - One field
- Complex group attributes
 - Each part is its own field
- Set of attributes
 - Make another relation
- Set of complex attributes
 - Find the part which identifies
 - Make it the key

January 13, 2010

ISE 323: Information Systems Engineering 1

20

Mapping Connections

- Often they are parallel (two sided)
- One-to-one
 - Add a field in one which points to the other
- One-to-many
 - Foreign key to the one on the many side
- Many-to-Many
 - Make a new relation
 - The primary key is the foreign key of both together
 - If they have complex attributes, another table
- Ternary connections
 - Already it's really one-to-many
 - We needed to make a class just for it
 - Use the one-to-many approach above

January 13, 2010

ISE 323: Information Systems Engineering 1

21

Mapping Inheritance

- A relation for each inherited class
 - Just the new attributes in the inherited
 - Many joins
- A relation for each inherited class
 - All of the attributes in the inherited
 - If there is coverage, no parent relation
- Extra fields to the parent relation
 - The non-used ones are null
- Mapping Containment is just like regular connections above

January 13, 2010

ISE 323: Information Systems Engineering 1

22

Conclusion

- Object Oriented Modeling and Class Diagrams
- Building Class Diagrams
- Mapping Class Diagrams to Relational Database Schemas

January 13, 2010

ISE 323: Information Systems Engineering 1

23