MTAT.03.306

Lecture 2: RE framework
Reminder

• Software intensive systems
• Project management
• Requirements engineering
• What is “Requirement”? 
• What is “Engineering”? 
• Lifecycles of engineering projects
Table of Contents

• What is system?
• Continuous RE
• RE framework

• Prof. Steve Easterbrook, Requirements engineering course, University of Toronto
Table of Contents

• What is system?
• Continuous RE
• RE framework

• Prof. Steve Easterbrook, Requirements engineering course, University of Toronto
Elements of a system

• **Boundary**
  – Separates a system from its environment
  – Often not sharply defined
  – Also known as an “interface”

• **Environment**
  – Part of the world with which the system can interact
  – System and environment are inter-related

• **Observable Interactions**
  – How the system interacts with its environment
  – E.g. inputs and outputs

• **Subsystems**
  – Can decompose a system into parts
  – Each part is also a system
  – For each subsystem, the remainder of the system is its environment
  – Subsystems are inter-dependent

• **Control Mechanism**
  – How the behaviour of the system is regulated to allow it to endure
  – Often a natural mechanism

• **Emergent Properties**
  – Properties that hold of a system, but not of any of the parts
  – Properties that cannot be predicted from studying the parts
Conceptual Picture of a System

System Boundary

How the system’s properties evolve over time

How the system is controlled

Inputs

Feedforward

Control

Feedback

Outputs
Hard vs. Soft Systems

Hard Systems:

- The system is...
  - ...precise,
  - ...well-defined
  - ...quantifiable
- No disagreement about:
  - Where the boundary is
  - What the interfaces are
  - The internal structure
  - Control mechanisms
  - The purpose ??
- Examples
  - A car (?)

Soft Systems:

- The system...
  - ...is hard to define precisely
  - ...is an abstract idea
  - ...depends on your perspective
- Not easy to get agreement
  - The system does not “really” exist
  - Calling something a system helps us to understand it
  - Identifying the boundaries, interfaces, controls, helps us to predict behaviour
  - The “system” is a theory of how some part of the world operates
- Examples:
  - All human activity systems
Types of System

• **Natural Systems**
  – E.g. ecosystems, weather, water cycle, the human body, bee colony,…
  – Usually perceived as hard systems

• **Abstract Systems**
  – E.g. set of mathematical equations, computer programs,…
  – Interesting property: system and description are the same thing

• **Symbol Systems**
  – E.g. languages, sets of icons, streetsigns,…
  – Soft because meanings change

• **Designed Systems**
  – E.g. cars, planes, buildings, freeways, telephones, the internet,…

• **Human Activity Systems**
  – E.g. businesses, organizations, markets, clubs,…
  – E.g. any designed system when we also include its context of use
    • Similarly for abstract and symbol systems!

• **Information Systems**
  – Special case of designed systems
    • Part of the design includes the representation of the current state of some human activity system
  – E.g. MIS, banking systems, databases,…

• **Control systems**
  – Special case of designed systems
    • Designed to control some other system (usually another designed system)
  – E.g. thermostats, autopilots,…
Information Systems

Source: Adapted from Loucopoulos & Karakostas, 1995, p73

Subject System

Needs information about

Usage System

Maintains information about

Information system

Usage System

Uses

Development System

contracts

builds
Control Systems

Subject system

Needs to ensure safe control of

Tracks and controls the state of

Usage System

Uses

Control system

contracts

Development System

builds

Development System
Purposefulness

• *Types of behaviours:*
  
  – **Reaction** to a stimulus in the environment
    • The stimulus is necessary and sufficient to cause the reaction
  
  – **Response** to a stimulus in the environment
    • The stimulus is necessary but not sufficient to cause the response
  
  – **Autonomous act:**
    • A system event for which a stimulus is not necessary
Purposefulness

• Systems can be:
  – **State-maintaining**
    • System *reacts* to changes in its environment to maintain a pre-determined state
    • E.g. thermostat, some ecosystems
  – **Goal-directed**
    • System can *respond* differently to similar events in its environment and can *act autonomously* in an unchanging environment to achieve some pre-determined goal state
    • E.g. an autopilot, simple organisms
  – **Purposive**
    • System has *multiple goals*, can choose how to pursue them, but no choice over the goals themselves
    • E.g. computers, animals (?)
  – **Purposeful**
    • System has multiple goals, and can choose to *change its goals*
    • E.g. people, governments, businesses, animals
Describing System Behaviour

