Intro to UML
(materials herein excerpted from Seeing the Forest in the Midst of the Trees)

What is happening in the software world
The Good News…

“26% of software projects succeed.”

Standish Group, CHAOS Report, 2000

The Bad News…

That means 74% failed!

Standish Group, CHAOS Report, 2000
Software Development is Complex

- Poorly designed project architectures require untimely changes
- Requirements are undefined or change mid-project
- Discovering defects late in project or flaws in architecture and design
- Lack of communication between disparate team members
- Artifacts are not accessible to all team members

Poor Management = CHAOS

How To Make Sure Your Project will Fail

- Lack of user input
- Unclear objectives
- Incomplete requirements and specifications
- Changing requirements and specifications
- Lack of planning
### Necessity of Communication

- Think of a 100 man-person team
  - Analysts, developers, QE, documentation, contractors
  - Marketing, product management, VPs
- Geographically dispersed
  - Different offices
  - Different countries
  - Different time zones
- Requirements change or priorities are rearranged
- Different sub-systems are developed at different times

**Number of communication paths increases by the square of the team size**

---

### The Software Development Paradox

Faster Time to Market

Higher Quality

**Internet time :(**

**Now do it with less ...**
The Software Effort Breakdown

- Over the life of a product, the distribution of effort is:
  - 30% development
  - 70% maintenance

- Development
  - 40% analysis & design
  - 20% implementation
  - 40% validation

- Maintenance
  - 20% adaptive
  - 60% perfective
  - 20% corrective

What is Missing

- Need a common language that unifies the different stake holders
- Different stake holders have different software abstractions (models) and artifacts
- We need ....
Communication Using the Unified Modeling Language

One language – One tool – One team

Who Should Model?

Requirements and Business Models

Business Analyst

C++

Java

SW Models

Data Models

Database

Everyone!

HTML

CGI

XML

Web Content

Developer
The Developer's View

The Model is
The Application

Host or Target Application

Sequence Diagram
Class Diagram
Use Case Diagram

Structure Diagram
Behavior Diagram
Component Diagram
Deployment Diagram

The Unified Modeling Language

The Unified Modeling Language
UML History
- 1994: Grady Booch and Jim Rumbaugh began unifying their modeling techniques at Rational Software
- 1995: Ivar Jacobson joins team at Rational
- 1996: Consortium of 12 companies formed to oversee UML
- Jan 1997: Version 1.0 published
- Sept 1997: Revised Version 1.1
- Nov 1997: Object Management Group standardized
- Version 1.4 just accepted
- Working on version 2.0

Why is the Word “Model” Important?
- Developing software is about developing executable abstractions
- An abstraction or view is a model
  - For example, a class is an abstraction of a real-world entity or concept
- Different stake holders have different abstractions
  - Marketing has the feature sheet
  - Developers have the requirements
  - Testing have test cases and configurations
- There are model types in building a system
UML Context

- It enables and promotes a
  - use-case-driven
  - architecture-centric
  - iterative
  - incremental process that is
  - object oriented and component based

- Justification is that
  - Use cases are used to manage and provide focus for a problem-solving effort.
  - Architecture is used to manage complexity and maintain integrity and focus as a solution to a problem evolves.
  - Iterations and increments are used to repeatedly apply a process to evolve a solution to a problem.

Why is UML So Great?

- Combines best ideas from software engineering, database theory, and system design
- Technology agnostic
- Problem domain agnostic
  - Extensibility mechanisms allow tailoring to the domain
- Scalable
  - Recursive, hierarchical decomposition
- Bootstrapping principle
  - Language that can define itself
- High information density
  - Visual
  - Packs a lot into a small space
**UML Models**

- Models capture
  - the structural, or static, features of systems
  - the behavioral, or dynamic, features of systems.
- Models have several independent dimensions
  - Each emphasize particular qualities of a model
  - Each dimension has a diagram type

**UML Diagrams**

- Use case diagrams depict the functionality of a system.
- Class and object diagrams for the static structure
- Sequence (collaboration) diagrams for behavior in a scenario
- State diagrams for execution
- Activity diagrams for process descriptions
- Component diagrams for dependencies between components
- Deployment diagrams for configuration and environment
Other Elements of UML

