The System Modeling Language (SysML) and the SYSMOD modeling approach

Polgár Balázs
Ákos Horváth

Model Driven Software Development
Lecture 10
Acknowledgement

- Portions of this presentation are from
  - *Systems Engineering with SysML/UML*, by Tim Weilkiens, published by Morgan Kaufmann Publishers, Copyright 2007 Elsevier Inc. All rights reserved.
  - *A Practical Guide to SysML*, by Sanford Friedenthal, Alan Moore, and Rick Steiner, published by Morgan Kaufmann Publishers, Copyright 2009 Elsevier Inc. All rights reserved.
  - IBM course, Requirements management
Overview

- Context

- SysML Overview

- SysML details + the SYSMOD Systems Engineering Methodology
Systems Engineering

- Systems Engineering is a multidisciplinary approach to develop balanced system solutions in response to diverse stakeholder needs
- ~ Integration Engineering
  - Software engineering
  - Hardware engineering
  - Mechanical engineering
  - Safety engineering
  - Security engineering
  - ...
- ~ Process Engineering
- System
  - Military, airplane, car, aviation, railway interlocking, notebook, etc.
Systems Engineering Process

- V-model
V-model – v2

EN50129
### Differentiating Problem and Solution

<table>
<thead>
<tr>
<th><strong>Problem</strong></th>
<th><strong>Solution</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stakeholder requirements</strong></td>
<td><strong>System requirements</strong></td>
</tr>
<tr>
<td>• A description of the problem</td>
<td>• An abstract representation of</td>
</tr>
<tr>
<td>and its context</td>
<td>the solution</td>
</tr>
<tr>
<td>• Describes what stakeholders</td>
<td>• Describes what the system</td>
</tr>
<tr>
<td>want from the system</td>
<td>will do</td>
</tr>
<tr>
<td>• Not the definition of the</td>
<td>• Not the definition of the</td>
</tr>
<tr>
<td>solution (except for environment)</td>
<td>design</td>
</tr>
<tr>
<td>• Quality of results</td>
<td>• How well it does it</td>
</tr>
<tr>
<td>• Created by stakeholders</td>
<td>• Created by systems engineers</td>
</tr>
</tbody>
</table>

“The user shall be able to ....”

“The system shall do ....”
Models Bridge Layers of Requirements

- Vision
  - e.g. Goal / Usage modeling
- Stakeholder Requirements
  - e.g. Functional modeling
- System Requirements
  - e.g. Performance modeling
- Subsystem Requirements
Importance of Traceability

Impact Analysis

Vision satisfies Stakeholder Requirements

Stakeholder Requirements satisfies System Requirements

System Requirements satisfies Architecture

Architecture satisfies Acceptance Strategy

Acceptance Strategy satisfies Operational Use

Accepting the product satisfies Validating the product

Validating the product satisfies Verifying the product

Verifying the product satisfies Evaluating subsystems

What if ...?
Importance of Traceability

Coverage analysis

Vision

Satisfaction

Stakeholder Requirements

Satisfaction

System Requirements

Satisfaction

Architecture

Operational Use

Acceptance Strategy

System Test

Acceptance Strategy

%s Complete ...?
Overview

- Context

- **SysML Overview**
  - SysML details + the SYSMOD Systems Engineering Methodology
SysML overview

- „UML for Systems Engineering”
  - Supports the specification, analysis, design, verification and validation of systems that include hardware, software, data, personnel, procedures, and facilities

- Developed by OMG and International Council on Systems Engineering (INCOSE)

- OMG SysML™ (http://www.omgsysml.org)
  - RFP – March 2003
  - Version 1.0 – September 2007
  - Version 1.1 – November 2008
  - Version 1.2 – June 2010
  - Version 1.3 – June 2012
Relationship Between SysML and UML

UML 2

SysML

UML4SysML

UML reused by SysML

SysML extensions to UML

UML not required by SysML (UML - UML4SysML)

SysML Profile
SysML Diagram Taxonomy

SysML Diagram

- Behavior Diagram
  - Activity Diagram
  - Sequence Diagram
  - State Machine Diagram
  - Use Case Diagram