Source: Adapted from Wieringa, 1996, p16-17

- **State**
  - a system will have memory of its past interactions, i.e. ‘state’
  - the state space is the collection of all possible states

- **Discrete vs continuous**
  - a discrete system:
    - the states can be represented using natural numbers
  - a continuous system:
    - state can only be represented using real numbers
  - a hybrid system:
    - some aspects of state can be represented using natural numbers

- **Observability**
  - the state space is defined in terms of the observable behavior
  - the perspective of the observer determines which states are observable
Table of Contents

• What is system?
• **Continuous RE**
• RE framework

• Prof. Steve Easterbrook, Requirements engineering course, University of Toronto
Lifecycle of Engineering Project

• **Lifecycle models**
  – Useful for comparing projects in general terms
  – Not enough detail for project planning

• **Examples:**
  – Sequential models: Waterfall, V model
  – Rapid Prototyping
  – Phased Models: Incremental, Evolutionary
  – Iterative Models: Spiral
  – Agile Models: eXtreme Programming
Waterfall Model

- **View of development:**
  - a process of stepwise refinement
  - largely a high level management view

- **Problems:**
  - Static view of requirements - ignores volatility
  - Lack of user involvement once specification is written
  - Unrealistic separation of specification from design
  - Doesn’t accommodate prototyping, reuse, etc.
Phased Lifecycle Models

**Incremental development** (each release adds more functionality)

- **Release 1**
  - reqts
  - design
  - code
  - test
  - integrate
  - O&M

- **Release 2**
  - reqts
  - design
  - code
  - test
  - integrate
  - O&M

- **Release 3**
  - reqts
  - design
  - code
  - test
  - integrate
  - O&M

- **Release 4**
  - reqts
  - design
  - code
  - test
  - integrate
  - O&M

**Evolutionary development** (each version incorporates new requirements)

- **Version 1**
  - reqts
  - design
  - code
  - test
  - integrate
  - O&M

- **Version 2**
  - reqts
  - design
  - code
  - test
  - integrate
  - O&M

- **Version 3**
  - reqts
  - design
  - code
  - test
  - integrate
  - O&M

**Source:** Adapted from Dorfman, 1997, p10
Table of Contents

• What is system?
• Continuous RE

• RE framework
  – System context
  – Core activities
  – Requirements artefacts
  – Validation
  – Management

• Prof. Steve Easterbrook, Requirements engineering course, University of Toronto
Table of Contents

• What is system?
• Continuous RE
• RE framework
  – System context
  – Core activities
  – Requirements artefacts
  – Validation
  – Management

• Prof. Steve Easterbrook, Requirements engineering course, University of Toronto
The part of the system environment relevant for defining, understanding, and interpreting the system requirements.
System context

Subject facet
Usage facet
IT system facet
Development facet

The part of the system environment relevant for defining, understanding, and interpreting the system requirements
System context

Subject facet | Usage facet | IT system facet | Development facet

The part of the system environment relevant for defining, understanding, and interpreting the system requirements
- **Subject facet**: objects and events that are relevant for the system,
  ✔ because the system must store or process information about these objects

- **Usage facet**: aspects concerning the usage of the system by people and other systems

- **IT system facet**: aspects concerning the operational or technical environment in which the system is deployed

- **Development facet**: aspects that influence the development of the system
  ✔ imposed by law, or by client and relate to the development process
Table of Contents

• What is system?
• Continuous RE
• RE framework
  – System context
  – Core activities
  – Requirements artefacts
  – Validation
  – Management

• Prof. Steve Easterbrook, Requirements engineering course, University of Toronto
Performing iteratively

Core activities

- Documentation
- Elicitation
- Negotiation

Requirements artefacts

- Goals
- Scenarios

Solution oriented requirements

Subject facet
Usage facet
IT system facet
Development facet

Validation Management

System context
Document important information elicited or developed when performing a core the RE activity

- i.e., documentation, elicitation, negotiation, validation and/or management
Achieve progress in the content dimension by eliciting new requirements as well as detailed information about existing requirements.

