Lecture 12: Integrating RE

Last Week:
- Evolving Requirements
- Change management
- Inconsistency management
- Product Families

This Week:
- Looking for patterns
- Method engineering
- Problem frames
- Analysis patterns

Next Week:
- Summary current RE practice
- Course Evaluation

Method Engineering

- We have looked at a number of RE methods
  - Methods for Elicitation: Interviews, Ethnography, Scenarios, task analysis, etc...
  - Methods for Modeling Enterprises, Goals & NFRs: KAOS, IT, Softgoal, etc...
  - Methods for Modeling System Functions: SSADM, SADT, OMT, UML, etc...
  - Methods for Writing Formal Specifications: SCR, RSML, etc...
  - Methods for Validating Reqs: Inspections, Prototyping, etc...
  - Methods for Negotiating Reqs: WinWin, Synoptic, Oz, etc...
  - Methods for Managing Evolving Reqs: ViewPoints, Default Logic, etc...
  - ...and some of these methods cover several different aspects of RE

- How do we choose which method(s) to adopt?
  - Method Engineering:
    - Development and customization of methods for specific purposes
    - Includes process guidance for when and how to use the methods
  - Method Integration:
    - Create normative RE process models that combine multiple methods
    - But you first need to know what type of RE problem you are tackling...

- Are methods the only way to capture good practice?
  - Some people argue that the focus on methods is wrong...
  - If we want to learn how good RE is done, look for patterns in the outputs...

The “Patterns” Movement

- Background
  - Engineers/Architects do not solve every problem from first principles
    - When they find a good solution, they use it repeatedly
    - C. F. Christopher Alexander “Notes on the Synthesis of Form”
    - Identified the need for a pattern language in architectural design

- Design Patterns
  - e.g. Book by Gamma, Helm, Johnson, Vlissides (aka “the gang of four”)
  - Presents a catalogue of patterns for object-oriented design
    - Really these are program-level (execution) patterns
    - Examples: factory; singleton; decorator; façade; visitor;...

- Analysis Patterns
  - e.g. Book by Martin Fowler
  - Presents a catalogue of patterns for conceptual modeling
    - Examples: Organizational structure; measurement; accounting; planning;...

- Problem Frames
  - e.g. Book by Michael Jackson
  - Presents a catalogue of patterns for figuring out what the problem is
    - Examples: workpieces; information display; commanded behaviour; connection;...

What is a pattern?

“an idea that has been useful in one practical context, and will probably be useful in others” - Fowler

- Elements:
  - Name - immensely useful for communicating your solution to others
  - Context - where the pattern is useful
  - Problem - that the pattern addresses
  - Forces - that play a part in forming a solution
  - Solution - that resolves those forces

- Example: (from Fowler)
  - Name: Contract
  - Context: any kind of financial deal
  - Problem: how to represent the transaction of buying and selling
  - Forces: distinguish two parties; buyer’s and seller’s views look different; a deal really involves 2 instruments, but one is usually money;...
  - Solution:
    - Party
    - Contract
    - Amount
    - Names
    - Price
    - Money
    - Instrument
Problem Frames

- Software is used to address an incredible variety of problems
  - Often there is little similarity between problem types
  - other than that the solution involves software
  - E.g. ticket machine vs. payroll system vs. signal processor vs. website vs. ...
- Need identify and classify problem types
  - Problem frames are an abstraction from classes of problems
    - A problem frame has principal parts and a solution task
    - Problem frames are ridiculously simplistic (but still helpful)
    - Some problems require multiple problem frames
  - Select a problem frame that achieves:
    - Separability: Must be able to separate the principal parts of the problem
    - Completeness: Every part of the problem must be accommodated
    - Part Characteristics: The parts of the problem must have the right characteristics in the model
    - Proportionality: The parts of the model should be filled roughly equally

Jackson's Frame Diagrams

Example:
- inputs → machine → outputs
- Machine domain
- Relationship between application domains
- source program → executable program
- Language and compiler semantics
- Input/Output relationship

Workpieces Frame

- Workpieces: An inert dynamic domain
  - Workpieces can change, but only in response to external stimuli
  - contained entirely in the machine domain
- Operation Requests: One dimensional active dynamic domain
  - time-ordered, no external stimulus
- Operation Properties: Define the effects of and constraints on operations
- Example ignores:
  - multiple users
  - operations no longer time ordered
  - Interaction between text files

Simple Information Display Frame

- Real World: An autonomous active dynamic domain
  - may be static for some problems
- Information Requests: Active dynamic domain
  - No assumed structure to the requests
- Information function:
  - This is the requirement
  - i.e. the system must preserve this function
- Information outputs must be accurate reflection of the state of the real world and must respond to information requests
- Frame ignores:
  - How outputs from the system might affect the real world

Example:
- Bank accounts
- Account Statements
- Account Requests
- Banking system
### Simple Control Frame

- **Controlled domain**
  - Dynamic
  - Both active and re-active
  - i.e., spontaneous changes, and externally influenced changes
  - May be several domains composed
  - Must be described indicatively
- **Controller**
  - Machine to be built
  - Directly connected to the controlled domain
- **Desired behaviour**
  - The Requirement
    - Described optatively
  - Example ignores:
    - Interaction of the user
    - Could be a non-reactive part of the controlled domain

**Example:**

- Controller
- Desired Behavior
- Washing Rules
- Washing machine
- Program sequencer

---

### Connection Frames

- **Use when...**
  - The machine and some part of the application domain have no shared phenomena
  - There is an unreliable connection between them
- **Two versions:**
  - The connection domain is the machine to be developed
  - The connection domain is given, and the machine is one end of the connection (not shown here)

**Example:**

- System
- MC
- Connection
- CR
- Real world

- Achievable correspondence
- Data modeling rules
- Data entry system
- Data collection
- Real world

---

### Multi-Frame Problems

- **Example: a CASE tool**
  - Editing diagrams
    - Workpiece Frame
  - Restricting Access
    - Simple Control Frame
    - Managing the process
      - Simple IS Frame
  - Managing Information
  - Definition
  - Users
  - CASE objects
  - Management Information
  - Information function
  - Information outputs
  - Access Restriction
  - Controlled domain
  - Desired behavior

---

*Source: Adapted from Jackson, 1995, p181-183*