Conceptual Modeling

Isaac Griffith Idaho State University

Outcomes

Idaho State University | Computer Science

After today's lecture you will be able to:

- Why we perform the conceptual modeling step
- How to extract conceptual classes and relationships from Use Cases
- Basic ideas of Domain Analysis

Conceptual Modeling

CS 2263

Why Conceptual Models

Idaho State University | Computer Science

1. **Design Facilitation**
   - Use Cases determine the functionality
   - Design determines how to implement this functionality
   - Thus we need to setup the design phase as best as possible
2. **Added Knowledge**
   - Use Cases do not completely specify the system
   - Conceptual models can supply what is missing
3. **Error Reduction**
   - This step forces analysts to review the system
   - Additionally, we can show the results to the client for verification
4. **Useful Documentation**
   - Conceptual models can help new personnel get up to speed quickly

Extracting Classes

- The simplest approach for identifying classes is **Noun-Verb Analysis** (also called **Nomative Analysis**).
  - Verbs are used to identify operations (so for now we will ignore this).
  - Nouns (and adjectives) are useful in identifying:
    - Potential classes
    - Attributes of classes

- To use this approach:
  1. work through the text of our use cases and highlight the nouns we find.
  2. extract these nouns and analyze for synonyms, duplicates, etc.
  3. identify what they represent (attributes, classes)

Noun Analysis Example

Idaho State University | Computer Science

1. The customer fills out an application form containing the customer's name, address, and phone number and gives this to the clerk
2. The clerk issues a request to add a new member
3. The system asks for data about the new member
4. The clerk enters the data into the system
5. Reads in data and if the member can be added, generates an identification number for the member and remembers information about the member Informs the clerk if the member was added and outputs the member's name address phone and id
6. The clerk gives the user his identification number

Noun Analysis Example

Idaho State University | Computer Science

1. The **customer** fills out an **application form** containing the **customer's name**, **address**, and **phone number** and gives this to the **clerk**.
2. The **clerk** issues a **request** to add a new **member**.
3. **The system** asks for **data** about the new **member**
4. The **clerk** enters the **data** into the **system**.
5. Reads in **data**, and if the **member** can be added, generates an **identification number** for the **member** and remembers **information** about the **member**. Informs the clerk if the member was added and outputs the **member's name**, **address**, **phone**, and **id**
6. The **clerk** gives the **user** his **identification number**.

Noun Analysis Example

Idaho State University | Computer Science

- In this analysis, we found several nouns, but natural language tends to be imprecise thus duplicates and synonyms will be found
- The list is then
  - **customer** -> synonym for member
  - **user** -> synonym for member
  - **application form** and **request** -> form is external to the system and request is a menu item (so disregarded)
  - **customer's name, address**, and **phone number** -> attributes of member
  - **clerk** -> no software representation
  - **identification number** -> attribute of member
  - **data** -> data related to member
  - **information** -> data related to member
  - **system** -> the software, will be called `Library`

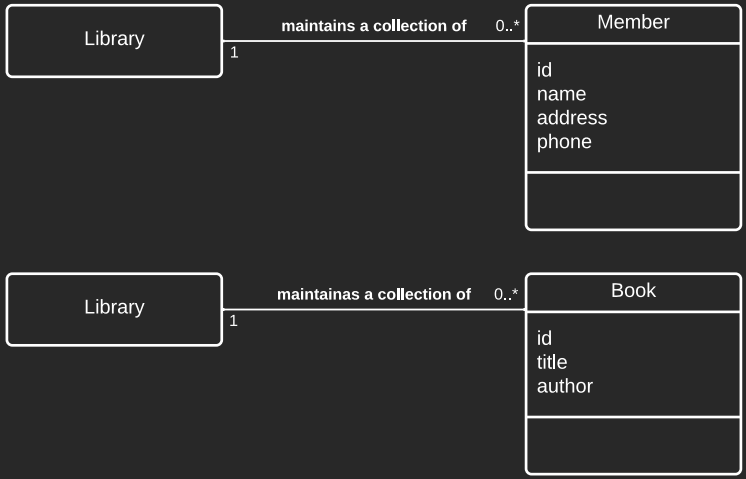
Noun Analysis Example

- The final list is:
  - **Member**
    - name
    - address
    - phone number
    - id
  - **Library**

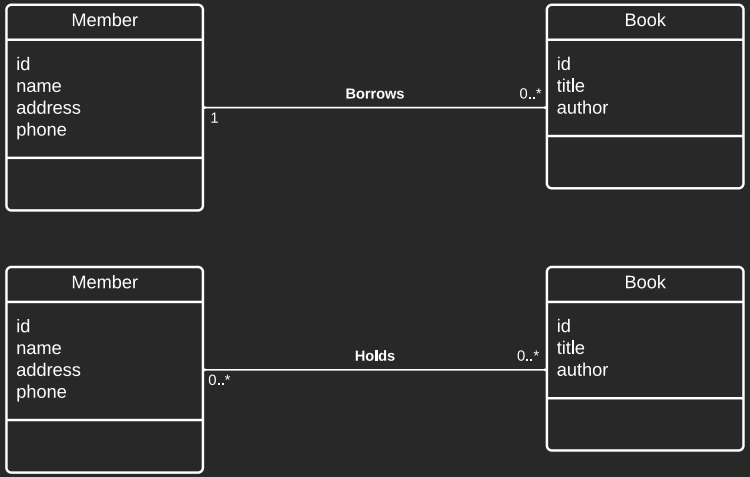
Trimming Down Classes

- From this analysis we have identified three classes key to the system:
  - Library -> the system itself
  - Book
  - Member

