Hochschule Konstanz
Fakultät Elektrotechnik
und Informationstechnik

Systems Modeling Language (SysML)

Mysam Haj Ali, Chin-I Feng

Content

- Systems
- Systems Engineering
- SIMILAR Process Modell
- MBSE
- Introduction to SysML
- Main similarities and differences to the UML
- SysML Diagram Types
- SysML Tools and Applications
- Using SysML with UML
- Example of Using SysML with UML
- SysML in RTS
- SysML Usage Consequences
- SysML pros and cons
- Summary

Systems

- What is a System?

A system is a human-created artifact consisting of system components that work together to achieve a goal that cannot be achieved by individual elements.

A component can consist of software, hardware, people, or any other units. [1]

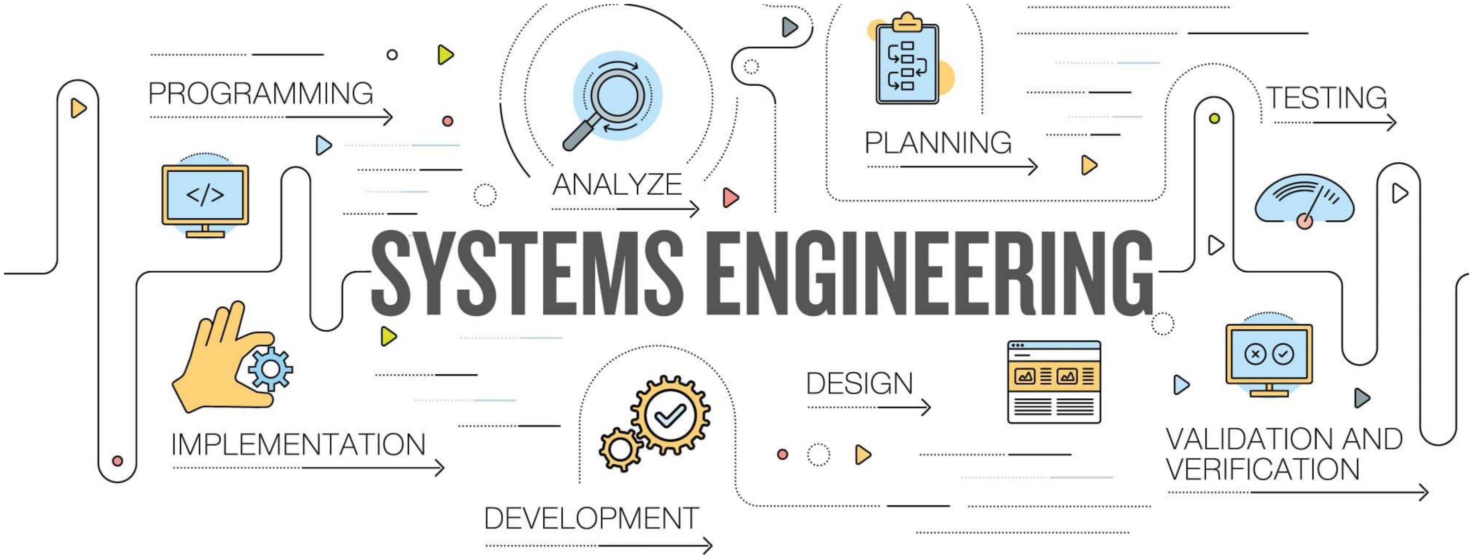
System Engineering

- What is Systems Engineering?

Systems engineering is an interdisciplinary approach and focuses on the definition and documentation of system requirements in the early development phase, the elaboration of a system architecture and the verification of the system for compliance with the set requirements, taking into account the overall problem: operation, time, test, deployment, cost & planning, training & maintenance and disposal.[1]

System Engineering

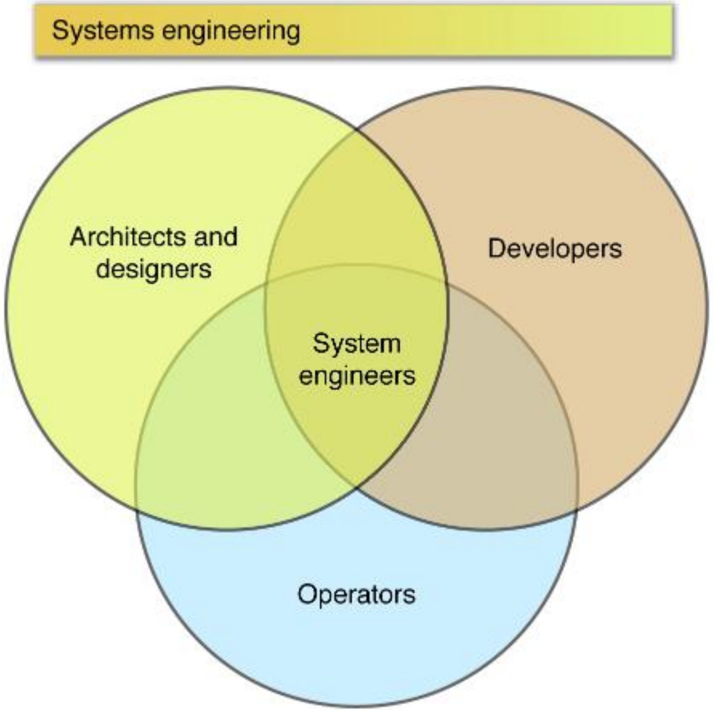
Integration of many disciplines to describe a structured development process.



[2]

System Engineers

- A link between different development departments.
- Whole system design.
- Overall architecture of systems.
- Big picture thinking.



[3]

System Engineering Processes

SIMILAR Process Modell:

State the Problem.

Investigate Alternatives.