- Requirement Diagram
  - Block Definition Diagram
  - Internal Block Diagram

- Structure Diagram
  - Package Diagram

Legend:
- Same as UML 2
- Modified from UML 2
- New diagram type
Aspects of SysML

**Diagram**
- Structure
  - Block definition diagram
  - Internal block diagram
  - Parametric diagram
  - Package diagram
- Behavior
  - Activity diagram
  - Use case diagram
  - State machine diagram
  - Sequence diagram

**Model**
- Structure model
- Behavior model

**Other**
- Requirement diagram, stereotype, model view, AP-233, XMI Metadata Interchange format
Overview

- Context

- SysML Overview

- SysML details + the SYSMOD Systems Engineering Methodology
Language vs. Methodology

- **Modeling Language**
  - Defines elements and their relationship
  - Defines syntax and semantics
  - *What type of elements can be used during modeling?*
  - E.g. SysML

- **Development Methodology**
  - Defines the steps of analyzing and designing the system
  - Defines the usage of the model elements and diagrams
  - *How shall the model be built?*
  - E.g. SYSMOD (SYStem MODeling) by Tim Weilkiens
The SYSMOD approach

- Analyzing Requirements
The SYSMOD approach for design
SysML

Represents behavior in terms of the ordering of actions based on the availability of inputs, outputs, and control, and how the actions transform the inputs to outputs.
Describe Project Context

- Car rental system
  - Works without staff
    - Customer identification needed
  - Central computer in radio compartment
    - Communicates with central reservation system
    - Collects usage data
    - Comfort features
      - Navigation
      - Radio
      - Phone
      - ...

**Determine requirements**

1. **Project context**
2. **Identify Stakeholders**
3. **Collect requirements**
4. **Requirements**

**Diagram:**
- **Act Analysis**
- **Describe project context**
- **Project context**
- **Requirements**
- **System processes**
- **Model system context**
- **Use cases**
- **Model domain knowledge**
- **Domain knowledge**
- **System context**
- **Glossary**
## Identify stakeholders

<table>
<thead>
<tr>
<th>Stakeholder</th>
<th>Priority (1–4)</th>
<th>Comments/Interests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>1</td>
<td>Wants easy and comfortable access to a car and low prices.</td>
</tr>
<tr>
<td>Reservation system</td>
<td>2</td>
<td>Requires interface to the on-board computer.</td>
</tr>
<tr>
<td>Car manufacturer</td>
<td>1</td>
<td>The on-board computer must control the central locking system and the drive-away protection, and collect mileage information.</td>
</tr>
<tr>
<td>Cellular communication vendor</td>
<td>1</td>
<td>The on-board computer and the reservation system will presumably communicate via SMS. Both speed and availability must be ensured.</td>
</tr>
<tr>
<td>Insurance company</td>
<td>1</td>
<td>Is break-in protection coverage for the on-board computer sufficient?</td>
</tr>
<tr>
<td>Car service</td>
<td>2</td>
<td>Installation, maintenance, and configuration of the on-board computer.</td>
</tr>
<tr>
<td>SpeedyCar call center</td>
<td>2</td>
<td>Handles customer enquiries with regard to the on-board computer’s operation.</td>
</tr>
<tr>
<td>Navigation system manufacturer</td>
<td>4</td>
<td>SpeedyCar wants the on-board computer to have navigation system functionality.</td>
</tr>
<tr>
<td>Car radio manufacturer</td>
<td>2</td>
<td>The on-board computer should integrate car radio functionality since it will replace the regular radio.</td>
</tr>
<tr>
<td>Card reader manufacturer</td>
<td>1</td>
<td>The access device will be purchased from third party.</td>
</tr>
<tr>
<td>Legacy systems takeback law</td>
<td>3</td>
<td>What does the law say about the disposal of old devices? Who is responsible?</td>
</tr>
<tr>
<td>Lawmaker</td>
<td>1</td>
<td>What size/weight is permitted for the on-board computer? Other legal provisions have to be checked yet.</td>
</tr>
</tbody>
</table>

![SYSMOD Diagram](image)
Identify stakeholders

uc [package] stakeholders [selection]

- Customer
- Principal
- Lawmaker
- Marketing principal
Collect requirements

**req [package] Functional requirements [selection]**

- **Car usage without staff**
  - «functionalRequirement»
  - id="REQ1"
  - text="Cars to be picked up and returned by the customers without assistance."

