

University of Toronto Department of Computer Science

Lecture 6: Requirements Modeling I

Last Week:
Elicitation (II)
Cognitive approaches
Contextual approaches
Ethnography

This Week:
Modeling and Analysis (I)
General Modeling Issues
Modeling Goals, Organizations,
and Non-Functional Requirements

Next Week:
Modeling and Analysis (II)
Modeling Functionality
Structured and OO methods
Formal Methods

© 2000-2003, Steve Easterbrook 1

University of Toronto Department of Computer Science

Motivation for modeling...

Imagine we have interviewed some stakeholders...

<p>Chief Executive</p> <ul style="list-style-type: none"> When flight is full VIPs are first to be upgraded. Discounted tickets should be offered to politicians, since they make important decisions affecting the airline. Info about frequent fliers should not be made available to outside contractors. <p>Chief Security Officer</p> <ul style="list-style-type: none"> The number of bags in the aircraft's hold should tally against the list of passengers on board. Passenger lists should not be made available to the public. Passengers should check-in only once. <p>Travel Agent</p> <ul style="list-style-type: none"> An agent is responsible for holding and canceling reservations. Tickets offered by an agency have different fares, negotiated with the airline sales department. 	<p>Catering Manager</p> <ul style="list-style-type: none"> The food loaded is dictated by the number of passengers travelling in a particular class. A predicted number of passengers on a flight must be available 24 hours prior to departure. Passengers requiring special meals must indicate their request 24 hours prior to departure. <p>Airline Sales manager</p> <ul style="list-style-type: none"> A ticket may only be issued when a fare is paid For some fares, a reservation can be held and not confirmed. When a discounted ticket is booked, the normal book-ahead requirements do not apply. All tickets must carry appropriate endorsements relating to the terms and conditions of issue of tickets.
--	---

How do we get from here to an agreed specification?

© 2000-2003, Steve Easterbrook 2

University of Toronto Department of Computer Science

RE involves a lot of modelling

→ A model is more than just a description

- ↳ it has its own phenomena, and its own relationships among those phenomena.
 - > The model is only useful if the model's phenomena correspond in a systematic way to the phenomena of the domain being modelled.
- ↳ Example:

The application domain

Designations for the application domain

B = Book
P = Person
R = Wrote

The modeling domain

Designations for the model's domain

B: entity
P: entity
R: relation(P,B)

Common Properties

For every B, at least one P exists such that R(P, B)

© 2000-2003, Steve Easterbrook Source: Adapted from Jackson, 1995, p120-122 3

University of Toronto Department of Computer Science

Remember: "It's only a model"

→ There will always be:

- ↳ phenomena in the model that are not present in the application domain
- ↳ phenomena in the application domain that are not in the model

Phenomena not captured in the model

...ghost writers...
...pseudonyms...
...anonymity...

Common Phenomena

...every book has at least one author...
...every book has a unique ISBN...

Phenomena not true in the world

...no two people born on same date with same name...

→ A model is never perfect

- ↳ "If the map and the terrain disagree, believe the terrain"
- ↳ Perfecting the model is not always a good use of your time...

© 2000-2003, Steve Easterbrook Source: Adapted from Jackson, 1995, p124-5 4

University of Toronto Department of Computer Science

Modeling...

- **Modeling can guide elicitation:**
 - Does the modeling process help you figure out what questions to ask?
 - Does the modeling process help to surface hidden requirements?
 - > i.e. does it help you ask the right questions?
- **Modeling can provide a measure of progress:**
 - Does completeness of the model imply completeness of the elicitation?
 - > i.e. if we've filled in all the pieces of the model, are we done?
- **Modeling can help to uncover problems**
 - Does inconsistency in the model reveal interesting things...?
 - > e.g. inconsistency could correspond to conflicting or infeasible requirements
 - > e.g. inconsistency could mean confusion over terminology, scope, etc
 - > e.g. inconsistency could reveal disagreements between stakeholders
- **Modeling can help us check our understanding**
 - Can we test that the model has the properties we expect?
 - Can we reason over the model to understand its consequences?
 - Can we animate the model to help us visualize/validate the requirements?