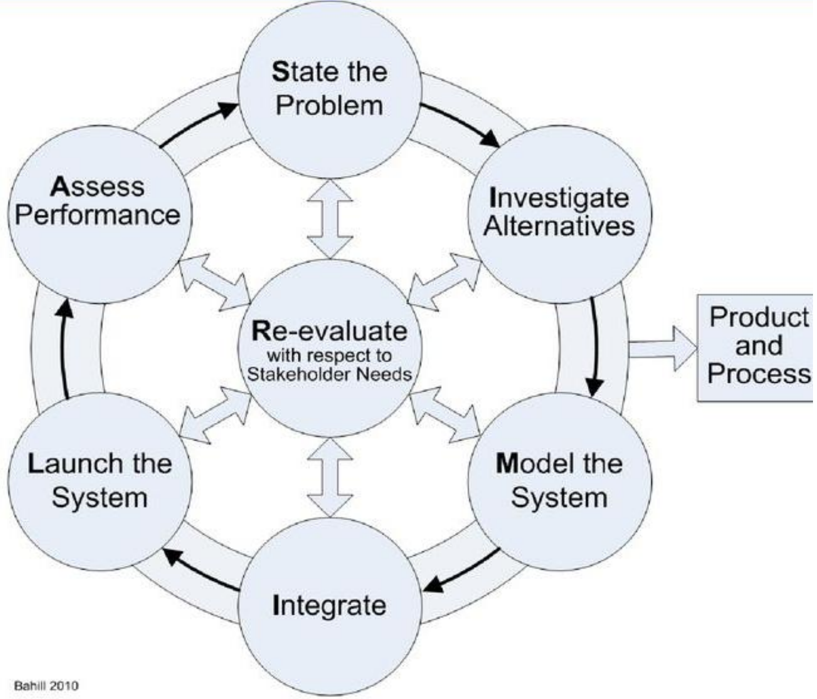
Model the System.

Integrate.

Launch the System.

Assess Performance.

Re-Evaluation.



The SIMILAR Process

[5]

MBSE

- What is **Model-based** Systems Engineering?

Model-Based Systems Engineering (MBSE) is the formalized application of modeling to support system requirements, architecture, analysis, verification and validation activities from the beginning of the conceptual architecture phase through development to the later phases of the system life cycle. [6]

Introduction

Definition and Purpose

- **Definition:**

SysML (**Sys**tems **M**odeling **L**anguage) is a specialized modeling language for complex systems engineering analysis, design, and verification, extending UML for broader application.

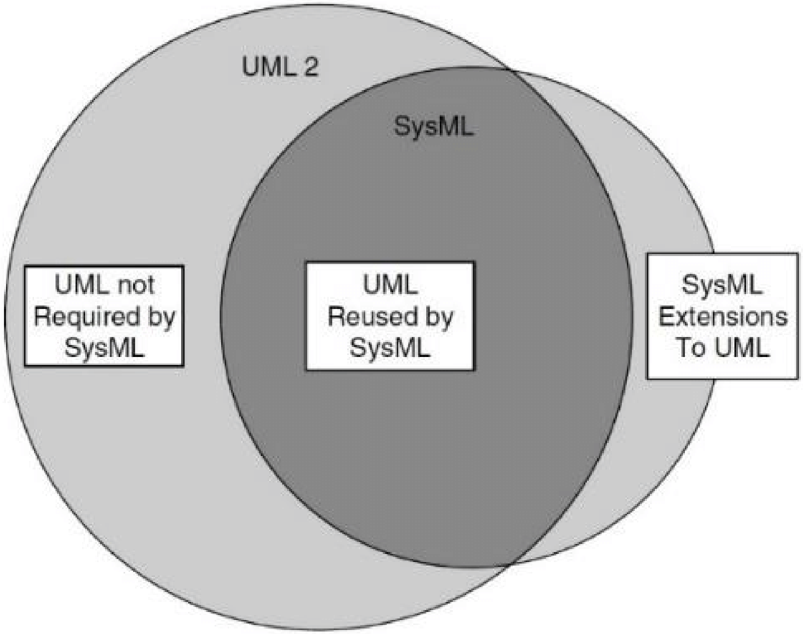
- **Purpose:**

Provide a standardized visual language for representing and communicating the structure, behavior, and requirements of a system.

Relationship

between SysML and UML

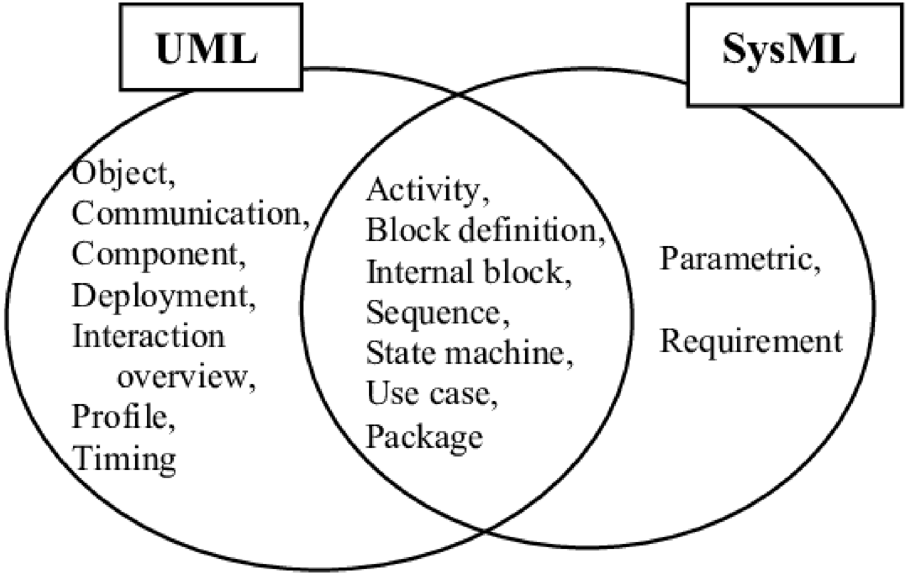
- SysML is an UML extension designed for systems engineers, focusing on systems engineering with both hardware and software components