- **Identify customer**
  - «functionalRequirement»
  - id="REQ1.1"
  - text="The system must do unique authorization of customers."

- **Start using car**
  - «functionalRequirement»
  - id="REQ1.2"
  - text="The system has to allow a customer to start using the car."

- **Grant access to car**
  - «functionalRequirement»
  - id="REQ1.7"
  - text="The system must be able to grant authorized customer access to car."

- **End usage of car**
  - «functionalRequirement»
  - id="REQ1.3"
  - text="The system must be able to allow customer to end usage of car."
SysML

Requirements diagram

**Type of context element**: req [package], Functional requirements, selection

**Name of context element**: {functionalRequirement}

**Name of diagram**: Requirements diagram

**Requirement composition**:

- **Identify customer**
  - id="REQ1.1"
  - text="The system must do unique authorization of customers."

- **Start using car**
  - id="REQ1.2"
  - text="The system must allow a customer to start using the car."

- **Grant access to car**
  - id="REQ1.7"
  - text="The system must be able to grant authorized customer access to car."

- **End usage of car**
  - id="REQ1.3"
  - text="The system must be able to allow customer to end usage of car."

**Requirements diagram**

Represents text-based requirements and their relationship with other requirements, design elements, and test cases to support requirements traceability.
Collect requirements

**req [package] Requirements [essential/technical]**

**Essential Requirements**
- **usabilityRequirement** Operability during driving
  - id = “REQ13”
  - text = “The system has to be operated by the driver during driving.”
- **functionalRequirement** Comfort functions

**Technical Requirements**
- **usabilityRequirement** Enclosure size of central unit
- **functionalRequirement** Radio
- **usabilityRequirement** Central unit display and keypad
- **functionalRequirement** Navigation system
- **functionalRequirement** Phone
Model System Context

**act** Model system context

**Identify system actors**

**Model system/actors information flow**

**Identify system interaction points**

**System context**

**Requirements**

**SYSMOD**

1. Describe project context
2. Determine requirements
3. Project context
4. Requirements
5. System processes
6. Use cases
7. Model use cases
8. System context
9. Use cases
10. Model domain knowledge
11. Domain knowledge
12. System context
13. Glossary

**Domain knowledge**

**System context**

**Glossary**

31
Identify System Actors

On-board computer

- Customer
- Car management system
- Car service employee
- Battery
- Windshield
- Billing system
- Car ignition
- Car movement data
- Car drive-away protection
- Central locking system

Temperature

Reservation system

Status request "flow"

receives usage data via SMS upon car return

External systems of car

Diagram:

- System context
- Identify system actors
- Model system context
- Model use cases
- System processes
- Use cases
- Domain knowledge
- Model system context
- Glossary
Block Definition Diagram

Represents structural elements called blocks, and their composition and classification
Model System-Actor
Information Flow
Identify System Interaction Points

**bdd [package] System context [interaction points]**

- CurrentPort
- CarPort
- KeyPort
- ServicePort
- ServerPort
- «System»
- On-board computer
- IOPort
- CarReaderPort
- CarControlPort
- MicrophonePort

**SYSMOD Diagram**

- Act Analysis
- Describe project context
- Determine requirements
- Project context
- Requirements
- System context
- Model system context
- System processes
- Model use cases
- Use cases
- Use cases (object flow)
- System context
- Domain knowledge
- Model domain knowledge
- Create glossary
- Glossary
- Identify system interaction points
- Model system/actors
- Information flow
- System context
System Context
Model Use Cases

- **act Model use cases**
- **System context**
- **Identify use cases**
- **Use cases**
- **Describe use cases essences**
- **Use cases**
- **Describe system processes**
- **Use cases**
- **Model use case flows**
- **Use cases**
- **Model use cases without redundancies**
- **Use cases**
- **Model object flow**
- **Use cases**
- **System processes**
- **Create glossary**
- **Model domain knowledge**
- **Domain knowledge**
- **System context**
Model Use Cases

