

INF 111 / CSE 121:  
Software Tools and Methods

Lecture Notes for Fall Quarter, 2007  
Michele Rousseau  
Set 4

(Some slides adapted from Susan E. Sim)

---

---

---

---

---

---

---

---

Previous Lecture

- What are Software Tools & Methods?

Topic 4

2

---

---

---

---

---

---

---

---

Today's Lecture

- Finish up with Methods & Notations
- Process Modeling
  - Agile Process
  - Extreme Programming

Topic 4

3

---

---

---

---

---

---

---

---

## Notations, Tools & Methods

- **Tools:**
  - Machines, Executable Programs
- **Methods:**
  - Processes, Procedures
- **Notations:**
  - Languages Used by Tools and Methods

Remember the Guitar Example

Tool: Guitar

Method: How I play (strum/pick/style)

Notation: Music

Topic 4 4

---

---

---

---

---

---

---

---

---

---

## To Answer A Question From Wed

- **Process**
  - The sequence in which something is done
- **Procedure**
  - A *specified* sequence in which you do something
- **Thus... A procedure is a specification of a process**

Topic 4 5

---

---

---

---

---

---

---

---

---

---

## Integrated Project Support Environments (IPSE)

- **Supports the Entire Project**
  - Analyst Workbench
  - Programmer Workbench
  - Management Workbench
- **Tight Integration vs. Loose Integration**

Topic 4 6

---

---

---

---

---

---

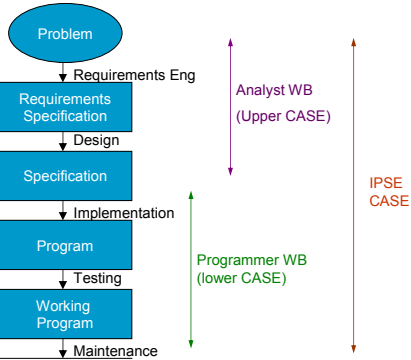
---

---

---

---

## Integrated Environments / Workbenches



Topic 4  
Adapted from Van Vliet

7

---

---

---

---

---

---

---

---

---

---

## Process-Centered S/E Environment (PSEE)

- Supports the entire Development Process
- Closely Tied to Process Modeling
  - Petri-Nets
  - State Transition Diagrams
  - Etc...
- Tends to support Back-End (Imp. & Testing)
  - Easier to Formalize

Topic 4

8

---

---

---

---

---

---

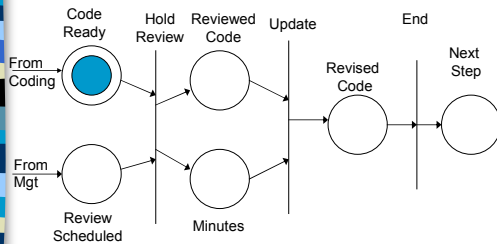
---

---

---

---

## Petri-Net View of PSEE



Topic 4

9

---

---

---

---

---

---

---

---

---

---

### Some of the Tools/Environments We Will Use

- Eclipse JDT
- JUnit
- Eclipse Plugins
- Argo UML (Or Rational Rose)
- Etc...

Topic 4 10

---

---

---

---

---

---

---

---

### Remember -- Selecting a Tool?

IDEAL TYPICAL

Topic 4 11

---

---

---

---

---

---

---

---

### Methods

- A **Method** is a technical prescription for how to perform a collection of activities, focusing on integration of techniques and guidance on their use.
- Prescribe → to lay down a rule
- A **Technique** is a prescription of how to perform a particular activity
  - May include rules on how to describe a product of that activity in a particular notation
  - Smaller than a Method
  - Example: Unit Testing

Topic 4 12

---

---

---

---

---

---

---

---

## Graphically

**Technique – How to perform as specific Activity**

- Activity 1
- Activity 2
- Activity 3

**Method – How to perform Many Activities**

Topic 4 13

---

---

---

---

---


---

---

---

## Tools vs. Methods

- **Construction**
  - Tools
    - ▣ Hammer
    - ▣ Saw
    - ▣ Measuring Tape
  - Methods
    - ▣ Rules for Construction



Topic 4 14

---

---

---

---

---


---

---

---

## Tools vs. Methods – Take 2

- **I give you a camera**
- **I teach you how to take a picture:**
  - Auto-focus
  - Push the Button
- **I teach you how to shoot a very nice picture**
  - Lighting
  - Aperture
  - Shutter Speed
  - Composition



Topic 4 15

---

---

---


---

---

---

---

---



## Method vs. Methodology

- A **method** is a description of how we do something
- A **methodology** is the study of methods

Topic 4 16

---

---

---


---

---

---

---

---



## Modeling

Topic 4 17

---

---

---


---

---

---

---

---



## Modeling

- A **model** is an abstract representation of a specification
- Defined by a consistent set of rules
  - Dictate the meaning of the components ... and interactions
- Some Basic Principles
  - Models are used for breaking down concepts
    - Requirements or Design (Unified Modeling Language)
  - Used for *communicating*
  - Choice of the Model influences the Product
    - Object Models
    - Data Repository Models
    - Pipe and Filter
    - Etc..