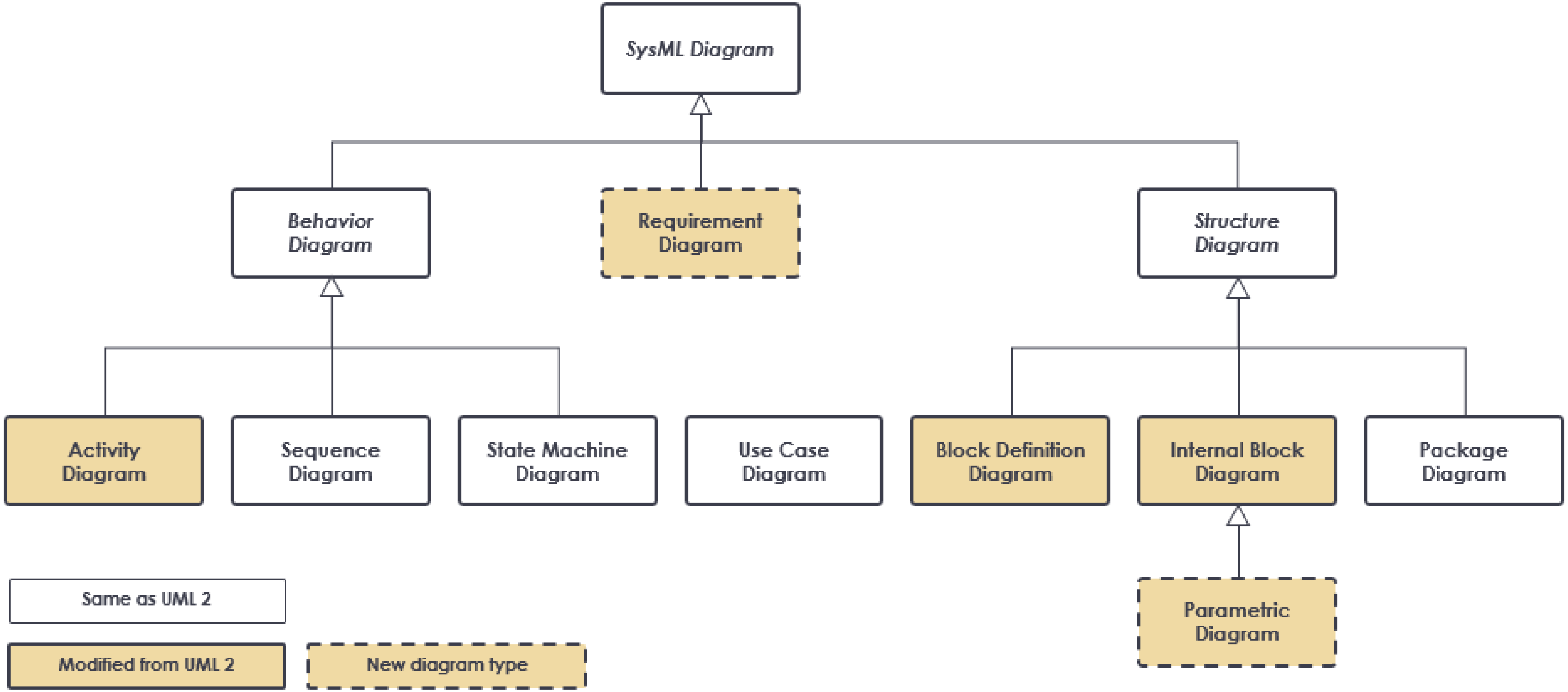
Differences

between SysML and UML

- SysML focuses on systems engineering, offers fewer diagram types compared to UML, but includes unique diagrams such as requirements and parametric diagrams to better support the engineering process



SysML Diagram Types



Requirement Diagrams

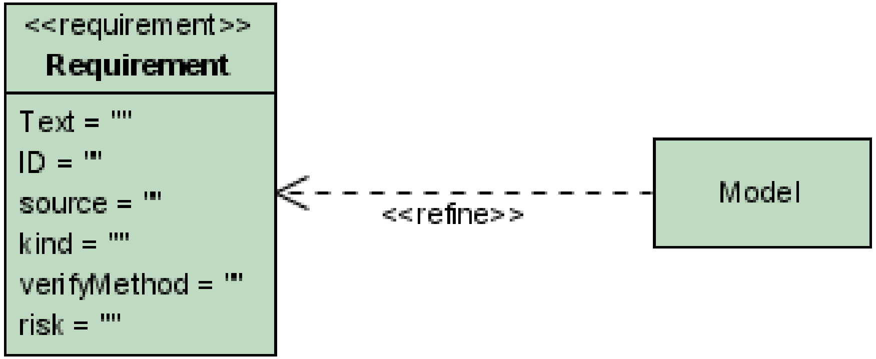
Purpose and Benefits

- **Purpose**:

Capture and organize system requirements, allowing traceability and impact analysis

- **Benefits**:

1. Clear communication of requirements among stakeholders
2. Systematic organization of requirements and their relationships
3. Improved change management and traceability



Requirement Diagrams

Basic Notation and Elements

- **Requirement**:
Rectangular shapes representing system requirements or constraints