**uc [package] Car usage [start car usage]**

Customer → Card data, PIN → Usage right → Reservation System → Start car usage

**uc [package] Car usage [show car usage data]**

Customer → Car movement data → Usage data → Show car usage data
SysML

Use Case Diagram

Represents functionality in terms of how a system or other entity is used by external entities (i.e., actors) to accomplish a set of goals.
Model Use Cases

Organizing use cases into packages

pkg Use case packages

Service

Poll system status
Start on-board computer
Configure on-board computer

Do system test

Car usage

Start car usage
End car usage
Show car usage data

Navigation system

Radio

Phone
SysML

Requirements traceability

req [package] Functional requirements [car usage without staff]

«requirement»
Car usage without staff

«refine»
Start car usage

«refine»
End car usage

uc [package] Car usage [non-functional requirements]

Start car usage

«trace»
«interfaceRequirement» Reservation system

«trace»
«performanceRequirement» Period unlock door
Describe Use Cases Essences

Start car usage

Pragmatic description

Apply customer card
Read card
Send SMS to reservation center
Receive SMS reply
Check usage right
Unlock car doors via central locking system
Poll PIN
Enter PIN
Verify PIN
Deactivate electronic drive-away protection
Remove key
Display customer welcome message
Describe Use Cases Essences

Start car usage

Pragmatic description

- Apply customer card
- Read card
- Send SMS to reservation center
- Receive SMS reply
- Check usage right
- Unlock car doors via central locking system
- Poll PIN
- Enter PIN
- Verify PIN
- Deactivate electronic drive-away protection
- Remove key
- Display customer welcome message

Essential description

- Identify customer
- Check usage right
- Unlock car
- Check disposal right
- Start car usage
- Confirm usage start
Describe Use Cases Essences

**Essence**
- Identify customer
- Check usage right
- Unlock car
- Check disposal right
- Activate car
- Start car usage
- Confirm usage start
- Open points

**Open points**
- Planned timeouts are still unclear.
- Does every car have a central locking system and drive-away protection?
Model Use Case Flows

**act** Start car usage [without object flow]

- «precondition» Car is not in use
  - «essential» Identify customer
    - [customer not identified]
    - [customer identified]
  - «essential» Check usage right
    - [usage not authorized]
      - «essential» Unlock car
        - Terminate if no PIN is entered and key is not removed within a defined time window after the car is unlocked.
  - Signal remove key
    - «essential» Check disposal right
      - [disposal right not o.k.]
      - [disposal right o.k.]
  - «essential» Start car usage
  - «essential» Activate car
    - [disposal right not o.k.]
  - «essential» Confirm usage start
    - Ask customer to get out and lock car; car will lock automatically after a defined time.
    - Car usage started
    - Car usage not started

**act** Model use cases

**Act** Analysis

- Project context
- Determine requirements
- Model system context
- Create glossary

**Act** Text

- System context
- Requirements

**Act** Use cases

- Identify use cases
- Describe use cases
- Describe system processes

**Act** Use cases

- System processes
- Use cases (essential)
- Use cases (non-essential)
- Use cases (detailed)
- Use cases (non-redundant)

**Act** System

- Use cases
- System processes
Model Domain Knowledge
Create Glossary

Table 2.17 Glossary entries.

<table>
<thead>
<tr>
<th>Usage right</th>
<th>Description:</th>
<th>Domain block:</th>
<th>Author, last change:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>A usage right describes whether or not a customer is entitled to use a car. It includes information about the customer, the booking period, and whether it is an emergency driving case.</td>
<td>Yes</td>
<td>Tim Weilkiens, April 30, 2004</td>
</tr>
<tr>
<td>Domain block:</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Author, last change:</td>
<td>Tim Weilkiens, April 30, 2004</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Supports better understanding
The SYSMOD approach for design

SYSMOD
Model System/Actor Interaction

sd [block] On-board computer context [system/actors start car usage]

- Customer
- Reservation system
- Central locking system
- On-board computer
- Drive-away protection

Apply card

SigLED(yellow, blinking=true)