© 2000-2003, Steve Easterbrook 5

University of Toronto Department of Computer Science

Type of Model

Can choose a variety of conceptual schema:

- **natural language**
 - extremely expressive and flexible
 - very poor at capturing the semantics of the model
 - better used for elicitation, and to annotate models for communication
- **semi-formal notation**
 - captures structure and some semantics
 - can perform (some) reasoning, consistency checking, animation, etc.
 - > E.g.s: diagrams, tables, structured English, etc.
- **formal notation**
 - very precise semantics, extensive reasoning possible
 - long way removed from the application domain
 - > note: requirements formalisms are geared towards cognitive considerations, hence differ from most computer science formalisms

© 2000-2003, Steve Easterbrook Source: Adapted from Loucopoulos & Karakostas, 1995, p72-73 6

University of Toronto Department of Computer Science

Desiderata for Conceptual Schema

- **Implementation Independence**
 - does not model data representation, internal organization, etc.
- **Ease of analysis**
 - ability to analyze for ambiguity, incompleteness, inconsistency
- **Abstraction**
 - extracts essential aspects
 - > e.g. things not subject to frequent change
- **Traceability**
 - ability to cross-reference elements
 - ability to link to design, implementation, etc.
- **Formality**
 - unambiguous syntax
 - rich semantic theory
- **Executability**
 - can animate the model, to compare it to reality
- **Constructability**
 - can construct pieces of the model to handle complexity and size
 - construction should facilitate communication
- **Minimality**
 - No redundancy of concepts in the modeling scheme
 - > i.e. no extraneous choices of how to represent something

© 2000-2003, Steve Easterbrook Source: Adapted from Loucopoulos & Karakostas, 1995, p77 7

University of Toronto Department of Computer Science

Meta-Modeling

- **Can compare modeling schema using meta-models:**
 - What phenomena does each scheme capture?
 - What guidance is there for how to elaborate the models?
 - What analysis can be performed on the models?
- **Example meta-model:**

```

graph TD
    Facts((Facts))
    Events((Events))
    Activities((Activities))
    Facts -- modify --> Activities
    Facts -- acknowledge --> Events
    Events -- trigger --> Activities
    
```

© 2000-2003, Steve Easterbrook 8

University of Toronto Department of Computer Science

Modeling Principle 1: Partitioning

→ **Partitioning**

- ↳ captures aggregation/part-of relationship

→ **Example:**

- ↳ goal is to develop a spacecraft
- ↳ partition the problem into parts:
 - > guidance and navigation;
 - > data handling;
 - > command and control;
 - > environmental control;
 - > instrumentation;
 - > etc
- ↳ **Note:** this is not a design, it is a problem decomposition
 - > actual design might have any number of components, with no relation to these sub-problems
- ↳ However, the choice of problem decomposition will probably be reflected in the design

© 2000-2003, Steve Easterbrook 9

University of Toronto Department of Computer Science

Modeling Principle 2: Abstraction

→ **Abstraction**

- ↳ A way of finding similarities between concepts by ignoring some details
- ↳ Focuses on the general/specific relationship between phenomena
 - > Classification groups entities with a similar role as members of a single class
 - > Generalization expresses similarities between different classes in an 'is_a' association

→ **Example:**

- ↳ requirement is to handle faults on the spacecraft
- ↳ might group different faults into fault classes

based on location of fault: **OR** based on symptoms of fault:

- ↳ instrumentation fault,
- ↳ communication fault,
- ↳ processor fault,
- ↳ etc
- ↳ no response from device;
- ↳ incorrect response;
- ↳ self-test failure;
- ↳ etc...

