Helena Kruus
Department of Computer Engineering

helena.kruus@ati.ttu.ee

SysML – Systems Modeling Language

Outline

- UML Overview
- UML Diagrams
- Drawbacks of UML
- SysML Introduction
- SysML Diagrams
- Online resources
- Software for SysML

Slides and additional materials

Available at:
http://www.ati.ttu.ee/~helena_k/sysml

UML overview

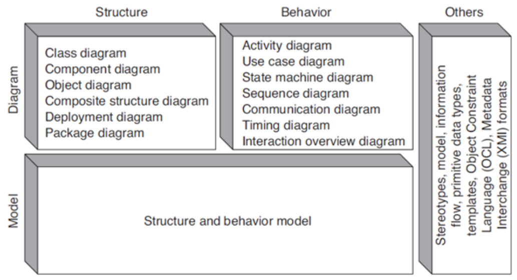
"The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system.

The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components."

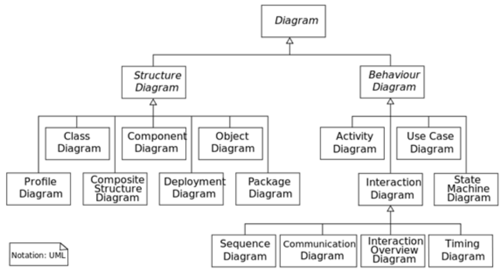
UML overview

- Was developed by Grady Booch, Ivar Jacobson and James Rumbaugh at Rational Software in 1990 s
- Adopted by Object Management Group (OMG) in 1997
- Accepted by International Organization for Standardization (ISO) in 2000
- The current vesrion of UML is 2.4.1 – published in 2011

UML diagrams

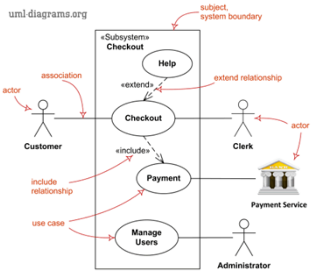

Structure
Behavior
Others
Diagram
Class diagram
Component diagram
Object diagram
Composite structure diagram
Deployment diagram
Package diagram
Activity diagram
Use case diagram
State machine diagram
Sequence diagram
Communication diagram
Timing diagram
Interaction overview diagram
Model
Structure and behavior model
Stereotypes, model, information flow, primitive data types, templates, Object Constraint Language (OCL), Metadata Interchange (XMI) formats

UML diagrams



UML Use Case

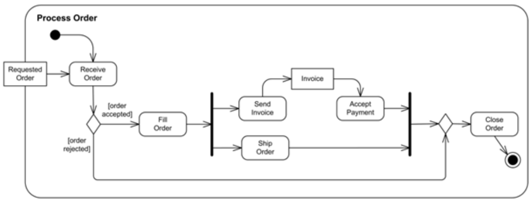
Used to describe a set of actions (use cases) that some system or or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).



Ref: http://www.uml-diagrams.org/

UML Activity Diagram

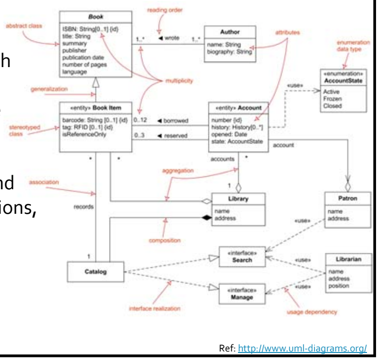
Activity diagram is UML behavior diagram which shows flow of control or object flow with emphasis on the sequence and conditions of the flow.



Ref: http://www.uml-diagrams.org/

UML Class Diagram

Class diagram is UML structure diagram which shows structure of the designed system at the level of classes and interfaces, shows their features, constraints and relationships - associations, generalizations, dependencies, etc.

