1

An Introduction to UML

Presenter: Josh Prowant

Purdue University Fort Wayne Campus
CPET 545 Service Oriented Architecture & Enterprise Applications
September 18, 2008

2

Outline

- Introduction
- Modeling Language Introduction
- UML Purpose/Benefits
- Fundamentals of UML Diagrams
- Models and Diagrams
- General Examples
- UML Process

3

Introduction

- Unified Modeling Language (UML)
  - Standard for software and systems development
  - High level of abstraction allows focus on important aspects of system's design
  - Ensures business functionality is complete and correct, end-user needs met, technical requirements met before expensive changes
  - System-independent

4

Modeling Language

- Helps describe a system
  - Notation: elements that make up a modeling language
  - Semantics: descriptions of what notation means
- Source code
  - Too detailed and not understood by common stakeholder
- Informal language
  - Room for interpretation

5

Why UML?

- Formal language
  - Every element has a strongly defined meaning
- Concise
  - Simple and straightforward notation
- Comprehensive
  - Describes all important aspects of a system
- Scalable
  - Can handle massive or small-scale projects
- Built on lessons learned
  - Culmination of best practices of Object Oriented community
- The Standard
  - Transformability and interoperability

6

Fundamentals of UML Diagrams

- *Notes* for additional comments
- *Stereotypes* for special use or intent (elements role)
  - Typically has an associated icon (actor symbol)
  - Guillemets

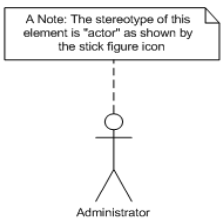
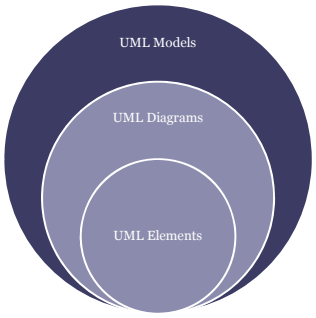


Figure 1: UML Note and Stereotype

Figure 2: A Stereotype using guillemets

7

Models and Diagrams



- UML as a sketch – convey key points
- UML as a blueprint – detailed specification of a system with UML diagrams
- UML as a programming language – UML model to executable code

8

Models and Diagrams

- Structure Diagrams
  - Class, Object, Component, Composite Structure, Package, Deployment
- Behavior Diagrams
  - Use Case, Activity, State Machine
- Interaction Diagrams
  - Sequence, Communication, Timing, Interaction Overview

9

Models and Diagrams

Table 1: UML 2.0 Diagrams [1]

Diagram type	What can be modeled?
Use Case	Interactions between the system and users or other external systems. Also helpful in mapping requirements to your systems.
Activity	Sequential and parallel activities within the system.
Class	Classes, types, interfaces, and the relationships between them.
Object	Object instances of the classes defined in class diagrams in configurations that are important to the system.
Sequence	Interactions between objects where the order of the interactions is important.
Communication	The ways in which objects interact and the connections that are needed to support that interaction.
Timing	Interactions between objects where timing is an important concern.
Interaction Overview	Used to collect sequence, communication, and timing diagrams together to capture an important interaction that occurs within the system.
Composite Structure	The internals of a class or component, and can describe class relationships within a given context.
Component	Important components within the system and the interfaces they use to interact with each other.
Package	The hierarchical organization of groups of classes and components.
State Machine	The state of an object throughout its lifetime and the events that can change that state.
Deployment	How the system is finally deployed in a given real-world situation.

10

Class Diagram

- UML models consist of objects that interact by sending messages
  - Objects have things they know (attributes) and things they can do (behaviors or operations)
  - Objects are instances of classes

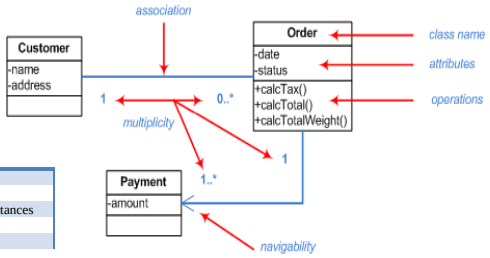


Table 2: Common multiplicities [3]

Multiplicities	Meaning
0..1	zero or one instance
0..* or *	no limit on the number of instances
1	exactly one instance
1..*	at least one instance

Figure 3: Class Diagram

11

Object Diagram

- Much like class diagram
- Name of specific instance of class on the left side of a colon; name of class on the right side of the colon
- Helpful for complicated relationships between instances of classes

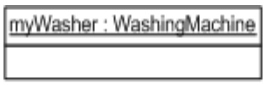


Figure 4: A UML Object

12

Package Diagram

- To simplify complex class diagrams, classes can be grouped into packages

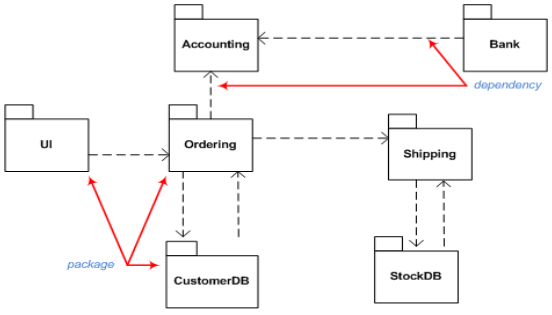


Figure 5: Package Diagram

13

Component and Deployment Diagrams

- Component (code module) diagrams are physical analogs of class diagrams
- Deployment diagrams show the physical configurations of software and hardware (nodes)

