The Unified Modeling Language (UML)

A Standard Graphical Modeling Notation

Eddie Roache
Outline

- Why Use UML
- History
- UML Characteristics
- Diagram Types
- Use Case Diagrams
- Class Diagrams
Why Use UML?

- Help analyze complex domains
- Help design complex systems
- Visualize analysis and design artifacts
- Clearly document development artifacts
- Is simple, yet expressive
- Can be applied to different processes
Some UML Examples

Some UML Examples

Some UML Examples

Supply Customer Data

Order Product

Arrange Payment

«include»

«include»

«include»

Place Order

Extension points
additional requests:
after creation of the order

1 *

the salesperson asks for
the catalog

Request Catalog

History

- OO takes a foothold
- New OO modeling notations spring up
- Three amigos get together
  - Grady Booch
  - Ivar Jacobson
  - James Rumbaugh
- UML 0.8 is born
UML Characteristics

- Unified Modeling Language
- Standard graphical modeling notation
- Supported by formal semantics
- Current version: 2.0
- Wide use and acceptance in the IT industry
- Large tool support
  - Rational Rose, Together, Visio, Jude, …
  - … although no tool fully supports the standard
- Is not a process
- Process independent
Diagram Types

- Structure diagrams
  - Class diagram
  - Object diagram
  - Component diagram
  - Deployment diagram
  - Package diagram
  - Composite Structure diagram

- Behavior diagrams
  - Use Case diagram
  - Interaction diagrams
    - Sequence diagram
    - Interaction Overview diagram
    - Communications diagram
    - Timing diagram
  - State Machine diagram
  - Activity diagram
Diagram Types

Use Case Diagrams

- Model user interaction with system
- Capture functional requirements
- Also used for
  - business modeling
  - component specification
- UML Spec does not include Use Case Specs
Use Case Definition

“A use case specifies a sequence of actions, including variants, that the system can perform and that yields an observable result of value to a particular actor.”

“The Unified Software Development Process”, Ivar Jacobsen*, Grady Booch, Jim Rumbaugh

*Author of “Object-Oriented Software Engineering: A Use Case Driven Approach”
Use Case Specification

**Name:** Create AddressEntry

**Description:** This use case allows the actor to create a new entry for an Address Book.

**Preconditions:**
1. An Address Book is open.

**Steps:**
1. The actor requests to create a new AddressEntry.
2. The system creates a new AddressEntry and returns it to the actor.

**Post Conditions:**
1. An Address Book Entry is created.

See “Writing Effective Use Cases” and “http://alistair.cockburn.us/usecases/usecases.html”
Use Case Diagrams

- Actor
- Use Case
- Association
- Dependency
  - extend
  - include
- Generalization
- Note (Comment)

Close AddressBook

This is a note.
Address Book Example

- The Address Book System provides distributed access to a set of address books.
  - An Address Book is made up of Address Book Entries
  - An Address Book Entry contains a name, street address, phone number, email address

- The Address Book System shall:
  - allow users to create, view, delete, and save address books
  - allow users to create, modify, and delete entries in an address book
Use Case Diagram

Address Book User

View AddressBook

Add New Address

Update Address

Delete AddressEntry

Create AddressEntry

Add AddressEntry

Create PersonAddressEntry

Create OrgAddressEntry

Save AddressBook

Close AddressBook

All dependencies are extend unless stereotyped otherwise
Class Diagrams

- Model system classes, interfaces, and class relationships
- Capture structural (vs. behavioral) info

Used for
- business domain modeling
- logical design
- implementation design
Class Diagrams

- Class
- Abstract Class
- Class with
  - Attributes
  - and
  - Operations

AddressBook

AddressBook

AddressBook

- id
- +name
- +phone

+getName()
+getPhone()
Class Diagrams

- Dependency
- Association
- Navigability
- Aggregation
- Composition
- Generalization
Class Diagrams

- Multiplicity
- Constraint
- Role
- Name
- Directionality
- Nesting
Class Diagrams

- Stereotype
- Realization
- Realization
- Derived Assoc.
- Association Class
- Note (Comment)
Class Model Element

Window

size: Area
visibility: Boolean

display ()
hide ()

Window

{abstract,
author=Joe,
status=tested}

+default-size: Rectangle
#maximum-size: Rectangle
-xptr: XWindow*

+display ()
+hide ()
+create ()
-attachXWindow(xwin:XWindow*)

## Class Model Element

<table>
<thead>
<tr>
<th>Reservation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>operations</strong></td>
</tr>
<tr>
<td>guarantee()</td>
</tr>
<tr>
<td>cancel ()</td>
</tr>
<tr>
<td>change (newDate: Date)</td>
</tr>
<tr>
<td><strong>responsibilities</strong></td>
</tr>
<tr>
<td>bill no-shows</td>
</tr>
<tr>
<td>match to available rooms</td>
</tr>
<tr>
<td><strong>exceptions</strong></td>
</tr>
<tr>
<td>invalid credit card</td>
</tr>
</tbody>
</table>

Address Book Domain

AddressBook

AddressEntry

* 1 1

Party

Person Organization
Logical Design

AddressBookView  <->  AddressBookController

AddressBook
Implementation Design

AddressBookView
- addrBookFrame: JFrame
- addrList: JList
- entryNames: String[]

+ show()
+ update()
# createAddrBookFrame(): JFrame

AddressBookController
+ getEntryNames(): String[]
+ createNewEntry(): AddressBookEntry
~ removeEntry()
+ findEntry(name: String): AddressBookEntry

AddressBookEntry
# name: String
# streetAddress: String
# city: String
# State: String
# Zip: String
# phoneNumber: String
# emailAddress: String

+ toString(): String

AddressBook
- name: String

+ load()
+ getEntryNames(): String[]
+ addEntry()
+ removeEntry()
+ findEntry(name: String): AddressBookEntry
More UML Characteristics

- Supported by formal semantics (meta-model)
  - Meta-Object Facility Specification
- UML Model Interchange using XML Metadata Interchange (XMI)
- Object Constraint Language (OCL)
- Action Semantics UML Extensions
- Standard profiles
Tools

- Paper and Pencil
- Any drawing program (preferably vector-based)
- Rational Rose (and XDE)
- Together
- Jude
- Poseidon
- ArgoUML
- StarUML
- Many others . . .
References

- www.omg.org
- www.uml.org
- www.rational.com
- www.martinfowler.com/books.html#uml
- www.oreilly.com/catalog/umlNut/
- www.oreilly.com/catalog/umlpr/
- www.oreilly.com/catalog/learnuml/
- The Unified Modeling Language User Guide
- http://alistair.cockburn.us/usecases/usecases.html
Questions
Class Diagram

Fruit

Seed

Apple
Orange
Pear

Granny Smith
Red Delicious
Rome

*
Class Diagram