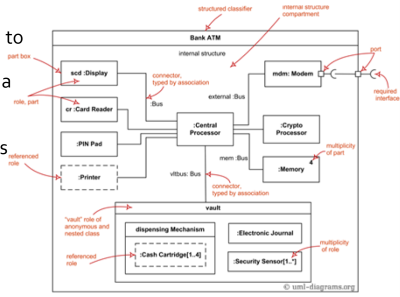


Ref: http://www.uml-diagrams.org/

UML Composite Structure Diagram

Composite Structure Diagram could be used to show:

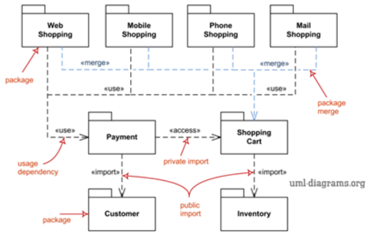
- internal structure of a classifier - internal structure diagram
- classifier interactions with environment through ports
- a behavior of a collaboration - collaboration use diagram.



Ref: http://www.uml-diagrams.org/

UML Package Diagram

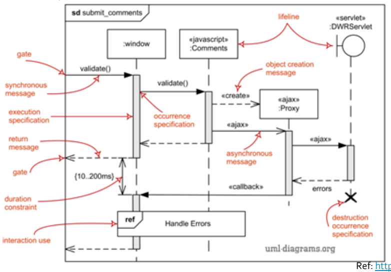
Package diagram is UML structure diagram which shows structure of the designed system at the level of packages.



Ref: http://www.uml-diagrams.org/

UML Sequence Diagram

Sequence diagram describes an interaction by focusing on the sequence of messages that are exchanged, along with their corresponding occurrence specifications on the lifelines.



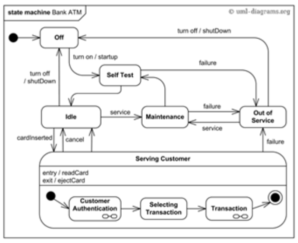
Ref: http://www.uml-diagrams.org/

UML State Machine Diagram

Desribes descrete behavior of a part of a system.

State represents a stuation when part of a system:

- satisfies a particular condition
- performs a particular activity
- waits for particular event



Drawbacks of UML for SE

- UML is standardized modeling language in the field of object-oriented **software engineering**.
- The strong link between the software and hardware is not expressed in UML
- The lack of unified representation of software-hardware functions makes it difficult to check the whole system
- The lack of visibility in terms of functional and structural specifications

Ref: N. Ouerdi, H. M. Ziane, M. Azizi and A. Azizi
„Modeling Embedded Systems with SysML
" *International Conference on Multimedia Computing and Systems (ICMCS)*
Oujda, Morocco, 2012

What is SysML?

*The OMG systems Modeling Language **(OMG SysML™)** is a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that may include hardware, software, information, personnel, procedures, and facilities. In particular, the language provides graphical representations with a semantic foundation for modeling system requirements, behavior, structure, and parametrics, which is used to integrate with other engineering analysis models. (http://www.omgsysml.org/)*

SysML - Introduction

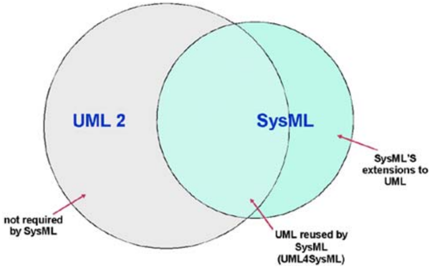
- **SysML** is a graphical modeling language derived from UML for Systems Engineering
- Developed by **Object Management Group** (OMG) and **International Council on Systems** Engineering (INCOSE)
- The OMG announced the adoption of SysML in January 2006
- The first version (SysML v1.0) became available in September 2007
- The current version (SysML v1.3) was published in June 2012

SysML – Introduction (2)

- OMG SysML page: http://www.omgsysml.org/
- INCOSE: http://www.incose.org/
- INCOSE SysML tutorial: http://www.omgsysml.org/SysML-Tutorial-Baseline-to-INCOSE-060524-low_res.pdf

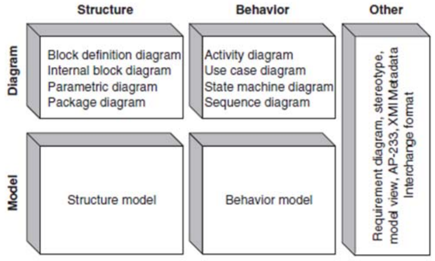
Relationship between SysML and UML

SysML includes extensions that are needed to satisfy the needs of Systems Engineering community



