Kap. 7:
Entwicklung von Anwendungen
(b) Modellierung

7. Entwicklung von Anwendungen

Konzentration auf

- Sicherheitskritische Anwendungen
- Echtzeitanwendungen

Gliederung

(a)

1. Einführung
2. IEC EN 61508
3. Programmiersprachen (MISRA C/C++, Ada)

(b)

4. Modellierung (SysML, UML MARTE, …)

(c)

5. Validierung
6. Systematisches Testen

7.4 Modellierung

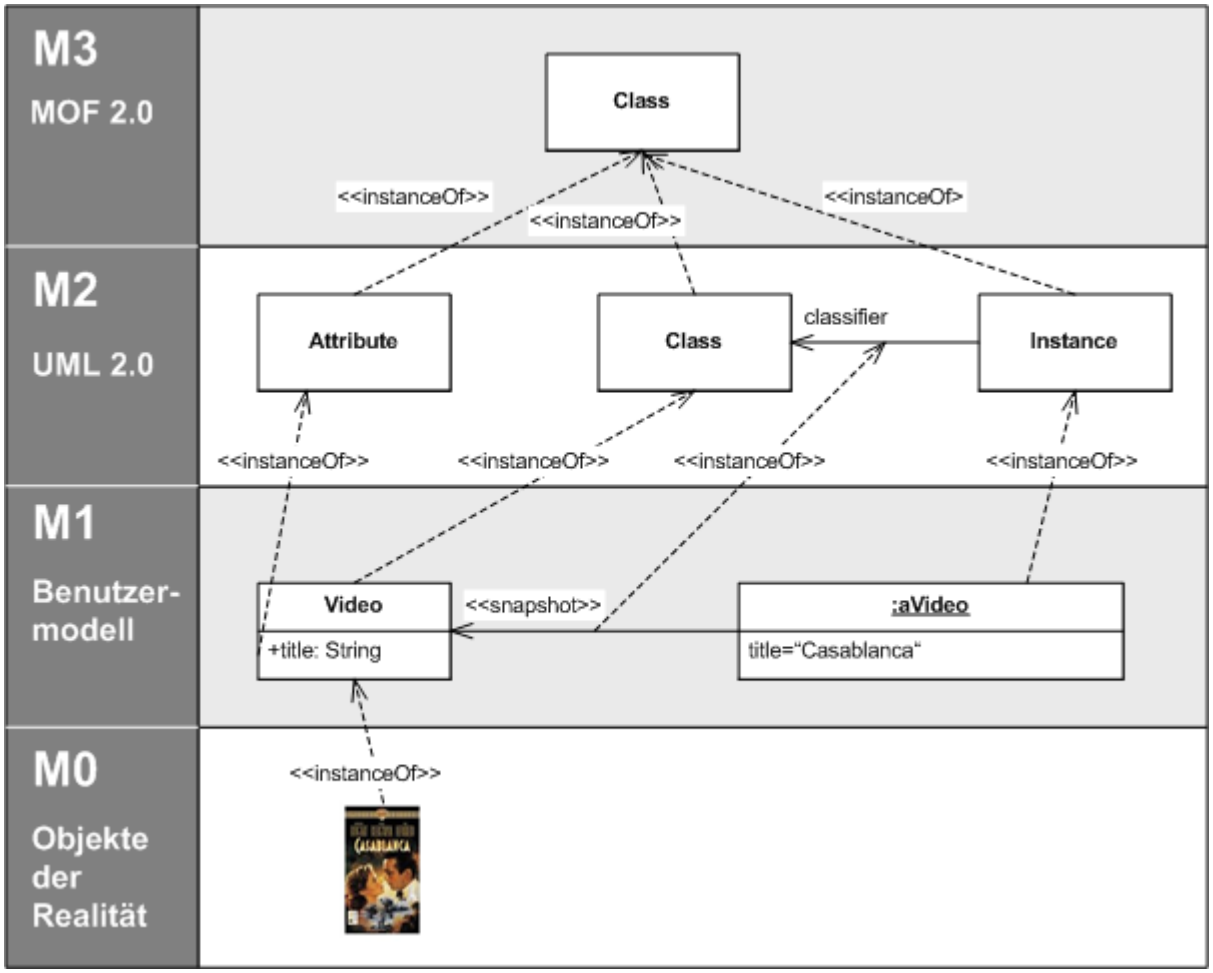
Gliederung

7.4.1 Einführung

- OMG Object Management Group
- UML und ihre Verwandten - Standards zu Modellierung
  - Bisher primär UML (Unified Modeling Language) kennengelernt (vgl. LV Softwaretechnik)
  - Katalog: http://www.omg.org/technology/documents/modeling_spec_catalog.htm#MOF