Request usage right

Usage right

SigLED(green, blinking=false)

Unlock car

par

PIN request

PIN

Deactivate drive-away protection

SigKey(removed=true)
Sequence Diagram
Represents behavior in terms of a sequence of messages exchanged between parts.
Derive System Interfaces

**bdd [package] On-board computer context [interfaces start car usage]**

- IUsageRight
- IReservationSystem
- ServerPort
- CarPort: Vibration
- CarControlPort
- ICustomer
- IStatusDisplay
- CardReaderPort
- Input
- IOutput
- IKey

**bdd [package] Interfaces [CardReaderPort]**

- **«interface» ICustomer**
  - Apply card
- **«interface» IStatusDisplay**
  - «signal» SigLED(f: Color, blinking: Boolean)
- **«enumeration» Color**
  - red
  - yellow
  - green
Model System Structures

Identify system components

```
bdd [package] On-board computer [section]

«system»
On-board computer

«block»
Customer card

«block»
On-board computer control

«subsystem»
Navigation system

«block»
Card reader

«block»
Communication unit

«block»
GPS antenna
```
Model System Structures

Identify relation of components
Internal Block Diagram

Represents interconnection and interfaces between the parts of a block.
Model System Structures

Structural allocation

Diagram showing structural allocation of components such as Card reader and Customer card with allocated ports and interfaces.
Model System Structures

Functional allocation:
Actions to System components
Model System Structures

Traceability to requirements:
Blocks satisfying a req.

req [package] Requirements [Authenticate customer]

«functionalRequirement» Authenticate and authorize customer
«satisfy» «block» Card reader
«satisfy» «block» Customer card
Complete

req [package] Requirements [context car usage without staff]

«requirement» Car usage without staff
«refine» «satisfy» «verify» «testCase»
Start car usage
TC1: Customer not identified
TC2: Customer identified

«block» On-board computer control

Derive State Model
Derive State Model

- On-board computer [standard flow start car usage]
  - Customer card: passive
  - Card reader: idle
  - On-board computer control: free

- Usage handover:
  - Check usage right
  - Unlock car
  - Check disposal right
  - Start car usage

- In use

- System context:
  - Model system/actor interaction
  - Use cases
  - System

- On-board computer control:
  - free
    - Card data
    - Return completed
      - [oc1 isInState(Return)]
        - Pickup
          - Pickup completed
            - Return
              - Engine off

- Use comfort function

- in use

- Pickup
  - Pickup completed
    - Return completed

- Return
State Machine Diagram
Represents behavior of an entity in terms of its transitions between states triggered by events
**Parametrics**

- **Constraints on block properties**

[Diagram of constraints on block properties]

- **bdd [package] Newton's world**
  - «block» Apple
    - values
      - d: Density
      - v: Volume
    - + apples
  - «block» Planet
    - + newton's tree
  - «block» Apple tree
    - values
      - height: Length
    - + trees
  - «block» Human
    - values
      - impact: Energy

- **bdd [package] Constraint library [Newton's world]**
  - «constraint» Mass relationship
    - constraints
      - \( m = v^2 d \)
    - parameters
      - m: Mass
      - v: Volume
      - d: Density
  - «constraint» Newton's law
    - constraints
      - \( f = m^2 a \)
    - parameters
      - f: Force
      - m: Mass
      - a: Acceleration
  - «constraint» Potential energy
    - constraints
      - \( E = f^2 h \)
    - parameters
      - f: Force
      - h: Height

- **par [block] Newton's world**
  - Apple
    - Tree.height
  - Human
    - impact: Energy
  - mb: Mass relationship
    - m:
    - v:
    - d:
  - gN: Newton's law
    - f:
    - a:
  - Apple
    - volume
  - Apple
    - density
  - Planet
    - gravitation
Parametric Diagram

Represents constraints on property values used to support engineering analysis.
Summary

- **SysML**
  - OMG’s most widely accepted standard
  - Heavily used by embedded system engineerings
  - Reuses the „better” part of UML

- **SYSMOD**
  - Provides a framework for MDE of embedded systems
  - (MDA) $\rightarrow$ more focused and fits to developer needs