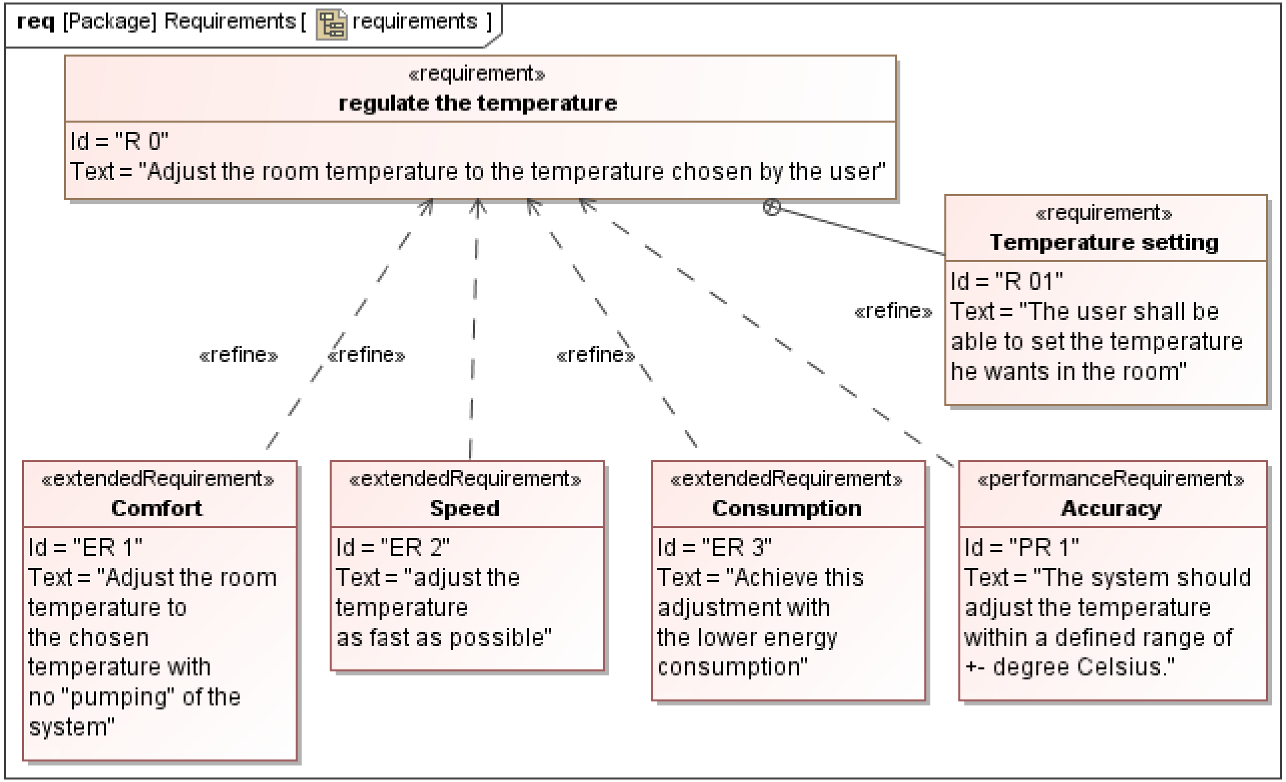
- **Containment**:
Solid lines with an arrowhead, indicating hierarchical relationships between requirements

- **Refinement:**
Dashed lines with an arrowhead, linking a requirement to a model element providing additional details

- **Relationships**:
Describe common relationships between requirements
  - refine: Represents a detailed specification of a requirement
  - verify: Denotes confirmation of a requirement via test case

Requirement Diagrams

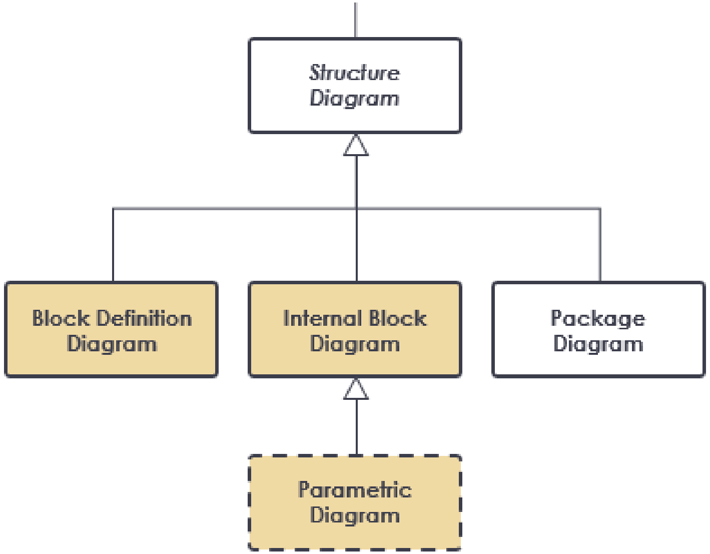
Example: Regulate the Temperature



req [Package] Requirements [requirements]
«requirement»
regulate the temperature
Id = "R 0"
Text = "Adjust the room temperature to the temperature chosen by the user"
«requirement»
Temperature setting
Id = "R 01"
Text = "The user shall be able to set the temperature he wants in the room"
«refine»
«refine»
«refine»
«refine»
«extendedRequirement»
Comfort
Id = "ER 1"
Text = "Adjust the room temperature to the chosen temperature with no "pumping" of the system"
«extendedRequirement»
Speed
Id = "ER 2"
Text = "adjust the temperature as fast as possible"
«extendedRequirement»
Consumption
Id = "ER 3"
Text = "Achieve this adjustment with the lower energy consumption"
«performanceRequirement»
Accuracy
Id = "PR 1"
Text = "The system should adjust the temperature within a defined range of +- degree Celsius."

SysML Diagram Types

Structure Diagrams



Block Definition Diagrams

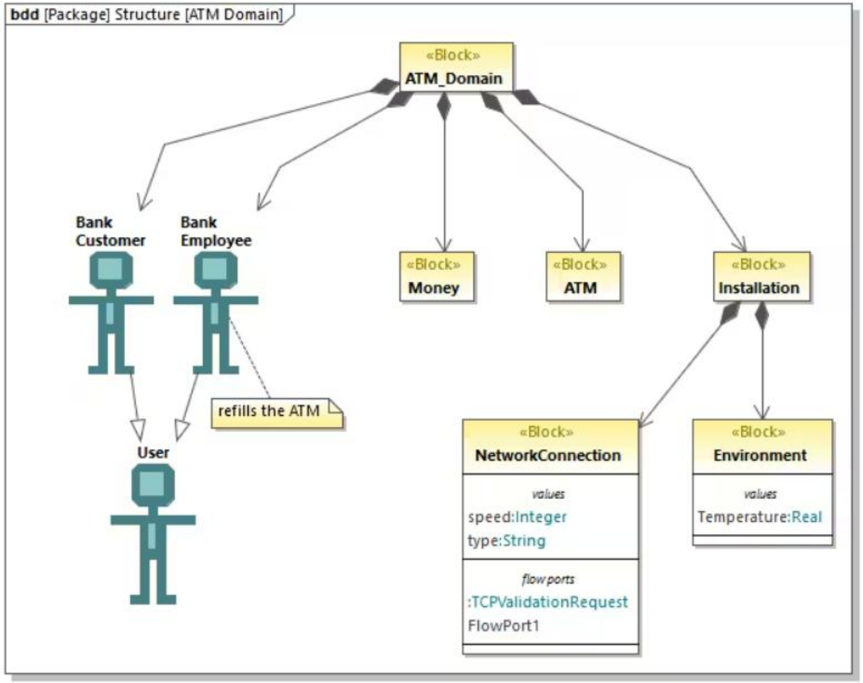
Purpose and Benefits

- **Purpose**:

