Expand requirement engineering with SysML and MagicRQ

Robert Ong
No Magic Asia Ltd
What is a Requirement?

- A condition or capability needed by a user to solve a problem or achieve an objective.
- A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.

Why Do Requirements Matter?

• Requirements are the foundation of a good software design
• Requirements are a project team’s success criteria
• Requirements determine the project scope, size and estimation
• Requirements provide the basis for designing, coding and testing
• Requirements are what the customers need
The Source of Software Defects

- Requirement Errors: 56%
- Design Errors: 27%
- Coding Errors: 7%
- Other Errors: 10%

Source: James Martin, An Information Systems Manifesto
The Cost to Fix Software Defects

Source: James Martin, An Information Systems Manifesto
How Does UML Capture Requirements?

- **Use Case Diagram**
  - Defines the needs from the stakeholder's perspective to the software requirements
  - Scopes out the system boundaries
  - Captures and describes the major behaviors of the system
  - Identifies the main user
  - Validates requirements
  - The source of estimation

- **Activity Diagram**
  - Alternatives for describing Use Cases - Use Case Scenarios
  - The use case model is enhanced by activity diagrams, which can show both the business process in which the use case occurs and the process of the use case itself.
  - Use case diagram and activity diagram have a recursive relationship with each other.
  - Use Case covers a range of situations, while a scenario is just one
  - It is very useful for system and user-acceptance testing
Capture Requirements with Use Case Description

• **Pre-Condition**
  - Describe a state of the system that must be true before you can use the use case, these conditions must be tested by the use case before doing anything else

• **Post-Condition**
  - Describe what the system must do if the pre-conditions are satisfied and the state the system must be in when the use case ends

• **Basic flow**
  - Describe a step-by-step interaction between the user and the executing use case

• **Alternative flow**
  - Describe a step-by-step interaction if the normal flow fails to execute

• **Others**
  - Goal, implementation issue, non-functional requirements, notes, etc
What is missing?

- How we define the different kinds of requirements in UML?
- How can we define requirements hierarchy?
- How can we specify the relationship among different kinds of requirements?
- How can we specify the relationship between requirements and model elements?
- How can we reuse a requirement and define its master/slave relationship?
- How can we handle and represent all kinds of non-functional requirements?
How About Non-Functional Requirements?

Non-functional

Product Requirements

Organizational Requirements

External Requirements

Efficiency Requirements

Reliability Requirements

Portability Requirements

Interoperability Requirements

Ethical Requirements

Usability Requirements

Delivery Requirements

Implementation Requirements

Standards Requirements

Legislative Requirements

Performance Requirements

Space Requirements

Privacy Requirements

Safety Requirements
When Use Cases Aren’t Enough

- Use case is not enough to capture any system that does not involve user interaction or involves complex business rules and performs complex calculations. For example:
  - Real-time hardware system
  - Embedded control system
  - Complex business rules system
  - Heavy computation system

- Use Case does not represent every kind of functional requirements

- Use Case makes good sense to users but not developers
What is SysML?

OMG System Modeling Language (SysML)
Developed by OMG and INCOSE, and AP233. Adopted by OMG in May ‘06
Realized as UML subset

SysML is a common and more natural modeling language for system engineering with specific terminology and substructures.

→ http://www.omgsysml.org
• SysML provides 11 diagrams for System Engineers to model a system

• 2 new diagrams introduced in SysML - Requirement Diagram & Parametric Diagram
• Describes textual requirements contained in the specification

• Represents requirements hierarchies and relationships

• Refine Requirement
• Verify Requirement
• Satisfy Requirement
• Derive Requirement
• Copy Requirement
Structural Diagrams

- Represent structural elements for implementing solution for defined requirements

Define
- System containing blocks and their inner structure;
- Domain and solution vocabulary;
- System decomposition into layers and subsystems;
- Interfaces of the system and its components;
- System constraints between structural elements.

Are represented by:
- Package Diagram
- Block Definition Diagram
- Internal Block Diagram
- Parametric Diagram
• Used to organize models

• Provide views of a system from multiple levels of abstraction

• Describe dependencies between packages and their inner elements
View and Viewpoint

- Viewpoints are a way of structuring the requirements to represent the perspectives of different stakeholders. Stakeholders may be classified under different viewpoints.

- This multi-perspective analysis is important as there is no single correct way to analyze system requirements.

- Identify viewpoints using
  - Providers and receivers of system services;
  - Systems that interact directly with the system being specified;
  - Regulations and standards;
  - Sources of business and non-functional requirements.
  - Engineers who have to develop and maintain the system;
  - Marketing and other business viewpoints.
• Represents the structure of a system
• Contains blocks and relationships
• May represent domain vocabulary (data types)
Internal Block Diagram

- Describes internal structure of the blocks
- Contains properties and connectors
Parametric Diagram

- Models system constraints
- Block definition diagram contains constraint blocks for equations
- Parametric diagram represents a usage of the constraints in an analysis context
Behavioral Diagrams

- Represents dynamic interaction between system components for implementing requirements
- Shows distribution of responsibilities
- Identifies system functions
- Models system behavior and the way these behaviors are related
- Allows identification of interactions and coupling bottlenecks

- Are represented by:
  - Activity Diagram
  - Sequence Diagram
  - State Machine Diagram
  - Use Case Diagram

![Diagram showing Behavioral Diagrams and their subtypes](image)
Activity Diagram