  - hier nur Ausschnitt
  - SysML (Systems Modeling Language, für komplexe Systeme)
  - MOF Meta Object Facility (extensible model driven integration framework, u.a. Modellierung von UML)
  - OCL Object Constraint Language (Festlegung von Prädikaten/Restriktionen für UML-Modelle und Metamodelle
  - XMI XML Metadata Interchange Specification
    OMG MOF 2 XMI Mapping Specification
    (MOF → XML Mapping, Exportieren/Importieren von Modellen aus/in Modellierungswerkzeugen)

MOF Meta Modeling

- **MOF: OMG Meta Object Facility**
- **M3 hat sich selbst als Metamodell**



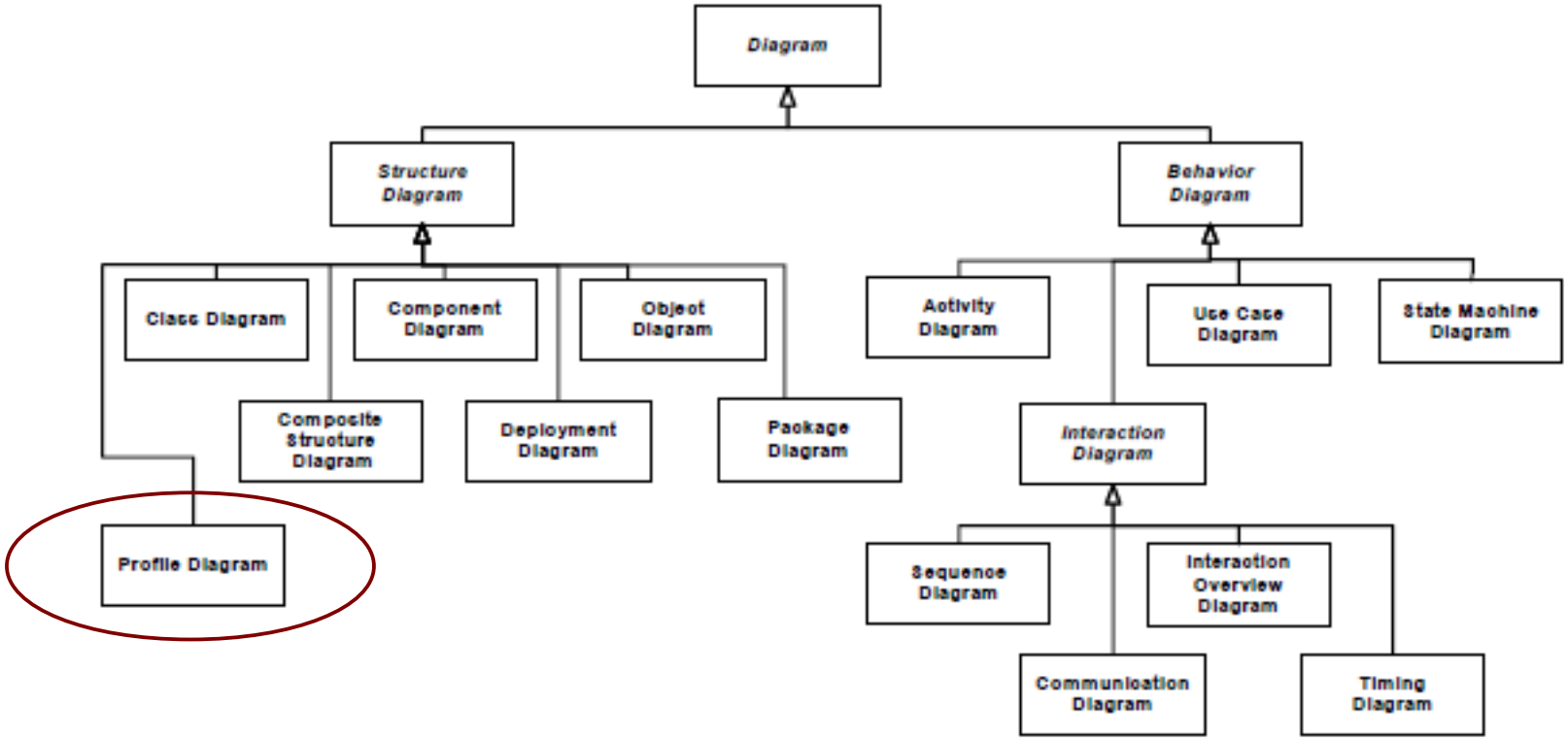
http://upload.wikimedia.org/wikipedia/commons/9/93/M0-m3.png