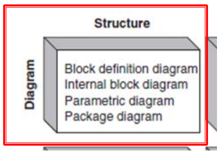
SysML structure

The structure of SysML



Ref: Tim Weilkiens.
„Systems Engineering with SysML/UML: Modeling, Analysis, Design"

SysML structure (2)



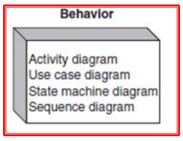
Structural diagrams:

- Unique to SysML
- Show the static structure of the system
- Enable to model both logical and physical structural elements
- Show the connections between them

SysML structure (2)

Behavioral diagrams:

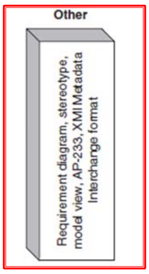
- The dynamic view of the system
- Show the behaviour of the system
- Dynamics of activities
- States and modes of the system
- Conditions that cause modes to change



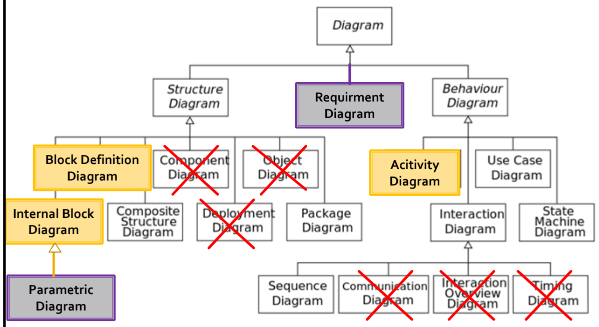
SysML structure (2)

Cross cutting diagrams:

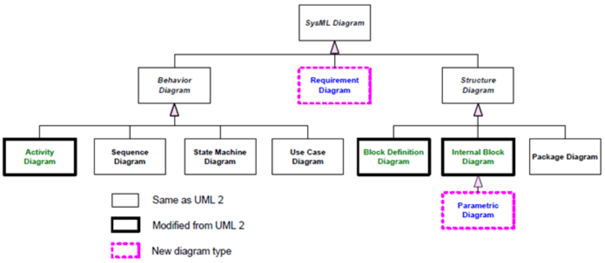
- Requirements diagram
- Allocation tables
- Package diagrams
- Allow to model relationships across stuctural and behavioral elements


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

SysML – UML 2.2 to SysML



SysML Taxonomy



SysML vs UML (1)

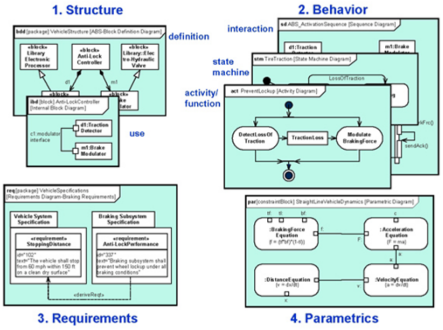
SYSML DIAGRAM	PURPOSE	UML ANALOG
Activity diagram	Show system behavior as control and data flows. Useful for functional analysis. Compare Extended Functional Flow Block diagrams (EFFBDs), already commonly used among systems engineers.	**Activity diagram**
Block Definition diagram	Show system structure as components along with their properties, operations and relationships. Useful for system analysis and design.	**Class diagram**
Internal Block diagram	Show the internal structures of components, including their parts and connectors. Useful for system analysis and design.	**Composite Structure diagram**
Package diagram	Show how a model is organized into packages, views and viewpoints. Useful for model management.	**Package diagram**
Parametric diagram	Show parametric constraints between structural elements. Useful for performance and quantitative analysis.	N/A
Requirement diagram	Show system requirements and their relationships with other elements. Useful for requirements engineering.	N/A

SysML vs UML (2)

Sequence diagram	Show system behavior as interactions between system components. Useful for system analysis and design.	**Sequence diagram**
State Machine diagram	Show system behavior as sequences of states that a component or interaction experience in response to events. Useful for system design and simulation/code generation.	**State Machine diagram**
Use Case diagram	Show system functional requirements as transactions that are meaningful to system users. Useful for specifying functional requirements. (Note potential overlap with Requirement diagrams.)	**Use Case diagram**
Allocation tables* *dynamically derived tables, not really a diagram type	Show various kinds of allocations (e.g., requirement allocation, functional allocation, structural allocation). Useful for facilitating automated verification and validation (V&V) and gap analysis.	N/A

Eric A. Barnhart
„Why Use SysML" ?