• Elicit all requirements at the level of detail for the system to be developed.
Achieve agreement among all stakeholders about the requirements
• has to deal with conflicts about requirements
Is there a “Requirements Lifecycle”

Source: Adapted from Pohl, CAISE 1993
Table of Contents

• What is system?
• Continuous RE

• RE framework
  – System context
  – Core activities
  – Requirements artefacts
  – Validation
  – Management

• Prof. Steve Easterbrook, Requirements engineering course, University of Toronto
What are requirements?

- **Domain Properties:**
  - things in the *application domain* that are true whether or not we ever build the proposed system

- **Requirements:**
  - things in the *application domain* that we wish to be made true by delivering the proposed system
    - Many of which will involve phenomena the machine has no access to

- **A Specification:**
  - is a description of the behaviours that *the program* must have in order to meet the *requirements*
    - Can only be written in terms of shared phenomena!
Intention with regard to objectives, properties, or use of the system
Document sequences of interactions in which the system either satisfies some goals or fails to satisfy them.
Specify at the required level of detail, the desired properties and features of the system to be developed.
Documenting requirements artefacts

- **Identifier** (1)
- **Description** (1)
- **Requirements artefact**
  - **Goal** (1..*), (1..*)
  - **Scenario** (1..*)
  - **Solution-oriented requirement** (1..*) (0..*)
- **Criticality** (1)
- **Priority** (1)
- **Risk** (0..*)

Relations:
- **Identifier** has **Requirements artefact** (has)
- **Description** has **Requirements artefact** (has)
- **Requirements artefact** has **Goal** (has)
- **Requirements artefact** has **Scenario** (has)
- **Requirements artefact** has **Solution-oriented requirement** (has)
- **Goal** has **Scenario** (example of satisfaction)
- **Scenario** has **Goal** (derived from)
- **Scenario** contributes to realisation of **Requirements artefact**
- **Requirements artefact** is realised by **Goal**
## Requirement Shell

**Volere template, 2010**

<table>
<thead>
<tr>
<th>Requirement #</th>
<th>Unique id</th>
<th>Requirement Type</th>
<th>Event/use case #</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>A one sentence statement of the intention of the requirement</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Rationale</strong></td>
<td>A justification of the requirement</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Originator</strong></td>
<td>Who raised this requirement?</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Fit Criterion</strong></td>
<td>A measurement of the requirement such that it is possible to test if the solution matches the original requirement</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Customer Satisfaction</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Priority</strong></td>
<td>The relative urgency of this requirement</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Supporting Materials</strong></td>
<td>Pointer to documents that illustrate and explain this requirement</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>History</strong></td>
<td>Creation, changes, deletions, etc.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Customer Disatisfaction</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Conflicts</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Other requirements</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Degree of stakeholder happiness** if this requirement is successfully implemented. Scale from 1 = uninterested to 5 = extremely pleased.
- **Measure of stakeholder unhappiness** if this requirement is not part of the final product. Scale from 1 = hardly matters to 5 = extremely displeased.
Table of Contents

• What is system?
• Continuous RE

• RE framework
  – System context
  – Core activities
  – Requirements artefacts
    – Validation
  – Management

• Prof. Steve Easterbrook, Requirements engineering course, University of Toronto
**Validating**
- Consideration of system context
- Execution of RE activities
- Created requirements artefacts

**Validation techniques:**
- Inspection
- Reviews
- Walkthroughs
- Perspective-based reading
- Prototyping
Table of Contents

• What is system?
• Continuous RE
• RE framework
  – System context
  – Core activities
  – Requirements artefacts
  – Validation
  – Management

• Prof. Steve Easterbrook, Requirements engineering course, University of Toronto
- Establishing requirements traceability
- Prioritising requirements
- Managing changes of requirements artefacts
Take Home!!!

• What is system?
• Continuous RE
• RE framework
  – System context
  – Core activities
  – Requirements artefacts
  – Validation
  – Management