© 2000-2003, Steve Easterbrook Source: Adapted from Davis, 1990, p48 and Loucopoulos & Karakostas, 1995, p78 10

University of Toronto Department of Computer Science

Modeling Principle 3: Projection

→ **Projection:**

- ↳ separates aspects of the model into multiple viewpoints
 - > similar to projections used by architects for buildings

→ **Example:**

- ↳ Need to model the communication between spacecraft and ground system
- ↳ Model separately:
 - > sequencing of messages;
 - > format of data packets;
 - > error correction behavior;
 - > etc.

→ **Note:**

- ↳ Projection and Partitioning are similar:
 - > Partitioning defines a 'part of' relationship
 - > Projection defines a 'view of' relationship
- ↳ Partitioning assumes a the parts are relatively independent

© 2000-2003, Steve Easterbrook Source: Adapted from Davis, 1990, p48-51 11

University of Toronto Department of Computer Science

Survey of Modeling Techniques

→ **Modeling Enterprises**

- ↳ Goals & objectives
- ↳ Organizational structure
- ↳ Activities, processes, and products
- ↳ Agents and work roles

→ **Modeling Functional Requirements**

- ↳ Information Structure
- ↳ Behavioral views
- ↳ Timing/Sequencing requirements

→ **Modeling Non-functional Requirements**

- ↳ Product requirements
- ↳ Process requirements
- ↳ External requirements

Information modeling:
ERD

Organization modeling:
i*, SSM, ISAC

Goal modeling:
KAOS, CREWS

Structured Analysis:
SADT, SSADM, JSD

Object Oriented Analysis:
OOA, OOSE, OMT, UML

Formal Methods:
SCR, RSML, Z, Larch, VDM

Quality tradeoffs:
QFD, win-win

Specific NFRs:
Timed Petri nets (performance)
Task models (usability)
Probabilistic MTTF (reliability)

© 2000-2003, Steve Easterbrook 12

University of Toronto Department of Computer Science

Approaches to Enterprise Modeling

→ 1970's

- Soft Systems Approaches:
 - involve the entire organisation
 - Be sensitive to political and social context for organisational change
- Examples: SSM, ISAC

→ 1980's

- Knowledge-based Approaches:
 - Use knowledge representation schemes to build executable domain models
 - capture static and dynamic aspects of the domain
- Examples: RML, Requirements Apprentice, Nature

→ 1990's

- Teleological Approaches:
 - Requirements are really just goals, so model goal hierarchies
 - Focus on the 'why' question, rather than 'what'/'how'
 - ...and use scenarios as concrete examples of how goals are (can be) satisfied
- Examples: KAOS, i*, CREWS,...

→ 2000's ...?

© 2000-2003, Steve Easterbrook 13

University of Toronto Department of Computer Science

Entity Relationship Diagrams

→ ER diagrams

- widely used for information modeling
- simple, easy to use
 - Note: this is a notation, not a method!

→ Used in many contexts:

- domain concepts
 - objects referred to in goal models, scenarios, etc.
- Data to be represented in the system
 - for information systems
- Relational Database design
- Meta-modeling

Key

- Entity
- Attribute
- Relationship
- Cardinality of relationship (a,b) (c,d)
- Identifier
- Composite Identifier

© 2000-2003, Steve Easterbrook 14

University of Toronto Department of Computer Science

ISAC

→ Information Systems Work & Analysis of Changes (ISAC)

- Developed in the 1970's in Sweden
- Emphasizes cooperation between users, developers and sponsors
 - Developers' role is to facilitate the process
- Good for information systems; not applicable to control systems.

→ ISAC Process

- Change Analysis
 - What does the organization want?
 - How flexible is the organization with respect to changes?
- Activity Study
 - Which activities should we regroup into information systems?
 - Which priorities do the information systems have?
- Information Analysis
 - Which inputs and outputs do each information system have?
 - What are the quantitative requirements on each information system?
- Implementation
 - Which technology (info carriers; h/w; s/w) do we use for the information systems?
 - Which activities of each information system are manual, which automatic?