Four pillars of SysML



Note that the Package and Use Case diagrams are not shown in this example, but are respectively part of the structure and behavior pillars

SysML diagram frame

The syntax for a full diagram header:

type [model element type] model element [diagram name]

For example:

req [package] HSUVRequirements [Requirement Derivation]

*The diagram frame is a mandatory description in SysML. The title has to include the **type** and **name of the model element**, while the model element type and diagram name are optional.*

SysML diagram frame (2)

req [package] HSUVRequirements [Requirement Derivation]

- **type**: indicates the type of diagram
- [model element type]: indicates the type of model element that the diagram represents
- model element: indicates the Name of the represented model element
- [diagram name]: indicates the Name of the diagram; used to provide a description of the diagram purpose
- description can be added as a note

Diagram abbreviations

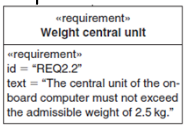
- **`act`** - Activity diagram
- **`req`** - Requirement diagram
- **`uc`** - Use case diagram
- **`bdd`** - Block definition diagram
- **`ibd`** - Internal block diagram
- **`pkg`** - Package diagram
- **`par`** - Parametric diagram
- **`sd`** - Sequence diagram
- **`stm`** - State machine diagram

Requirements

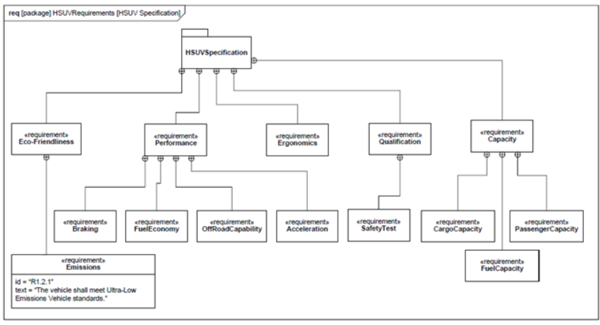
- A **requirement** describes one or more properties or behaviors of a system that always have to be met.
- **Functional requirements** represent capabilities of the system (*can be modeled with Use Case*)
- **Non-functional requirements** cover areas such as performance or reliability, constraints (*no element in UML to explicitly describe non-functional requirements*)

Requirement element

- Two elementary properties of a requirement:
  - a unique identifier (ID)
  - a descriptive text.
- Other properties
  - priority level (high, medium, low)
  - source
  - risk
  - status

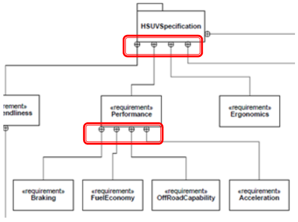

«requirement»
Weight central unit
«requirement»
id = "REQ2.2"
text = "The central unit of the on-board computer must not exceed the admissible weight of 2.5 kg."

Requirments hierarchy


req [package] HSUVRequirements [HSUV Specification]
HSUVSpecification
«requirement» Eco-Friendliness
«requirement» Performance
«requirement» Ergonomics
«requirement» Qualification
«requirement» Capacity
«requirement» Braking
«requirement» FuelEconomy
«requirement» OffRoadCapability
«requirement» Acceleration
«requirement» SafetyTest
«requirement» CargoCapacity
«requirement» PassengerCapacity
«requirement» Emissions
id = "R1.2.1"
text = "The vehicle shall meet Ultra-Low Emissions Vehicle standards."
«requirement» FuelCapacity

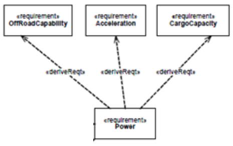
Requirments relationships (1)

- Containment – requirement is contained in another requirement
- Used for decomposition of requirements


HSUVSpecification
«requirement» Performance
«requirement» Ergonomics
rement» endliness
«requirement» Braking
«requirement» FuelEconomy
«requirement» OffRoadCapability
«requirement» Acceleration

Requirments relationships (2)