UML 2 Profil

- Erweiterung des UML 2 Metamodells durch einen leichtgewichtigen Erweiterungsmechanismus
- „A profile defines limited extensionsto a reference metamodel with the purpose of adapting the metamodel to a specific platform or domain“ (OMG 10-05-05)
- UML Metamodell bleibt unverändert
- Profile Diagram als neuer Diagrammtyp für die Anwendungvon Profilen
- Erweiterung basiert auf
  - Stereotypes mit Tagged Values
  - Constraints (Kommentare oder in formaler OMG Object Constraint Language (OCL))
- Analogie zu Einschränkungen auf Quellcode-Ebene wie z.B. in MISRA C

UML Diagrammtypen

- **UML 2.3**



UML 2.3 OMG 10-05-05.pdf

UML Profil (Definition der OMG)

- **A UML profile is a specification that does one or more of the following:**
  - **Identifies a subset of the UML metamodel.**
  - **Specifies “well-formedness rules” beyond those specified by the identified subset of the UML metamodel.**
  - **“Well-formedness rule” is a term used in the normative UML metamodel specification to describe a set of constraints written in UML’s Object Constraint Language (OCL) that contributes to the definition of a metamodel element.**
  - **Specifies “standard elements” beyond those specified by the identified subset of the UML metamodel.**
  - **“Standard element” is a term used in the UML metamodel specification to describe a standard instance of a UML stereotype, tagged value or constraint.**
  - **Specifies semantics, expressed in natural language, beyond those specified by the identified subset of the UML metamodel.**
  - **Specifies common model elements, expressed in terms of the profile.**

Profile-Notationen

Figure 18.5 - The notation for an Extension

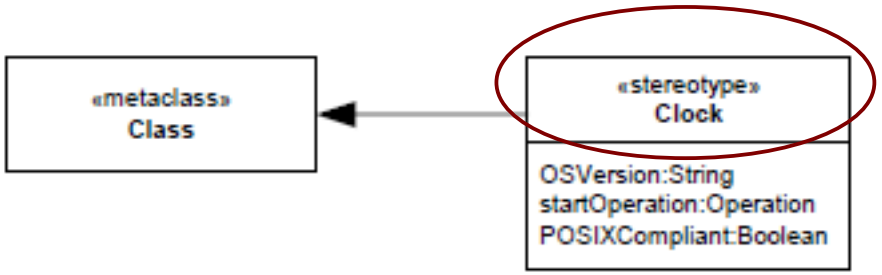


Figure 18.13 - Defining a stereotype

Figure 18.17 - Using a stereotype

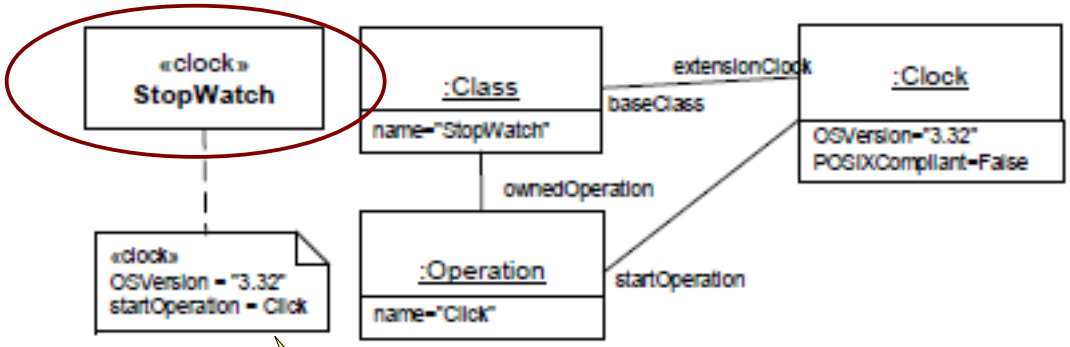
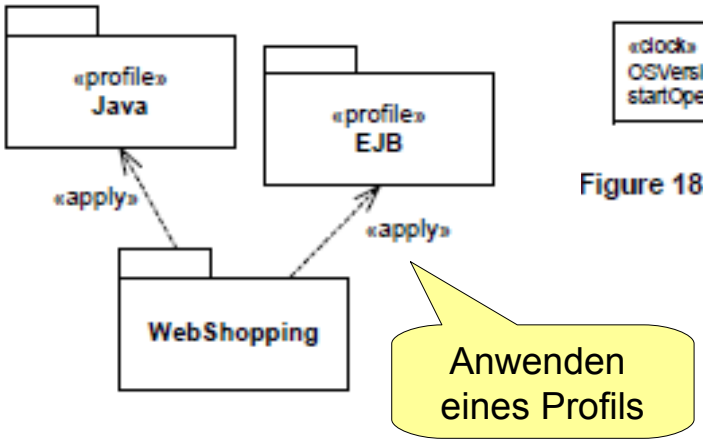