- There are many
  - Package, sub-system, class, classifier, interface, ...
- We really don’t have the time to discuss this
- Talk to your professors
- There are many good books around

USE CASEs

- Describes the proposed functionality of a system
- Represent functional requirement
- Notation
  - Use cases: ellipse with action phase
  - Actors is a user of the system or other systems
Logical Model

- Class and Object Diagrams

- Class Diagram Notation
  - 3-compartment rectangle
  - Relationship among classes

- Object diagram: instance of a class

- Accessibility Notation

Logical Model (continued)

- Class and Object Relationship
  - Inheritance: generally describes the hierarchical relationship between classes (family tree)
Logical Model (continued)

- Class and Object Relationship
  - Association: generally relate to one object having an instance of another as an attribute or owning.

Sequence Diagrams (dynamic relationship)

- illustrates this message passing and the sequence in which it occurs normally within a given usecase
Logical Model (continued)

- Class and Object Relationship
  - Aggregation: generally define whole/part relationships.

Some materials herein are excerpted from *The Logical Model* by Geoffrey Sparks

Cool Things to do with UML
Do all of this for Multiple Languages

- UML models can be targeted for different languages
  - Java
  - Microsoft Visual C++
  - Microsoft Visual Basic
  - ANSI C++
  - Ada
  - IDL
  - XML-DTD
  - SQL

Keeping the Model and Code Synchronized

- Manual model and code synchronization
  - On-demand synchronization
  - Complete control as updates occur

- Auto synchronization
  - Source is updated when model is modified
  - Rational Rose model updated when source is modified
**Unit Test Functionality**

- Generate test code directly from model
- Provide test data and expected results

![Diagram of test generation process](image)

**System Test Functionality**

- Automatically generate code for component testing from a UML model
- Enable scenario-based testing during component integration, before system is complete

![Diagram of system test functionality](image)
Code Templates For Architecture Design

- Ready made design and code solutions for common development tasks
  - COM, MFC, ATL
  - MTS, ADO
  - ASP, DHTML

- Fully customizable
  - You can create your own code templates to automate common design and implementation tasks to ensure consistency in both design and code.

Frameworks For Architecture Definition

- Frameworks: Predefined model element sets for modeling specific systems
- Used to:
  - Define the architecture of specific types of systems
  - Provide a set of reusable components
  - Create templates for new models
- Simplify development with commercial frameworks
- Promote reuse and standards with custom user frameworks
Robust Development Using Proven Patterns

- Develop your application using predefined industry recognized patterns:
  - Apply patterns to existing model elements
  - Create new model elements automatically via patterns
  - Leverage proven designs

UML Model Debugging

Rational Rose RealTime
Distributed UML Designs

- Enables deployment and visualization of distributed applications
- Supports patterns for creating high-availability applications
- Provides the distributed communication infrastructure

That’s all
Some Important Web Sites

- The SEEDS program will let your college get Rose
  - http://www.rational.com/corpinfo/college_relations/seed/terms_cond.jsp
- .NET development
- Java development
  - www.jroundup.com
- Project management
  - www.ganthead.com

Thank You
### Rational: Ongoing Leadership

<table>
<thead>
<tr>
<th>Category</th>
<th>Leader</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.1 in Visual Modeling, 4 years running¹</td>
<td>Rational Rose</td>
<td>&quot;...the battle for dominance is over: Rational wins.&quot; Ed Yourdon</td>
</tr>
<tr>
<td>No.1 in SCM, 3 years running¹</td>
<td>Rational ClearCase</td>
<td>&quot;ClearCase is the dominant SCM tool.&quot; Ovum</td>
</tr>
<tr>
<td>No.1 in Requirements Management²</td>
<td>Rational RequisitePro</td>
<td>&quot;Easy-to-use...ideal for team based development...&quot; InfoWorld</td>
</tr>
<tr>
<td>Real-time embedded leadership</td>
<td>Rational Rose RealTime</td>
<td>&quot;...a major contender as the de facto standard for real-time embedded...&quot; IDC</td>
</tr>
<tr>
<td>Driving Standards in Best Practices</td>
<td>UML, WebDAV</td>
<td>&quot;...the company that put the 'unified' in modeling languages...&quot; JavaPro</td>
</tr>
</tbody>
</table>

¹IDC, ²Standish