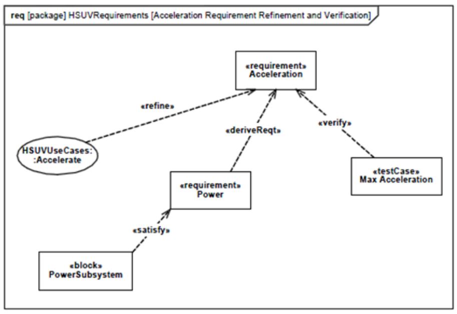
- A **derive requirement** relationship describes that a requirement was derived from another requirement. Notation: *«deriveReqt»*
- Provides more detail in the derived requirement


«requirement» OffRoadCapability
«requirement» Acceleration
«requirement» CargoCapacity
«deriveReqt»
«deriveReqt»
«deriveReqt»
«requirement» Power

Requirments relationships (3)

- A **satisfy relationship** describes that a design element satisfies a requirement. Notation: *«satisfy»*
- A **verify relationship** connects a test case with the requirement that is verified by that test case. Notation: *«verify»*
- A **refine relationship** connects requirement to a behavioral element; adds detail to the requirement

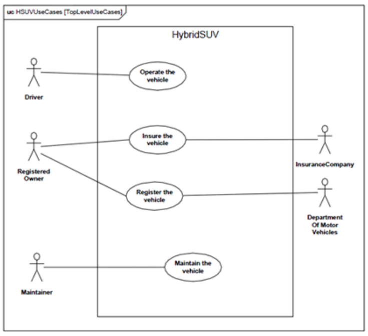
Requirement relationships (4)


req [package] HSUVRequirements [Acceleration Requirement Refinement and Verification]
«requirement» Acceleration
«refine»
«deriveReqt»
«verify»
HSUVUseCases: :Accelerate
«requirement» Power
«testCase» Max Acceleration
«satisfy»
«block» PowerSubsystem

SysML Use Case

- Integrated into SysML unchanged
- Describes the functionality of a system – what can a system do
- Presents a collection of sequences of actions performed by the system, which produces an observable result for actors

SysML Use Case (2)



SysML Use case (3)

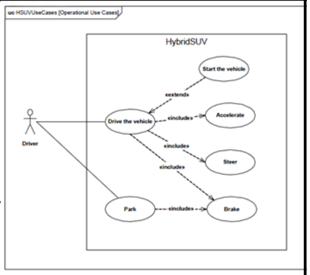
- **Actors** - interact with the system
- Can be roles for people, other systems or subsystems
- What does the actor need from the system?

- **Use Case** - return results to actors
- **Verbs (+noun)** are used to desribe use cases to express action

SysML Use Case (4)

Use Case relations:

- generalization – child use cases that add specialized behaviour
- <<include>> one use case includes the other
- <<extetnd>> provides optional extention of another



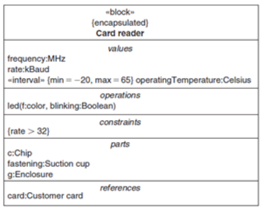
Block Definition Diagram

- Basic structural element used to model the structure of systems
- Replaces UML **Class Diagram**
- Used to represent blocks, their properties and inter-relations
- Block is represented graphically by a rectangle subdivided into compartmets
- Used to **define** blocks

Block Definition Diagram

Block characteristics:

- Name
- Values
- Operations
- Constraints
- Etc.



«block»
{encapsulated}
Card reader

values
frequency:MHz
rate:kBaud
«interval» {min = −20, max = 65} operatingTemperature:Celsius

operations
led(f:color, blinking:Boolean)

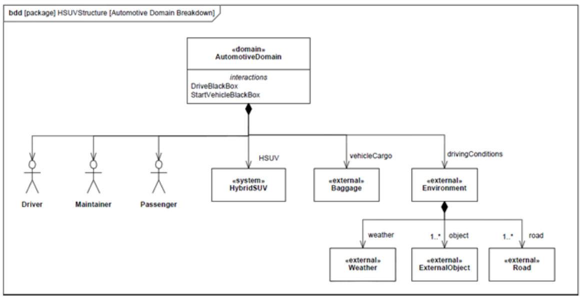
constraints
{rate > 32}

parts
c:Chip
fastening:Suction cup
g:Enclosure

references
card:Customer card

Ref: Tim Weilkiens.
„Systems Engineering with SysML/UML: Modeling, Analysis, Design"