Figure 18.18 - Showing values of stereotypes and a simple instance specification



UML 2.3 OMG 10-05-05.pdf

Figure 18.12 - Profiles applied to a package

7.4.2 SysML

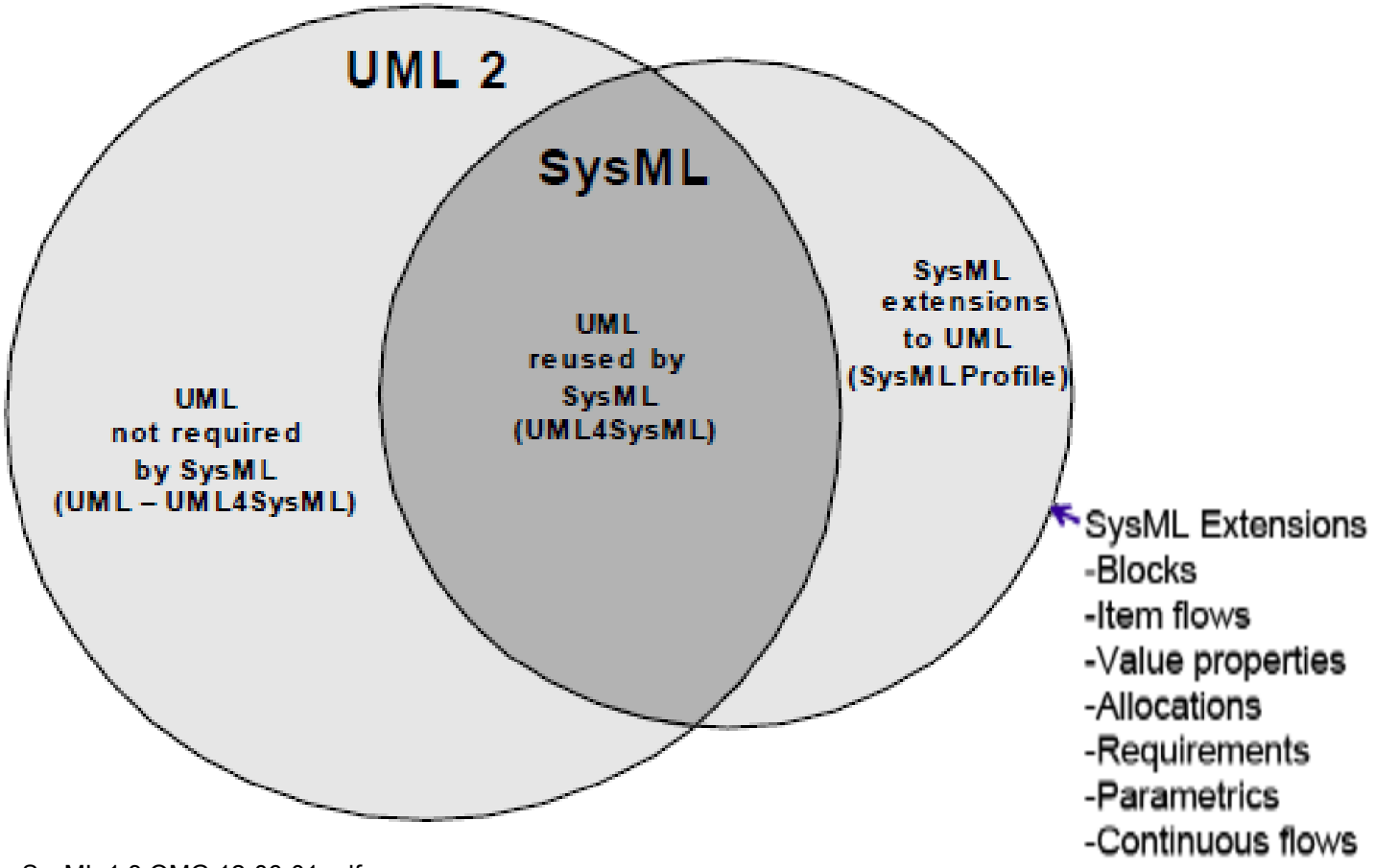
- OMG SysML: Systems Modeling Language
- UML 2 Profil für Systems Engineering
- OMG Standard formal/2012-06-01 (Version 1.3 beta)
- The OMG Systems Modeling Language (OMG SysML™) is a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that 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. SysML represents a subset of UML 2.0 with extensions needed to satisfy the requirements of the UML™ for Systems Engineering RFP. SysML uses the OMG XML Metadata Interchange (XMI®) to exchange modeling data between tools.
- Erweiterung durch UML Stereotypes, UML Diagramm-Erweiterungen, Modell-Bibliotheken
- Engineering-Prozess: Von Dokumenten zu Modellen