Topic 4 18

---

---

---

---

---

---

---

---

## For Example

- o **Different Modeling Styles Can Influence the Product**

- Styles restrict the way in which components can be connected
- Prescribe patterns of interaction
- Promote fundamental principles
  - Rigor, separation of concerns, anticipation of change, generality, incrementality
  - Low coupling
  - High cohesion

- o **Architectural styles are based on success stories**

- Almost all compilers are build as "pipe-and-filter"
- Almost all network protocols are build as "layers"

Topic 4

19

---

---

---

---

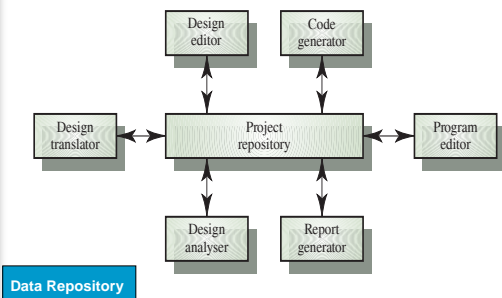
---

---

---

---

## Model Case Toolset



Topic 4

20

---

---

---

---

---

---

---

---

## Modeling (2)

- o **One Model → One Viewpoint**

- Specification View vs. Design View
- Runtime View vs. Compile-Time View
- Static View vs. Dynamic View

- o **Model should be realistic**

- o **Model is usually incomplete**

- Abstraction – doesn't include details

- o **Other Disciplines use Models too...**

- Architects – Buildings
- Circuit Board Designers

Topic 4

21

---

---

---

---

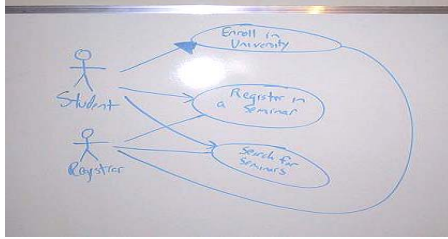
---

---

---

---

## Another Example



Topic 4

Models Can Be Informal or Formal

22

---

---

---

---

---

---

---

---

## Process Modeling

- A **process model** is an abstract description of how to conduct a collection of activities, focusing on resource usage and **dependencies** between activities
  - Often expressed using a notation

Remember: a **notation** is a representation scheme (or language)

Topic 4

23

---

---

---

---

---

---

---

---

## Software Process (revisited)

### General Software Process Activities

Phase	Purpose	Deliverables
User Requirements	Problem Def	User Req. Spec. Acceptance Test Plan
S/W Requirements	Problem Analysis	S/w Req. Spec. Support Service Brief System Test Plan
Architectural Design	High Level Solution	Architectural Design Support Serv. Design Integration Test Plan
Production	Implementation & Testing	Detailed Design Tested Software Est. Sup. Serv.
Transfer	Installation	Installed Software
Maint. & Support	S/w Operations & Support	Maintained & Supported S/W

24

---

---

---

---

---

---

---

---





## We Discussed Traditional S/W Process Models

- **Waterfall**
- **Spiral**
- **Incremental**
- **...etc**

Topic 4 25

---

---

---


---

---

---

---

---



## Criticisms with Traditional Process Models

- **Generally don't handle change well**
- **Implementation is delayed until uncertainties are completely resolved**
- **Too mechanistic to be used in detail**

Topic 4 26

---

---

---


---

---

---

---

---



## The Agile Method

- **Agile** – “having a quick resourceful and adaptable character” – Merriam-Webster
- **For smaller teams and businesses**
- **Quick Product Releases**

Topic 4 27

---

---

---


---

---

---

---

---



## Four Central Values of Agile Methods

- 1. Focus on the human role of s/w dev**
  - Interactions Between Developers “Communality”
  - Close Team Relationships
  - Close Working Arrangements
  - Team Spirit
- 2. Continuously turn out tested working software**
  - Small releases
  - Frequent Intervals (Hourly → Monthly)
  - Keep Code Simple & Technically Advanced → Reduces Documentation

Topic 4 28

---

---

---

---

---


---

---

---

---

---



## Four Central Values of Agile Methods

- 3. Foster the relationship with the client (over nitpicking the contract)**
  - Short releases allow clients to see progress
- 4. The Development Group**
  - Developers and Customer Reps
  - Should be:
    - ▣ Informed
    - ▣ Competent
    - ▣ Authorized to make changes
  - Contracts need to be formed with tools that support these changes

Topic 4 29

---

---

---

---

---


---

---

---

---

---



## Examples of Agile Methods

- **XP → Extreme Programming**
- **Scrum →**
  - “Getting out-of play ball back into the game”
- **FDD → Feature Driven Development**
- **RUP → Rational Unified Process**

Topic 4 30

---

---

---

---

---

---

---

---

---

---

## Extreme Programming (XP)

- o **Invented by Kent Beck in 1996**
  - "Seat of the pants" fix to Chrysler project
- o **Beck Published in 1999**
  - "Extreme Programming Explained: Embrace Change"
  - Current hot topic in S/W Process
  - Loved and Hated
  - Tries to associate s/w process with eXtreme sports
- o **Idea: Take a good programming practice and push it to the extreme**
  - Eg. Testing
  - Testing is good so... do it all the time

Topic 4

31

---

---

---

---

---

---

---

---

## Premise of XP

### o The Four Values



Hmmm... But aren't these standard "Best Practices"?  
What's new here?

Topic 4

32

---

---

---

---

---

---

---

---