Block Definition Diagram



bdd [package] HSUVStructure [Automotive Domain Breakdown]

«domain»
AutomotiveDomain

interactions
DriveBlackBox
StartVehicleBlackBox

Driver
Maintainer
Passenger

HSUV
«system»
HybridSUV

vehicleCargo
«external»
Baggage

drivingConditions
«external»
Environment

weather
«external»
Weather

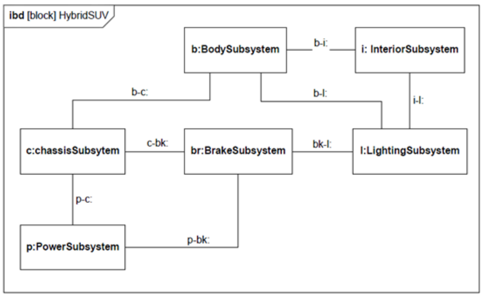
1..* object
«external»
ExternalObject

1..* road
«external»
Road

Internal Block diagrams

- Descibes the internal structure of a block in terms of parts, ports and connections
- Several levels of decomposition can be presented
- Used to **depict usage** of blocks in a context
- Shows the connectons of ports
- Shows the flows between parts and ports

Internal Block diagram



ibd [block] HybridSUV

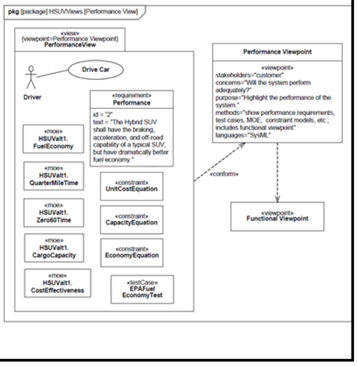
b:BodySubsystem
b-i:
i: InteriorSubsystem
b-c:
b-l:
i-l:
c:chassisSubsytem
c-bk:
br:BrakeSubsystem
bk-l:
l:LightingSubsystem
p-c:
p:PowerSubsystem
p-bk:

Package Diagram

Shows the organization of the model into **packages**

Concept of view and viewpoint reflect perspectives of different stakeholders.

View presents only a certain selection of model elements and diagrams, realizes exactly the viewpoint that describes the stakeholders and their interests.



pkg [package] HSUVViews [Performance View]

«view»
{viewpoint=Performance Viewpoint}
PerformanceView

Driver
Drive Car

«requirement»
Performance
id = "2"
text = "The Hybrid SUV shall have the braking, acceleration, and off-road capability of a typical SUV, but have dramatically better fuel economy."

«moe»
HSUValt1.
FuelEconomy

«moe»
HSUValt1.
QuarterMileTime

«moe»
HSUValt1.
Zero60Time

«moe»
HSUValt1.
CargoCapacity

«moe»
HSUValt1.
CostEffectiveness

«constraint»
UnitCostEquation

«constraint»
CapacityEquation

«constraint»
EconomyEquation

«testCase»
EPAFuel
EconomyTest

«conform»

Performance Viewpoint
«viewpoint»
stakeholders="customer"
concerns="Will the system perform adequately?"
purpose="Highlight the performance of the system."
methods="show performance requirements, test cases, MOE, constraint models, etc.; includes functional viewpoint"
languages="SysML"

«viewpoint»
Functional Viewpoint

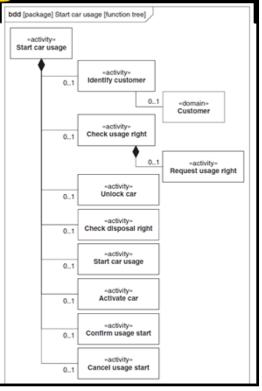
SysML Activity Diagram

- Specifies the flow as well as input and output of data.
- Flows can run in parallel, or they can be synchronized, or split based on conditions.
- SysML extends several properties of UML 2 activity model
  - Extended control flow with additional information to stop actions or control the flow via so-called control operators.
  - Support for modeling of continuous systems.
    - Continuous or discrete object flow.
    - Object nodes can reject data that are no longer current.
  - Probabilities of flows.
  - Modeling rules for activities in the form of a block definition diagram (function trees).

