

Programming, Debugging and Unit Testing

- ◆ **Programming**
 - Translating design into code
 - Related to programming language competence
- ◆ **Debugging**
 - Removing programming faults
 - Related to workmanship
- ◆ **Unit testing**
 - Removing translation faults
 - Related design comprehension

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 6-7

Approaches to Unit Testing

White box

Black box

Performance

Reliability

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 6-8

Several Techniques for Reviewing Code

- ◆ **Code inspection**
 - Formal evaluation technique
 - Team consensus on code quality
- ◆ **Walkthrough**
 - Author leads reviewers through the code
 - Cognitive synchronization of team members
- ◆ **Code reading**
 - Individual read the code according to standard practices
 - Code compliance to programming guide

UPEDU: Work GUIDELINE: REVIEW

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 6-9

Several Techniques for Reviewing Code

- ◆ **Code inspection**
 - Formal evaluation technique
 - Team consensus on code quality
- ◆ **Walkthrough**
 - Author leads reviewers through the code
 - Cognitive synchronization of team members
- ◆ **Code reading**
 - Individual read the code according to standard practices
 - Code compliance to programming guide

UPEDU: Work GUIDELINE: REVIEW

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 6-10

Components are Divided into Two Types

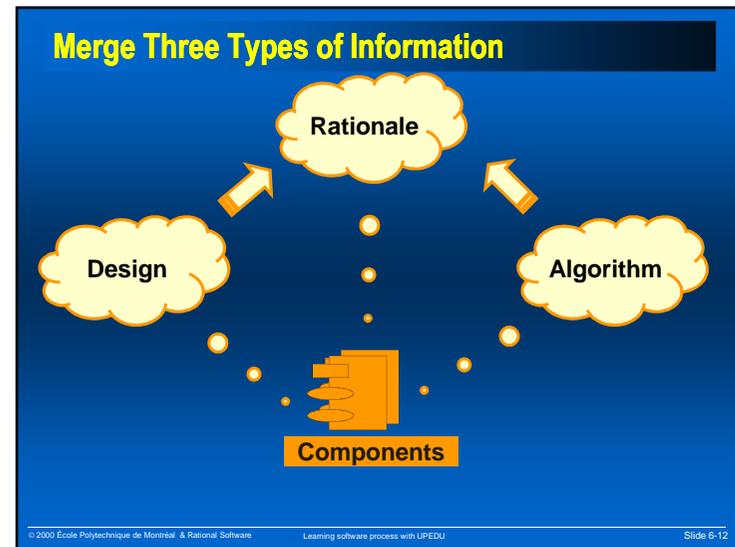
Deliverables

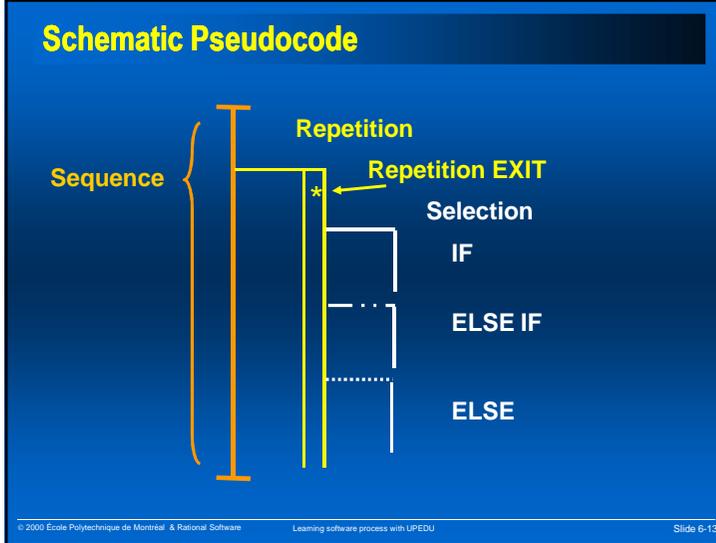
| | |
|------------------------|----------------------|
| Executables | .exe files |
| Load libraries | .dll files |
| Applets | .class for Java |
| Web pages | .htm and .html files |
| Database tables | |

Deliverables produced from components

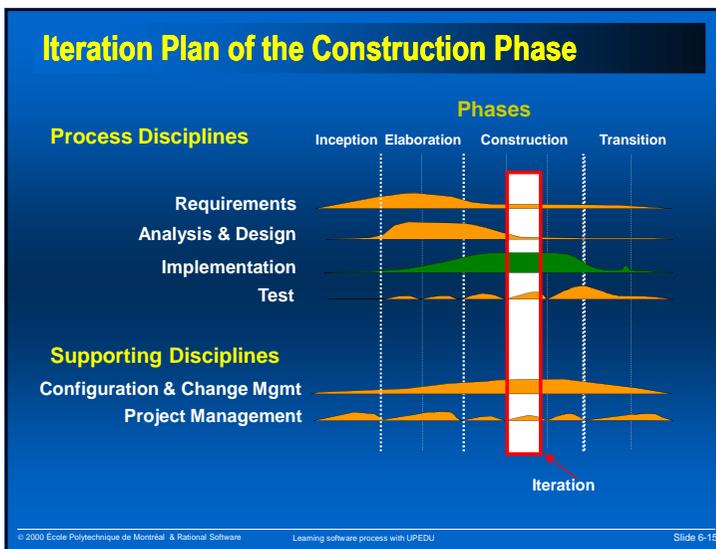
| | |
|--------------------------|----------------------------------|
| Source code files | .h, .cpp and .hpp files for C++ |
| Binary files | .o and .a files linked into exec |
| Build files | makefiles |

© 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 6-11





- ### Programming Guidelines
- ◆ Establish under the responsibility of the organization
 - ◆ Adapt to the organizational needs
 - ◆ Define level of source code documentation
 - ◆ Specify naming convention for file and variable names
 - ◆ Explain restricted use of programming features
 - ◆ Create iteratively in collaboration with the team
- UPEDU Concept: Mapping from design to Code**
UPEDU Guideline: Programming Guide
- © 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 6-14



- ### Artifacts of the Construction Phase
- ◆ **Software Development Plan**
 - (updated for the iteration)
 - ◆ **Use-Case Model**
 - (updated)
 - ◆ **Design Model**
 - ◆ **Implementation Model**
 - ◆ **Test Model**
 - ◆ **Change Requests**
- ◆ **For each iteration:**
 - Test Plan
 - Iteration Plan
 - End-User Support Material (preliminary)
 - Installation Artifacts
 - Release Notes
- © 2000 École Polytechnique de Montréal & Rational Software Learning software process with UPEDU Slide 6-16

Objectives of this Iteration

- ◆ Requirements are stable.
- ◆ The architecture is fully implemented.
- ◆ Many functionalities have been implemented and integrated
- ◆ Most of the effort will be spent in the implementation and test disciplines
- ◆ The project is approaching its first "beta" release

**UPEDU Concepts: Development and
Integration Workspaces**