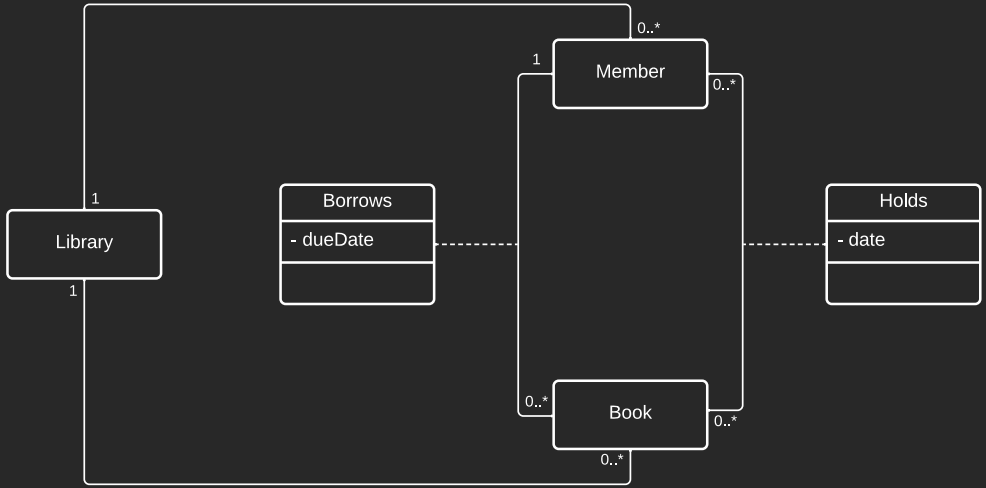
Identifying Relationships



Identifying Relationships



Final Model



Finding the Right Classes

Idaho State University | Computer Science

1. Select a Use Case
2. Analyze the selected use case
   - Extract out identified Nouns, verbs, etc.
   - Identify potential classes (but keep the initial list of nouns and synonyms)
   - Add the potential classes to the class diagram.
   - Then consider potential relationships between classes, adding them as needed.
3. If there are more Use Cases, return to step 1
4. Once you have the completed conceptual diagram, it will need to be reviewed
   - Using your analysis from each use case (that you saved), return to the Use Case and ensure:
     - there are no additional classes need that you missed
     - that you did not miss any relationships
   - For each class and relationship within the diagram
     - ensure that they really need to be there
   - Finally, consider any possible design patterns that may be applicable

Conceptual Class Guidelines

Idaho State University | Computer Science

1. In general, do not build classes around functions
   - Write a description for each class, if it start with “this class performs…”, then it should probably be merged with another class
   - Don’t name classes with verbs (it is bad form) classes are things not actions

2. Classes should more often than not have more than one method.
   - A class with a single method is probably begging to be merged with another class

3. Resist using inheritance (unless you have a predefined taxonomy already)
   - More often than not what you really want to convey is composition
   - Inheritance is for showing a relationship among well-understood abstractions

Conceptual Class Guidelines

Idaho State University | Computer Science

4. Be wary of Data Classes
   - Classes with a bunch of fields
   - No real methods only getters and setters
   - These occur for several reasons
     - Attempting to model entities external to the system
     - Encapsulating facilities, constants, or shared variables
     - Classes used to describe non-modifiable objects
5. Look for the properties of an **ideal class**:
   - clearly associated abstraction, which should be a data abstraction and not a process abstraction
   - a descriptive noun/adjective for the class name
   - a non-empty set of runtime objects
   - queries and commands
   - abstract properties that can be described as pre/post conditions and invariants

Domain Modeling

- **Domain:** Any area in which we develop software systems. Examples include
  - Library systems
  - Hotel reservation systems
  - University registration systems

- Some domains may even encompass subdomains

Domain Analysis

- **Goal:** analyze related application systems in a domain in order to discover
  - features they have in common
  - parts that are variable
  - Essentially, we identify common requirements within the domain rather than trying to solve the problem from scratch.

- **Goal:** Reuse. We want to apply what we know of similar systems to speed up:
  - specification
  - design
  - implementation

Domain Analysis

Idaho State University | Computer Science

- Domain analysis is performed prior to analysis and construction of a specific system
  - Provides knowledge about the concepts of the domain
  - Provides knowledge about competing products
  - Discovers business rules that any such system must conform to
  - Reduces necessary development time

Domain Analysis

For Next Time

- Review Chapter 6
- Review this lecture
- Read Chapter 7
- Come to Class

Are there any questions?