Ref: Tim Weilkiens.
„Systems Engineering with SysML/UML: Modeling, Analysis, Design"

SysML Activity Diagram

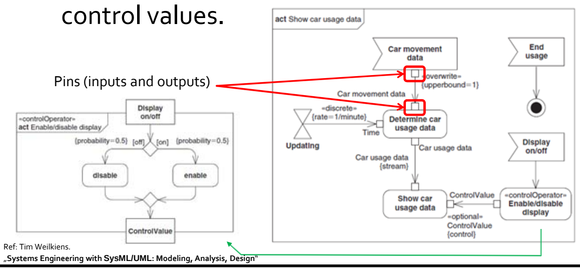
- Function trees can be represented by block definition diagrams


bdd [package] Start car usage [function tree]
«activity» Start car usage
0..1
«activity» Identify customer
0..1
«domain» Customer
0..1
«activity» Check usage right
0..1
«activity» Request usage right
0..1
«activity» Unlock car
0..1
«activity» Check disposal right
0..1
«activity» Start car usage
0..1
«activity» Activate car
0..1
«activity» Confirm usage start
0..1
«activity» Cancel usage start

Ref: Tim Weilkiens.
„Systems Engineering with SysML/UML: Modeling, Analysis, Design"

SysML Activity Diagram

- A **control operator** specifies a behavior that can enable and disable actions through control values.


Pins (inputs and outputs)

Ref: Tim Weilkiens.
„Systems Engineering with SysML/UML: Modeling, Analysis, Design"

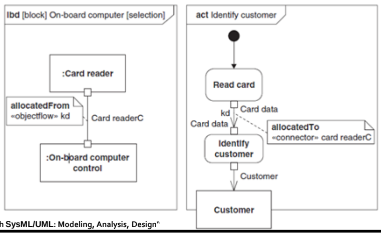
Allocations

- A mechanism that enables user to connect elemets to each other
- Three types of allocations:
  - Behavior – links a behavior to the block that realizes the behavior
  - Structure – links logical structures with physical structures (and vice versa)
  - object flow – connects an item flow (found in structure diagram) with an object flow edge (found in the activity diagram)
- Shown vy a dashed arrow in many of the SysML diagrams

Ref: A.Kossiakoff, W.N Sweet, S.J.Seymour, S.M. Biemer.
„Systems Engineering: Principles and practice"

Allocations (2)

- binding an item flow in the internal block diagram, which occurs in a different form, as an object flow, in an activity diagram


ibd [block] On-board computer [selection]
:Card reader
allocatedFrom «objectflow» kd
Card readerC
:On-board computer control
act Identify customer
Read card
Card data
kd
Card data
allocatedTo «connector» card readerC
Identify customer
Customer
Customer

Ref: Tim Weilkiens.
„Systems Engineering with SysML/UML: Modeling, Analysis, Design"

Allocations (3)

- Allocation tables capture record the allocations

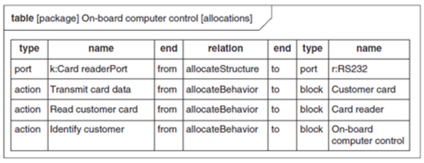


table [package] On-board computer control [allocations]

type	name	end	relation	end	type	name
port	k:Card readerPort	from	allocateStructure	to	port	r:RS232
action	Transmit card data	from	allocateBehavior	to	block	Customer card
action	Read customer card	from	allocateBehavior	to	block	Card reader
action	Identify customer	from	allocateBehavior	to	block	On-board computer control

Ref: Tim Weilkiens.
„Systems Engineering with SysML/UML: Modeling, Analysis, Design"

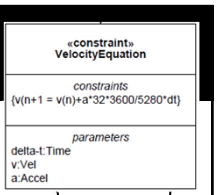
Parametric diagram

- SysML provides possibility to simulate portions of the model, based on mathematical and physical laws that describe key aspects of the system.
- Allows to represent constraints on system parameter values – for example, performance, reliability and mass

