

Analysis and Design - Lecture Outline

- ◆ **Introducing the crystallization analogy**
 - A lattice of information
 - Stages in the Crystallization Process
- ◆ Understanding the analysis and design discipline
- ◆ Defining the Analysis and Design Activities
- ◆ Documenting the Analysis and Design Discipline
- ◆ Viewing the Model

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-1

Process & Life-cycle Viewpoint

Development Cycle Phases

| Engineering Disciplines | Inception | Elaboration | Construction | Transition |
|-------------------------------|-----------|-------------|--------------|------------|
| Requirements | High | Medium | Low | Very Low |
| Analysis & Design | Low | High | Medium | Low |
| Implementation | Very Low | Low | High | Medium |
| Test | Very Low | Low | Medium | High |
| Supporting Disciplines | | | | |
| Configuration & Change Mgmt | Low | Medium | High | Very High |
| Project Management | High | Medium | Low | Very Low |

Principal iteration for the discipline

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-2

Analysis and Design - Lecture Outline

- ◆ Introducing the crystallization analogy
- ◆ **Understanding the analysis and design discipline**
 - The concepts behind the analysis and design activities
 - The Quality of the analysis and design activities
 - The Designer Role
- ◆ Defining the Analysis and Design Activities
- ◆ Documenting the Analysis and Design Discipline
- ◆ Viewing the Model

UPEDU Concept: Analysis Mechanisms

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-3

Analysis and Design Activities

Analysis blends with design

Analysis precedes design

Analysis mixes with design

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-4

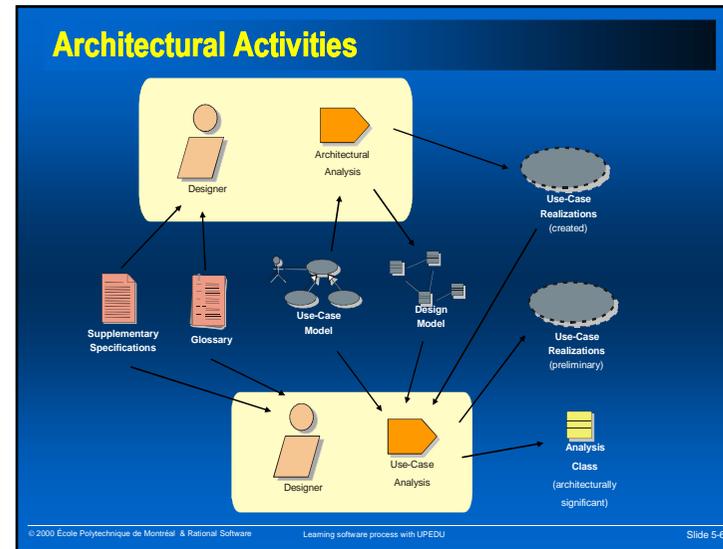
Analysis and Design - Lecture Outline

- ◆ Introducing the crystallization analogy
- ◆ Understanding the analysis and design discipline

- ◆ **Defining the Analysis and Design Activities**
 - Defining the Architecture
 - Growing the Design
 - Reviewing Architecture and Design

- ◆ Documenting the Analysis and Design Discipline
- ◆ Viewing the Model

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-5



Benefits to a Good Architecture

- ◆ Architecture lets developers gain and retain **intellectual control** over a project, to manage its complexity, and to maintain system integrity
- ◆ Architecture provides an effective basis for **large-scale reuse**
- ◆ Architecture provides a basis for project **management**
- ◆ Architecture facilitates **component-based development**
 - A component fulfills a clear **function** in the context of a well-defined architecture
 - A component conforms to and provides the physical **realization** of a set of interfaces
 - Components exist relative to a given **architecture**