Depict the static structure of a system, including its components, relationships, and properties

- **Benefits**:

1. Clear visualization of system architecture
2. Identification of system components and their relationships
3. Facilitation of component reuse and modularity



Block Definition Diagrams

Basic Notation and Elements

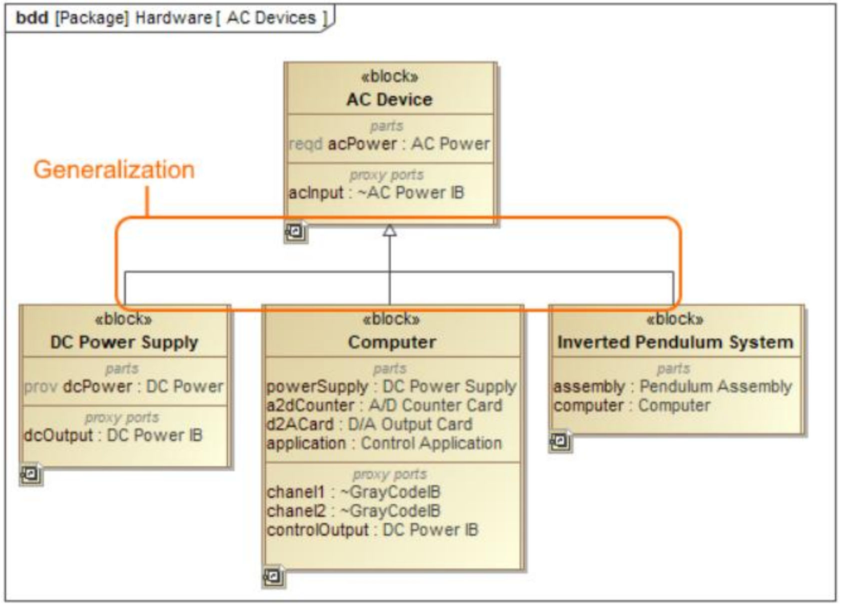
- **Block**:
  System components or entities

- **Ports**:
  Interaction points between blocks

- **Connectors**:
  Relationships and communication paths

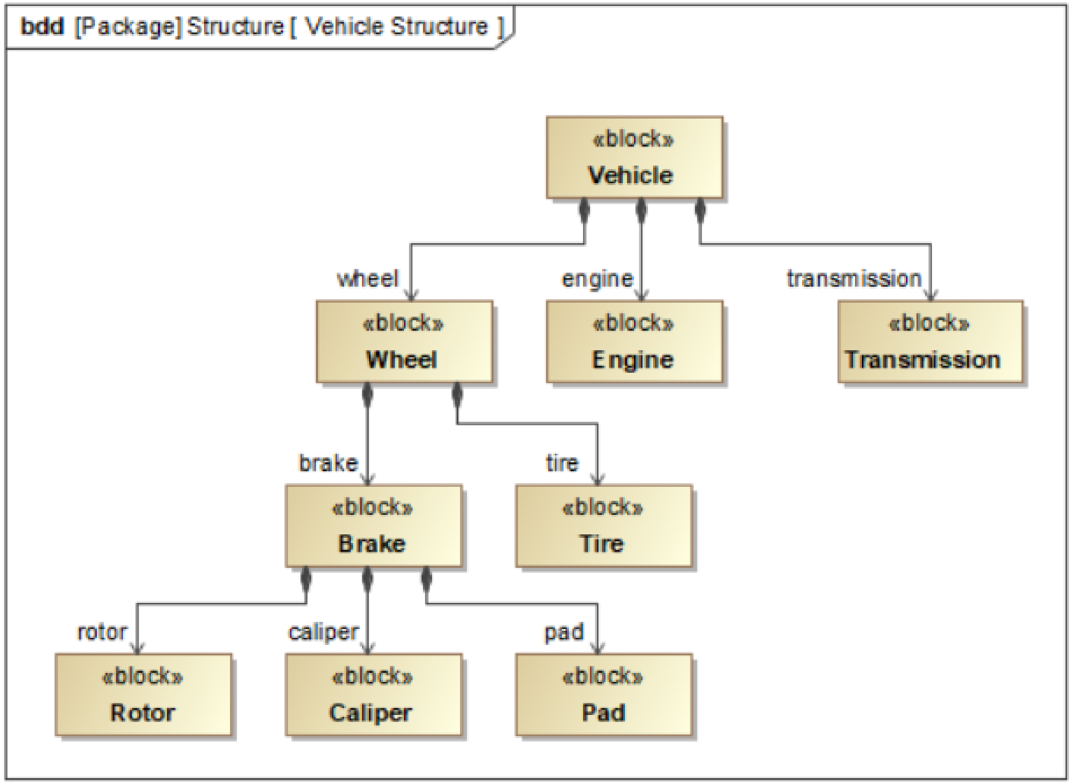
- **Compartments**:
  Organize block properties

- **Inheritance**:
  Generalization and specialization relationships



Block Definition Diagrams

Example: Vehicle Structure



Internal Block Diagram

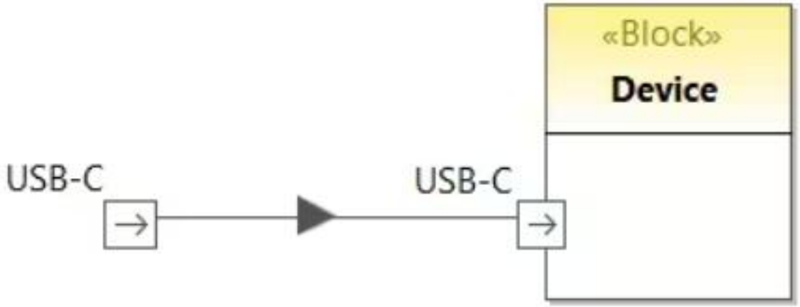
Purpose and Benefits

- **Purpose**:

Illustrate a system's internal structure, highlighting connections and interactions between components, and enabling better analysis and design of the system's configuration

- **Benefits**:

1. Visualizes system's internal configuration
2. Illustrates component interactions and data flow
3. Supports system analysis and design