Constraint blocks

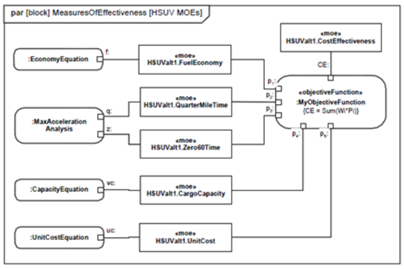
- A **constraint block** describes constraints on system structures and the parameters required. Notation: *«constraintBlock»* (or *«constraint»*)
- Constraints are declared in Block Definition diagram
- Decraled constrains are applied to the parametric diagram

Constraint blocks (2)



- Support the construction of parametric models
- Define equations so that they may be re-used and inter-connected
- Define a set of parameters
  - Contained in the 'parameters' compartment
- Define an expression that constrains the parameters
  - Contained in the 'constraints' compartment
- Depicted with the keyword <<constraint>>

Parametric diagram


par [block] MeasuresOfEffectiveness [HSUV MOEs]
:EconomyEquation
f:
«moe» HSUVaIt1.FuelEconomy
:MaxAcceleration Analysis
q:
z:
«moe» HSUVaIt1.QuarterMileTime
«moe» HSUVaIt1.Zero60Time
:CapacityEquation
vc:
«moe» HSUVaIt1.CargoCapacity
:UnitCostEquation
uc:
«moe» HSUVaIt1.UnitCost
«moe» HSUVaIt1.CostEffectiveness
CE:
«objectiveFunction» :MyObjectiveFunction {CE = Sum(Wi*Pi)}
p_1:
p_2:
p_3:
p_4:
p_5:

Simulation example

- http://www.pld.ttu.ee/~helena_k/sysml/simulation/simulatsioon.swf

Summary

- SysML is an extention of UML for Systems Engineering
- Includes nine diagram kinds to desribe the model
- The semantics of a language enable a modeler to develop an integrated model of a system

Online resources

UML:

- UML Resource Page: http://www.uml.org/
- UML graphical notation overview: http://www.uml-diagrams.org/
- UML Tutorial: http://uml-tutorials.trireme.com/
- Google !!!

SysML:

- OMG SysML page: http://www.omgsysml.org/
- INCOSE SysML Tutorial: http://www.sysmlforum.com/sysml-tutorials/
- Think SysML: http://www.thinksysml.org/Tutorials.html
- SysML FAQ: http://www.sysmlforum.com/sysml-faq/
- HSUV sxample: http://www.omg.org/ocsmp/HSUV.pdf

Online resources (2)

SysML (cont.):

- YouTube: Introduction to Systems Modeling Language
  - http://www.youtube.com/watch?v=xGDErNmqNLw
  - http://www.youtube.com/watch?v=1tJq6DgYoBE
- Vimeo: Model-Based SE using SysML
  - https://vimeo.com/16618093
- (Enterprise Architect) Embedded Systems Development using SysML http://www.sparxsystems.com/downloads/ebooks/Embedded_Systems_Development_using_SysML.pdf
- Links to some additional resources: http://www.ati.ttu.ee/~helena_k/sysml
- Google !!!!

Modeling Task

- Will be specified by Prof. G.Jervan
- Model the subsystem of your project using SysML
- Give 0,5-1 A4 feedback on the chosen software: reasons for choosing, advanatges, disadvanatges etc

Software

- Commercial software providing free trial versions:
  - IBM Rational Rhapsody Architect for Systems Engineers (30 days)
  - Enterprise Architect (30 days)
  - Altova UModel (30 days)
  - Astah UML (20 days, can be extended)
  - Cameo Systems (MagicDraw + SysML plug-in)
  - …

Software

- Community software and free software
  - Visual Paradigm Community Edition
  - Modelio Free / OpenSource
  - TopCASED (with Eclipse)
  - …

References

- UML Diagrams: http://www.uml-diagrams.org/
- Tim Weilkiens. „**Systems Engineering with SysML/UML: Modeling, Analysis, Design**“ 2006
- F.Kordon, J.Hugues, A.Canals, A.Dohet „Embedded systems: **Analysis and Modeling with SysML, UML and AADL**“2013
- S.Friedenthal, A.Moode, R.Steiner. „A practical Guide to SysML“ 2012
- Eric A. Barnhart „Why Use SysML“ ?
- OMG SysML: http://www.omgsysml.org/