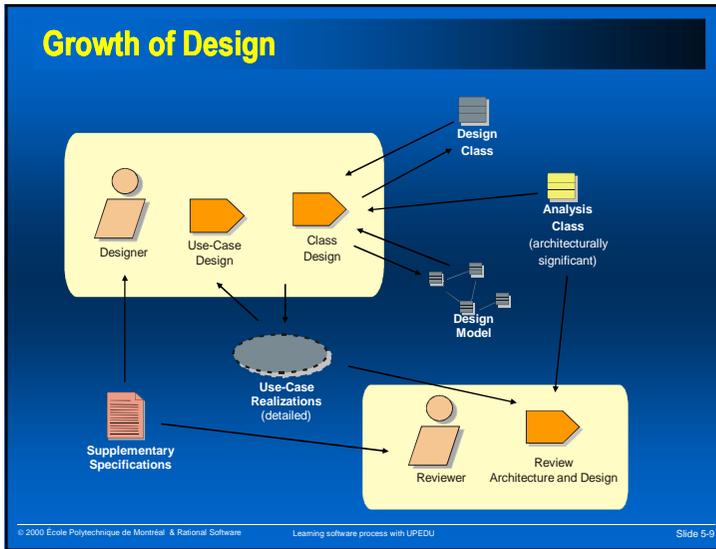
UPEDU Concept: Software Architecture

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-7

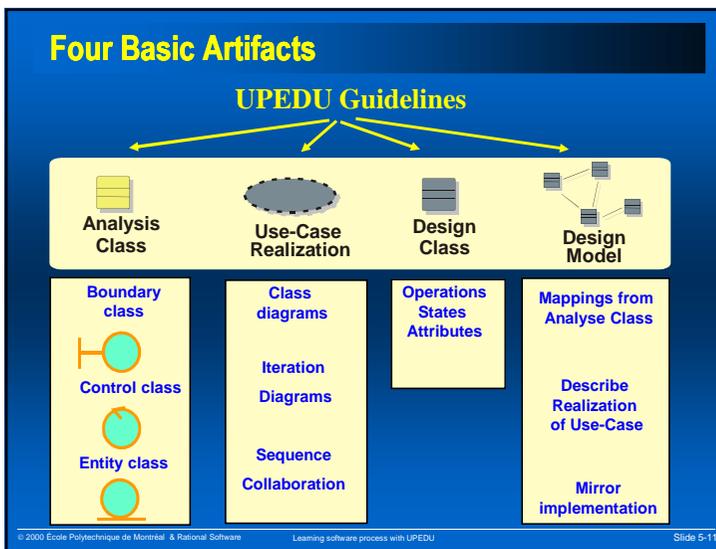
System's Reuse and Resilience

- ◆ Good architectures meet their **requirements**, are **resilient**, and are **component-based**
- ◆ A **resilient architecture enables**
 - Improved maintainability and extensibility
 - Economically-significant reuse
 - Clean division of work among teams of developers
 - Encapsulation of hardware and system dependencies
- ◆ A **component-based architecture permits**
 - Reuse or customization of existing components
 - Choice of thousands of commercially-available components
 - Incremental evolution of existing software

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-8



- ### Analysis and Design - Lecture Outline
- ◆ Introducing the crystallization analogy
 - ◆ Understanding the analysis and design discipline
 - ◆ Defining the Analysis and Design Activities
- ◆ Documenting the Analysis and Design Discipline
 - Artifacts
 - The Analysis Classes
 - Use-Case Realization
 - Design Class
 - The Design Model
- ◆ Viewing the Model
- © 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-10



- ### Analysis and Design - Lecture Outline
- ◆ Introducing the crystallization analogy
 - ◆ Understanding the analysis and design discipline
 - ◆ Defining the Analysis and Design Activities
 - ◆ Documenting the Analysis and Design Discipline
- ◆ Viewing the Model
 - Views
 - The Use-Case View
 - The Logical View
 - The Implementation View
- © 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-12

2+1 Complementary Views

The diagram illustrates three interconnected views of a software system. At the top left is the **Logical View**, associated with **Analysts/Designers Structure**. In the center is the **Use-Case View**, associated with **End-user Functionality**. At the bottom right is the **Implementation View**, associated with **Programmers Software management**. The Use-Case View is highlighted with a yellow oval, indicating its central role in connecting the other two views.

© 2000 Ecole Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-13

Use Case View

- ◆ **Describes:**
 - an architecturally significant subset of the use-case model.
- ◆ **Concerns:**
 - functionality, critical functions, performance
- ◆ **Represents:**
 - graphically, on Use Case Diagrams

The diagram shows a computer monitor on the left with an arrow pointing to a Use Case Diagram on the right. The diagram includes a stick figure actor connected to two use cases. To the right of the diagram, the text reads: **• Actors • Use Cases**.

UPEDU Concept USE-CASE View

© 2000 Ecole Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-14

Logical View

- ◆ **Describes:**
 - an architectural subset of the design model, (a subset of the classes and use-case realizations).
- ◆ **Concerns:**
 - functionality, behavior, use of frameworks, patterns
- ◆ **Represents:**
 - graphically, on Class Diagrams, Interaction Diagrams and State Diagrams

The diagram shows a computer monitor on the left with an arrow pointing to a Class Diagram on the right. The diagram shows several classes connected by lines. To the right of the diagram, the text reads: **• Package structure • Interesting classes • Use Case Realizations**.

UPEDU Concept: Logical View

© 2000 Ecole Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-15

Implementation View

- ◆ **Describes:**
 - the software component organization in the development environment
- ◆ **Concerns:**
 - ease of development, team organization, inclusion of existing systems or components, software configuration and management
- ◆ **Represents:**
 - graphically on Component Diagrams

The diagram shows a computer monitor on the left with an arrow pointing to a Component Diagram on the right. The diagram shows several components connected by lines. To the right of the diagram, the text reads: **• Code library structure • Source code Deliverables (.exe , .DLL, Data files)**.

UPEDU Concept: Implementation View

© 2000 Ecole Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 5-16