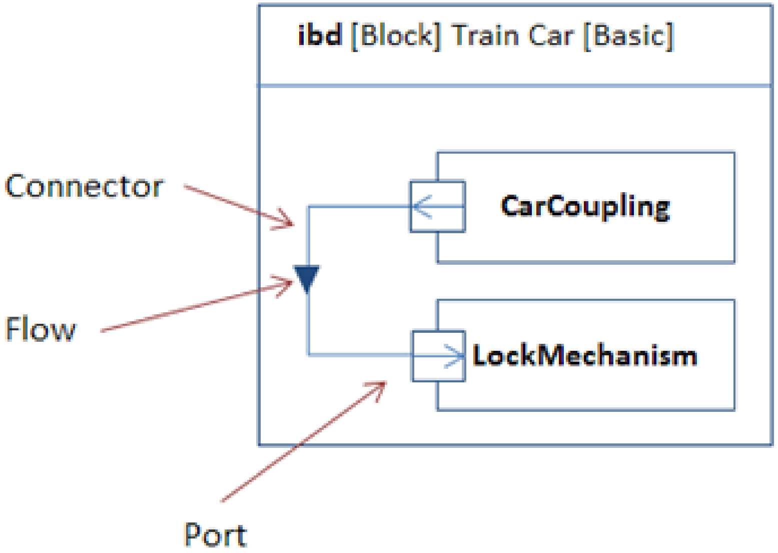
Internal Block Diagram

Basic Notation and Elements

- **Block**:
  Reiterate the definition of a block

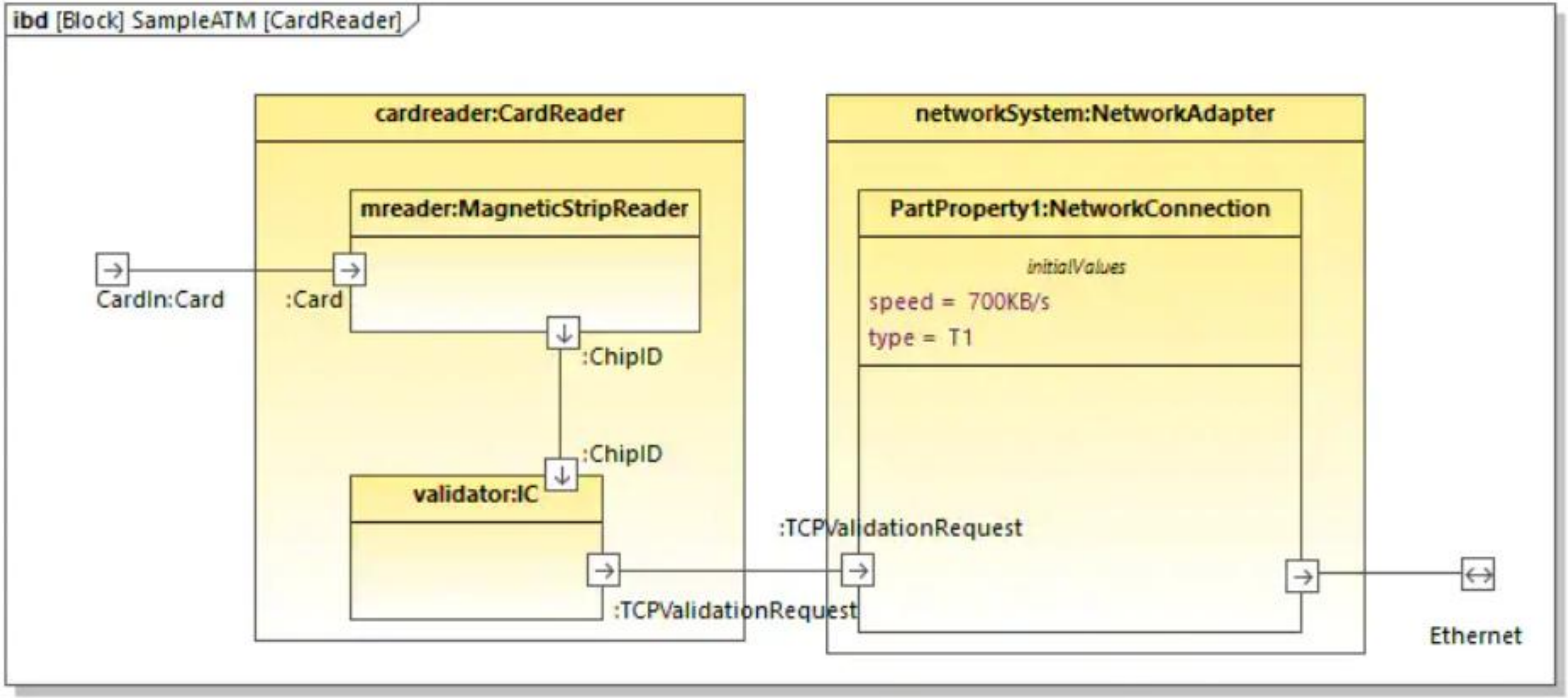
- **Ports**:
  Interaction points between components

- **Connectors**:
  Links between ports to establish data flow



Internal Block Diagram

Example: Card Reader



Parametric Diagrams

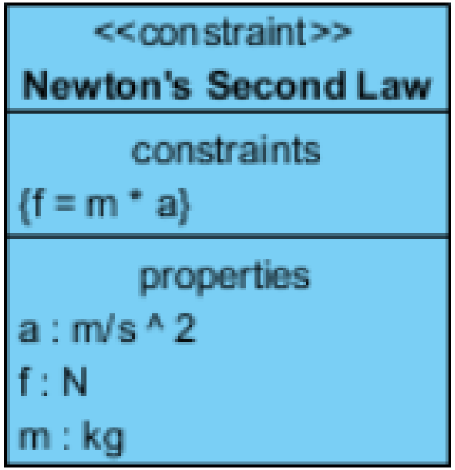
Purpose and Benefits

- **Purpose**:

Represent system constraints and dependencies for quantitative analysis and performance optimization

- **Benefits**:

1. Enables performance and constraint analysis
2. Facilitates trade-off studies and optimization
3. Supports system validation and verification