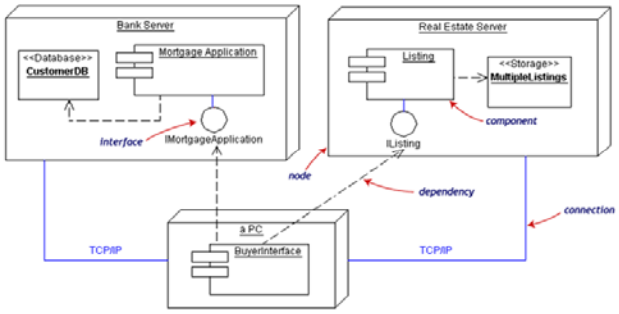


Figure 6: Component and Deployment Diagram [3]

14

Use Case Diagram

- Describe system's behavior from standpoint of external user
- Actor can be person or another system

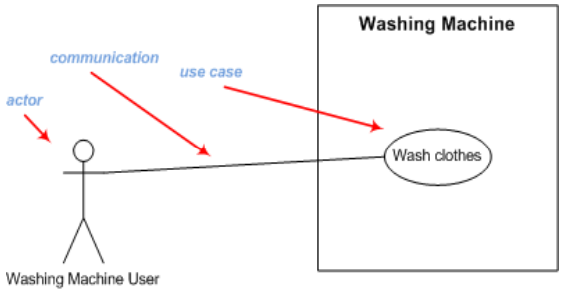


Figure 7: Use Case Diagram

15

State Machine Diagram

- State of an object depends on its current activity or condition (rounded rectangles)
- State machine diagram shows states of an object and transitions that cause a change in state

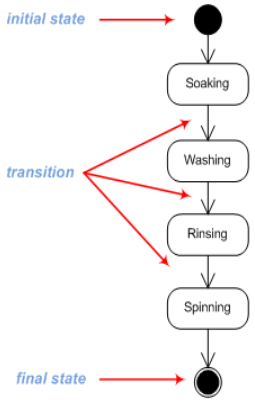


Figure 8: State Machine Diagram

16

Activity Diagram

- Similar to a flowchart
- Shows activities involved in a single process (state diagram shows process itself)

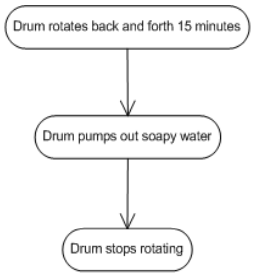


Figure 9: Activity Diagram representing "washing" process

17

Sequence and Communication Diagrams

- Interactions among objects – interaction diagrams
- Sequence Diagram
  - How operations are carried out (messages sent)
  - Organized by time and progresses from top to bottom
- Communication Diagram
  - Focus on object roles instead of the times that messages are sent
  - Sequence number shows order of messages

18

Sequence Diagram

- Operation of a washing machine
  - Lifeline: time that an object exists
  - Activation bar: duration of the execution message

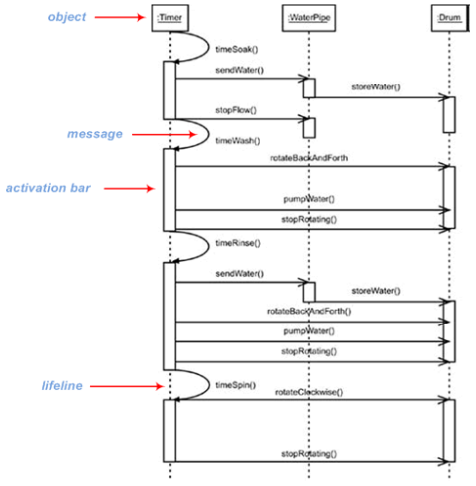


Figure 10: Sequence Diagram

19

Communication Diagram

- Messages among timer, water pipe, and drum of washing machine example

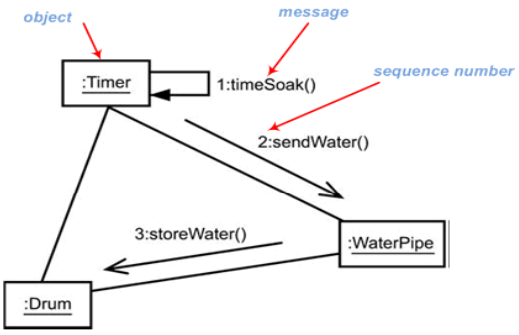


Figure 11: Communication Diagram

20

UML Process

- Object Management Group
  1. Select a methodology (process used to gather and analyze requirements and design application to meet them)
  2. Select a UML Development Tool (aligned with a methodology)
  3. Get training

References

- [1] "Learning UML 2.0," by Russ Miles and Kim Hamilton, ISBN 0-596-00982-8, O'Reilly Media, Inc., 2006.
- [2] "Introduction to OMG's Unified Modeling Language (UML)", Object Management Group, July 2005, http://www.omg.org/gettingstarted/what_is_uml.htm.
- [3] "Practical UML: A Hands-On Introduction for Developers," by Randy Miller, Dec. 2003, http://dn.codegear.com/article/31863.
- [4] "Sams Teach Yourself UML in 24 Hours," 3rd edition, by Joseph Schmuller, ISBN 0-672-32640-X, Sams Publishing, 2004.