


Object-Oriented Development

04-OO-Development 1




The OO Solution

- The problem domain is relatively constant
 - Creating cards
 - Assemble the card and get the right thing at the right place
 - Auto pilot
 - Get a plane from point A to point B using the available control surfaces
- The functionality and data representation is what is likely to change
 - Creating cards
 - The type of information and placement of information changes often
 - The options available to the user evolve with time
 - Auto pilot
 - The hardware interfaces are different between different models
 - The operational modes vary between models and evolve over time

Structure the system based on the structure of the problem domain, NOT based on the structure of the solution


04-OO-Development 2



What is OO

- A way of thinking about a problem (software) based on abstractions of concepts that exist in the real world
- OO is not constrained by implementation language (C, Pascal, FORTRAN , etc. will work)

04-OO-Development 3




What is not OO

- Using an "object oriented" language (C++, Eiffel, Smalltalk)
 - You can easily misuse the OO support in these languages
- Using an "OO notation" for design
 - Misuse the notation for a non-OO design
- Calling what you do OO
 - Management and customers like OO, therefore, that is what we are doing

OO is not the answer to all your problems


04-OO-Development 4



Several Complementary Models


- Structural Models
 - Describes the structure of the objects in a system
 - Structure of individual objects (attributes and operations)
 - Relationships between the objects (inheritance, sharing, and associations)
 - Clustering of objects in logical packages and on the actual hardware
- Dynamic models (behavioral models)
 - The aspects related to sequencing of operations
 - Changes to attributes and sequences of changes
 - The control aspects of the system

04-OO-Development 5




Intentionally Blank

04-OO-Development 6



The OO Development Process


04-OO-Development 7



We Will Cover

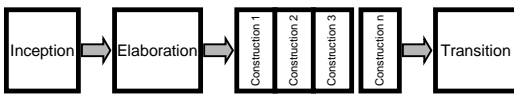
- Requirements specification
 - Very briefly
- Iterative development
- Different models
 - Three distinct models for which you can use UML
 - ◆ Domain (or conceptual) model
 - ◆ Analysis (specification) model
 - ◆ Design (implementation) model
- How do we move between the models

04-OO-Development 8




Process Overview

- Inception
- Elaboration
- Construction
 - Many iterations
- Transition




04-OO-Development 9



Inception

- Creation of the basic idea that we want to implement (presumably with software)
- Could take many shapes
 - A discussion over a beer at the pub
 - A full fledged feasibility study
- Figure out (roughly)
 - The business case
 - ◆ How much money will this make the company
 - Project scope


04-OO-Development 10



Elaboration

- Answer the following:
 - What is it you are going to build?
 - How are you going to build it?
 - What technology are you going to use?
- Your decisions should be guided by the risks
 - Requirements risks
 - Technological risks
 - Skills risks
 - Political risks

04-OO-Development 11



Requirements Risks

- Poor or wrong requirements a serious problem
- Use UML notations to help you understand the customers requirements and the inherent structure of the problem domain
 - Use-case diagrams and use cases to understand customer requirements
 - Class diagrams, activity diagrams, and possibly other diagrams to understand the domain

04-OO-Development 12



Plan the Construction Phase

- We will never build the entire system at once
 - Incremental development
- Categorize the use cases
 - "I absolutely must have this function in the system"
 - "I can live without this feature for a little while"
 - "This is an important function, but we might be able to live without it"
- Time estimate and allocate the use cases to iterations

04-OO-Development

13



Construction

- Construct the system as a series of iterations
 - Each iteration is a "mini" project
 - ◆ Analyze the use case, design, code, test, and integrate.
- Refine your domain model
 - Identify all attributes and operations
 - Define the dynamic behavior of all objects
 - ◆ State machines
 - ◆ "Contracts"
- Make decisions influenced by platform and language

04-OO-Development

14



Transition

- The phase between the beta release and the final product
- Wrap up all the issues that should not be done or cannot be done during the iterations
 - Examples include performance evaluation and optimization
 - Complete system testing
- No new functionality added
 - Fix bugs
 - Refactor your system a final time

04-OO-Development

15



Three Distinct Models

- A conceptual model (domain model)
 - Try to figure out what is really going on
 - Build a model to better understand the problem
 - Used to communicate with the customer and "domain" experts
- An analysis model (specification model)
 - Model the software that will implement the system
 - Focus on the software structure and the module interfaces
- Design model (implementation model)
 - A detailed design of the software
 - Including all attributes and detailed descriptions of the operations

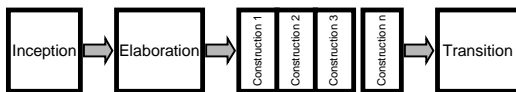
04-OO-Development

16



Summary

- Inception
- Elaboration
- Construction
 - Many iterations
- Transition



04-OO-Development

17