<<constraint>>

Newton's Second Law

constraints

{f = m * a}

properties

a : m/s ^ 2

f : N

m : kg

Parametric Diagrams

Basic Notation and Elements

- **Constraint Blocks**:

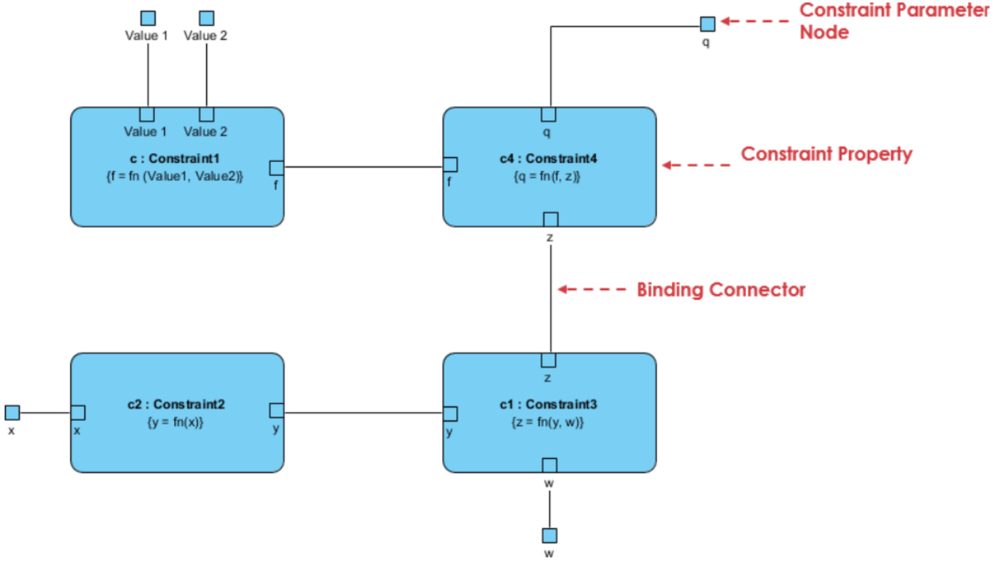
Encapsulate mathematical equations

- **Parameters**:

Represent inputs and outputs of constraint blocks

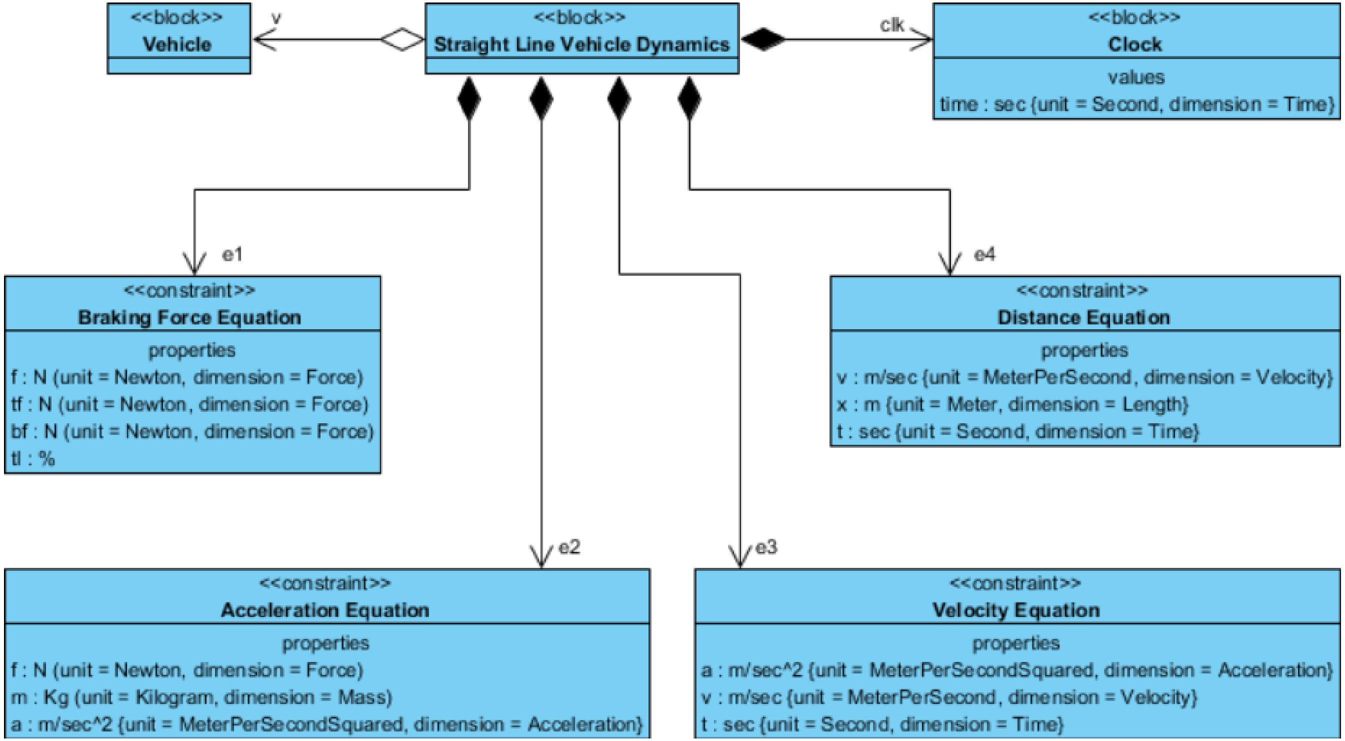
- **Bindings**:

Connect parameters to establish dependencies



Parametric Diagrams

Example: Vehicle



<<block>>
Vehicle

<<block>>
Straight Line Vehicle Dynamics

<<block>>
Clock
values
time : sec {unit = Second, dimension = Time}

<<constraint>>
Braking Force Equation
properties
f : N (unit = Newton, dimension = Force)
tf : N (unit = Newton, dimension = Force)
bf : N (unit = Newton, dimension = Force)
tl : %

<<constraint>>
Distance Equation
properties
v : m/sec {unit = MeterPerSecond, dimension = Velocity}
x : m {unit = Meter, dimension = Length}
t : sec {unit = Second, dimension = Time}

<<constraint>>
Acceleration Equation
properties
f : N (unit = Newton, dimension = Force)
m : Kg (unit = Kilogram, dimension = Mass)
a : m/sec^2 {unit = MeterPerSecondSquared, dimension = Acceleration}

<<constraint>>
Velocity Equation
properties
a : m/sec^2 {unit = MeterPerSecondSquared, dimension = Acceleration}
v : m/sec {unit = MeterPerSecond, dimension = Velocity}
t : sec {unit = Second, dimension = Time}

SysML Tools and Applications

- **Popular SysML Tools**:
  - Enterprise Architect
  - Visual Paradigm
  - MagicDraw

- **Open-Source Option**:
  - Papyrus

> **Selecting the Right Tool**:

Consider project needs, evaluate team preferences, assess tool capabilities & support

SysML in Real-Time Systems

- Important for RTS is satisfying their time constrainers while their behavior executes.