© 2000-2003, Steve Easterbrook 15

University of Toronto Department of Computer Science

ISAC Change Analysis

- List problems
 - dissatisfactions with current system
 - list all problems...
 - ...then remove any that are trivial or intractable
- List interest groups
 - these are "problem owners"
 - draw matrix of problems against owners
 - This exercise is done with the problem owner's involvement
- Analyze problems
 - Use cause-effect analysis
 - Eliminate solution-oriented problems, to get to underlying causes
 - performed by domain specialists
 - quantify the problems
- Make Current Activity Model
 - Notation: A-schemas (similar to dataflow diagrams)
- Analyze Goals
 - Declarative statement of goals
 - i.e. desired result, not how to get there
 - Result should be a tree of goals
- Define Change Needs
 - Goals should explain why the problems exist; problems frustrate goals
 - Cluster problems into related groups
 - Each group is a change need
- Generate Change Alternatives
- Model desired situations
 - make packages of change alternatives
- Evaluate Alternatives
- Choose an alternative

© 2000-2003, Steve Easterbrook 16

University of Toronto Department of Computer Science

Soft System Methodology (SSM)

→ **Background**

- Developed by Checkland in late 1970's
- Reality is socially constructed, and therefore requirements are not objective
- Rationale:
 - Problem situations are fuzzy (not structured) and solutions not readily apparent.
 - Impact of a computerization may be negative (e.g. intro of new system reduced productivity as it removed employee motivation)
 - Full exploitation of computerization may need radical restructuring of work processes.

→ **Approach**

- Analyze problem situation using different viewpoints
 - Determining the requirements is a discussion, bargaining and construction process.
- Out of this process emerges not just a specification, but also:
 - plans for a modified organization structure
 - task structures
 - objectives
 - understanding of the environment

© 2000-2003, Steve Easterbrook 17

University of Toronto Department of Computer Science

SSM Approach

- Existing situation (unstructured problem)**
- Analyze the problem situation**
 - Draw a rich picture
 - look for problem themes (describe them in natural language)
- Define relevant systems and root definitions (CATWOE)**
 - a root definition is a concise description of a human activity system
- Build a conceptual model**
 - of the activity system needed to achieve the transformation
 - process oriented model, with activities & flow of resources
- Compare conceptual model with step (2)**
 - Ordered questioning - questions based on the model
 - Event reconstruction - take past events and compare them to the model
 - General comparison - look for features of the model that are different from current situation
 - Model overlay - point by point comparison of the two models
- Debate feasible and desirable changes**
 - Three types of change: structural, procedural, attitudinal
- Implement changes**

© 2000-2003, Steve Easterbrook 18

University of Toronto Department of Computer Science

SSM modeling

Root definition:

"A hospital-owned system, which provides records of spending on drugs so that control action by administrators and doctors to meet defined budgets can be taken jointly"

Customers: Administrators, Doctors
Actors: not stated
Transformation: Need to know spending on drugs → Need met by recording info.
Weltanschauung: Monitoring spending on drugs is possible and is an adequate basis for joint control action
Owner: Hospital
Environment: Hospital mechanisms, roles of administrators and doctors, defined budgets

© 2000-2003, Steve Easterbrook 19

University of Toronto Department of Computer Science

i*

→ **Background**

- Developed in the early 90's
 - provides a structure for asking 'why' questions in RE
 - models the organisational context for information systems
 - based on the notion of an "intentional actor"
- Two parts to the model
 - Strategic dependency model - models relationships between the actors
 - Strategic rationale model - models concerns and interests of the actors

→ **Approach**

- SD model shows dependencies between actors:
 - goal/softgoal dependency - an actor depends on another actor to attain a goal
 - resource dependency - an actor needs a resource from another actor
 - task dependency - an actor needs another actor to carry out a task
- SR model shows interactions between goals within each actor
 - Shows task decompositions
 - Shows means-ends links between tasks and goals

© 2000-2003, Steve Easterbrook 20