- Shows a procedural flow for a system behavior
- Useful for workflow modeling
- Supports continuous flow modeling
• Describes how a process is performed by a group of objects in a sequential set of interactions

• Provides an object-oriented view of a procedural view

• Facilitates assignment of responsibilities to blocks

• Helps discover new methods and new blocks

• Shows timing very explicitly
State Machine Diagram

- Represents block life cycle.
- Defines system behavior as sequence of states that a component or interaction experiences in its response to events.
Crosscutting Constructs

- Allocations represent general relationships that map one model element to another

- May be displayed in diagram or in table
SysML in the System Development Process

- System Requirements & Business Analysis
- System Analysis & Design
  - Evaluation & Optimization (V&V)
  - Detailed Design: Software & Hardware
    - System Integration
      - System Implementation
SysML in the System Development Process

- Requirements diagram
- Use Case Diagram
- Activity diagram

System Requirements & Business Analysis

System Analysis & Design

Evaluation & Optimization (V&V)

Detailed Design: Software & Hardware

System Integration

System Implementation
SysML in the System Development Process

- Requirements diagram
- Use Case Diagram
- Activity diagram

- Package diagram
- Block Definition diagram
- Activity diagram
- Sequence diagram
- State diagram

System Requirements & Business Analysis

System Analysis & Design

Evaluation & Optimization (V&V)

Detailed Design: Software & Hardware

System Integration

System Implementation
SysML in the System Development Process

- System Requirements & Business Analysis
  - Requirements diagram
  - Use Case Diagram
  - Activity diagram

- System Analysis & Design
  - Package diagram
  - Block Definition diagram
  - Activity diagram
  - Sequence diagram
  - State diagram

- Detailed Design: Software & Hardware
  - Package diagram
  - Block Definition diagram
  - Internal Block diagram
  - Parametric diagram
  - Sequence diagram
  - State diagram

- System Integration

- System Implementation
  - Requirements diagram
  - Use Case diagram
SysML in the System Development Process

- Requirements diagram
- Use Case Diagram
- Activity diagram

System Requirements & Business Analysis

- Package diagram
- Block Definition diagram
- Activity diagram
- Sequence diagram
- State diagram

System Analysis & Design

- Package diagram
- Block Definition diagram
- Internal Block diagram
- Parametric diagram
- Sequence diagram
- State diagram

Evaluation & Optimization (V&V)

- Requirements diagram
- Use Case diagram

Detailed Design: Software & Hardware

- Block Definition diagram
- Internal Block diagram
- Package diagram

System Integration

- Package diagram
- Block Definition diagram
- Internal Block diagram
- Parametric diagram
- Sequence diagram
- State diagram

System Implementation
Where To Go To Learn More

• **SysML Web Resources:**
  • [http://www.omgsysml.org/](http://www.omgsysml.org/)
  • SysML specification and other SysML resources
  • [http://www.incose.org/](http://www.incose.org/)
  • International Council on System Engineering web page
  • SysML specification, articles and other resources
  • [http://groups.google.com/group/SysMLforum/topics](http://groups.google.com/group/SysMLforum/topics)
  • Discussion group for topics related to the Systems Modeling Language (SysML)
Requirements Engineering with MagicRQ 14.0
• Stakeholders don’t know what they really want.
• Stakeholders express requirements in their own terms.
• Different stakeholders may have conflicting requirements.
• Organizational and political factors may influence the system requirements.
• The requirements change during the analysis process. New stakeholders may emerge and the business environment changes.
Requirements Modeling

- Requirements decomposition

- Noun phrase analysis
  - Underline all nouns for actors and circle all verbs for use cases
  - Nouns always represent people, organizations, software, hardware data items that interact with our system
  - Verbs always represent actions, things a user can do, or events that can take place

- Requirements modeling
  - Use Case Diagram
  - Activity Diagram
  - Requirement Diagram, etc
Where Do We Keep the Requirements?

• We need to have a repository to keep all the requirements

• Why do we need “Requirements Management System”?
  • Help control change.
  • Improve team communications.
  • Improve compliance with: CMM, CMMi, ISO, IEEE, etc

• Requirements Management Systems such as IBM RequisitePro and Telelogic DOORS
• MagicRQ 14.0 is a MagicDraw plugin to allow full integration with IBM RequisitePro & Telelogic DOORS

• MagicRQ 14.0 works seamlessly between MagicDraw and IBM’s RequisitePro or Telelogic’s DOORS, allowing an alternative UML view in MagicDraw for software analysts and architects

• Once the requirements are loaded into MagicDraw, many additional Traceability, Analysis, Metrics, and reporting facilities are available!
• Is a two-way analysis tool to associate/create model elements from and to requirements

• Maintains the change status between the associated requirements and model elements. It also keeps information about associations such as suspect, deleted, or orphan when the associated elements or requirements are being modified or deleted
Best in class tools:

- UML 2.0 modeling: MagicDraw
- Requirements Management: RequisitePro and DOORS

Linking of Requirements and Design through process

- Considered critical to achieve improvement in software engineering.
- Linking (or ‘traceability’) is considered essential for auditing, compliance issues and to ensure that the deliverables match the requirements.
Thank You for Your Attention

Let’s keep in touch:
training@nomagicasia.com

Awards

- Best Java Database Tool
- Jolt Productivity Winner
- Best Java Modeling Tool
- Best Team Development Tool