- SysML is designed for modeling systems on a high level of abstraction:

It focuses on system's overall structurer, behavior and requirements.

- Best practice for more detailed systems:

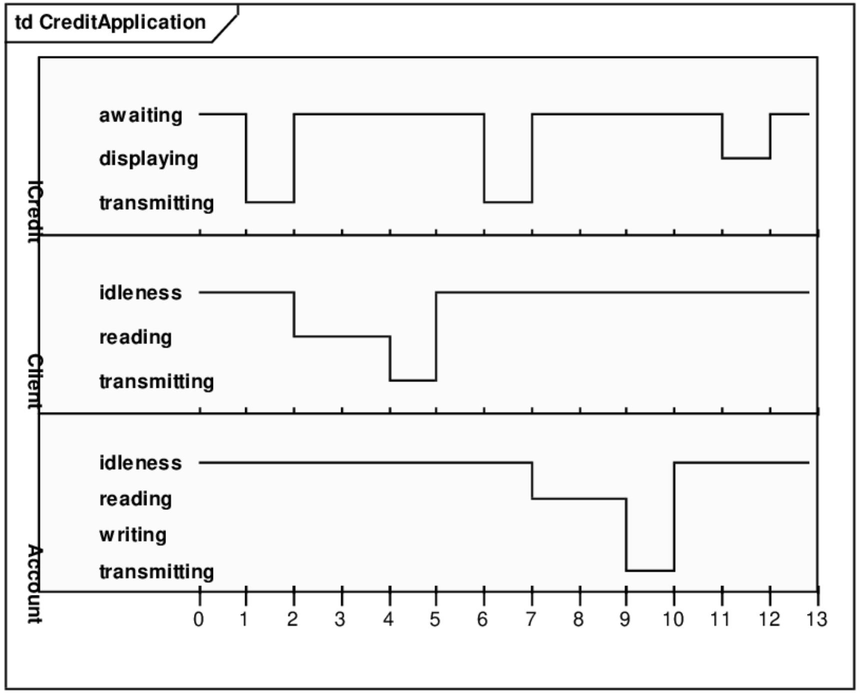
A combination of SysML, UML and other MBD languages.

UML 2.0 in RTS

- Best practice for more detailed real time Systems:

UML 2.0.

- Provides a profile for schedulablity, performance and time.
- New:
  - data types: Time and TimeExpression.
  - diagram: Timing Diagram



[7]

SysML with UML

A combination of SysML, UML is possible even in one model:

- At a higher level of abstraction: SysML.
- At a lower level of abstraction: UML.

Example: Marketplace

- In SysML:
  - Use case diagram: Actors (buyers, sellers, administrators, and their interactions with the system.)
  - Activity diagram: Process modeling (adding items to a shopping cart, checking out, and processing payments.)

- In UML:
  - Class diagram: Classes in the system (user, Item, Order, and Payment.)
  - State diagram: Model the state changes (for an order: being placed, shipped, and delivered.)

When is it realistic to use SysML

- System development process requires a model-based approach.
- While designing a system to create a communication among different stakeholders (system engineers, software developers, hardware engineers,…).
- High level of traceability of relationships among requirements.
- While designing complex systems with multiple components.
- Various types of system simulations and analyses needed.

SysML Usage Consequences

Advantages:

- Helps in designing complex system
- Provides a common language and framework for communication among stakeholders
- Offers a standardized notation for system modeling
- Allows for simulation and analysis of system behavior
- Supports model-based systems engineering practices

Disadvantages:

- Steep learning curve for new users
- Requires a significant investment in time and resources to implement effectively
- May not be suitable for small or simple systems
- Lack of standardization across different tools and implementations can lead to interoperability issues
- May not be well-suited for certain types of systems or domains.

Summary

- The recent advances in UML and SysML allow for efficient communication between project stakeholders.
- Neither UML nor SysML solve the difficulty associated with systems analysis.

Sources

- SysML, https://www.omgsysml.org/
- No Magic, https://docs.nomagic.com/display/SYSMLP190/Getting+started
- Visual Paradigm, https://www.visual-paradigm.com/guide/
- Altova, https://www.altova.com/manual/UModel/umodelprofessional/umsysml.html
- Innoslate, https://help.innoslate.com/users-guide/diagrams-view/types/sysml/
- Enterprise Architect, https://sparxsystems.com/products/ea/index.html
- Papyrus, https://projects.eclipse.org/projects/modeling.mdt.papyrus/governance

Sources

- [1] Technical Board INCOSE. Systems Engineering Handbook, V. 2a, June 2004.
- [2] https://engineering.cofc.edu/systems/index.php
- [3] https://www.nasa.gov/pdf/311199main_Art_and_Sci_of_SE_SHORT_1_20_09.pdf
- [4] Systems Engineering mit SysML/UML: Modellierung, Analyse, Design / TimWeilkins
- [5] https://slideplayer.com/slide/16196759/
- [6] www.incose.org/ProductsPubs/products/sevision2020.aspx/ February 2009.
- [7] https://www.researchgate.net/figure/UML-2-timing-diagram-Further-example-analysis-reveals-that-only-the-alternative-in-which_fig4_337679680
- [8] http://www.omgsysml.org/Building_SE-SW_Bridges%20.pdf
- [9] https://www.omgsysml.org/SysML_SE_applied_to_SoC.pdf

Thank you!