SysML Tools

- Basieren i.w. auf UMl-Modellierungswerkzeugen
- Kommerziell
  - Artisan (Studio)
  - EmbeddedPlus (SysML Toolkit)
  - No Magic (Magic Draw)
  - SparxSystems (Enterprise Architect)
  - IBM / Telelogic (Tau and Rhapsody)
- Open Source basierend auf Eclipse
  - TopCased
  - Papyrus

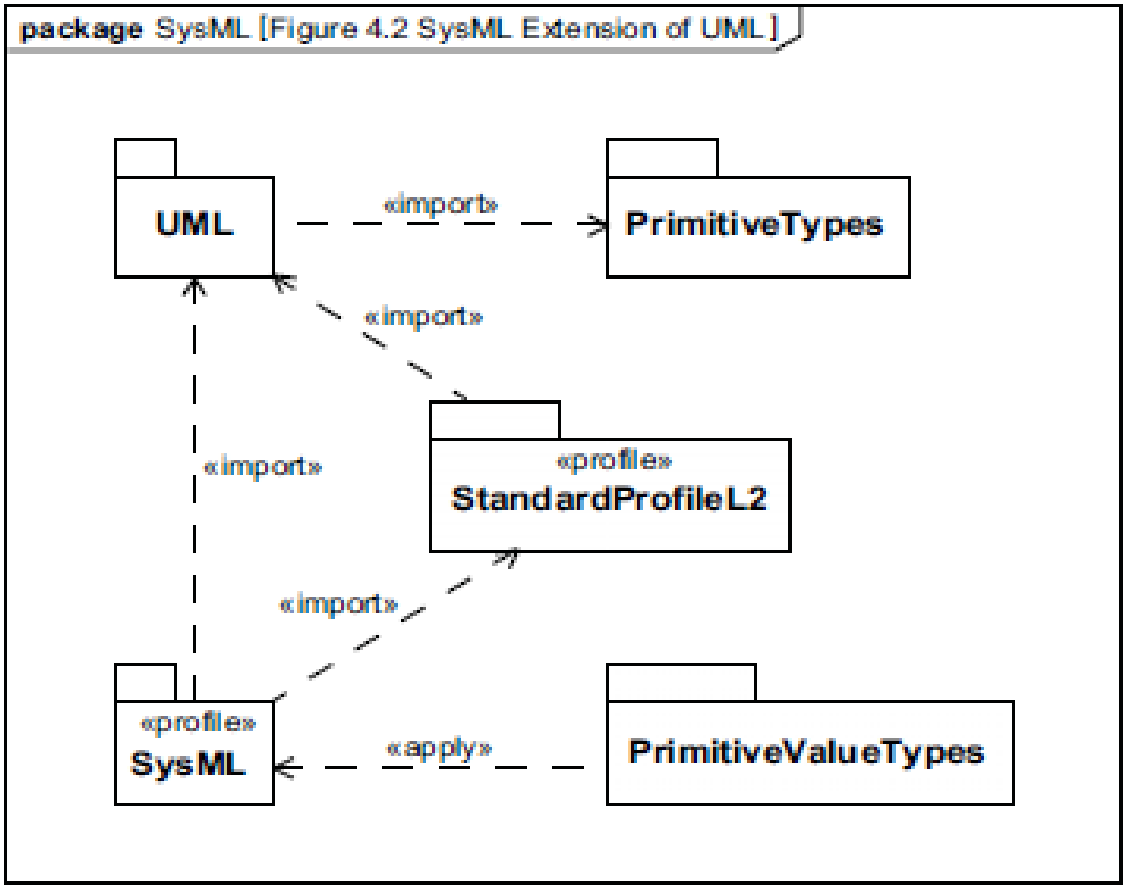
SysML ↔ UML 2

- **Überblick zum Zusammenhang**



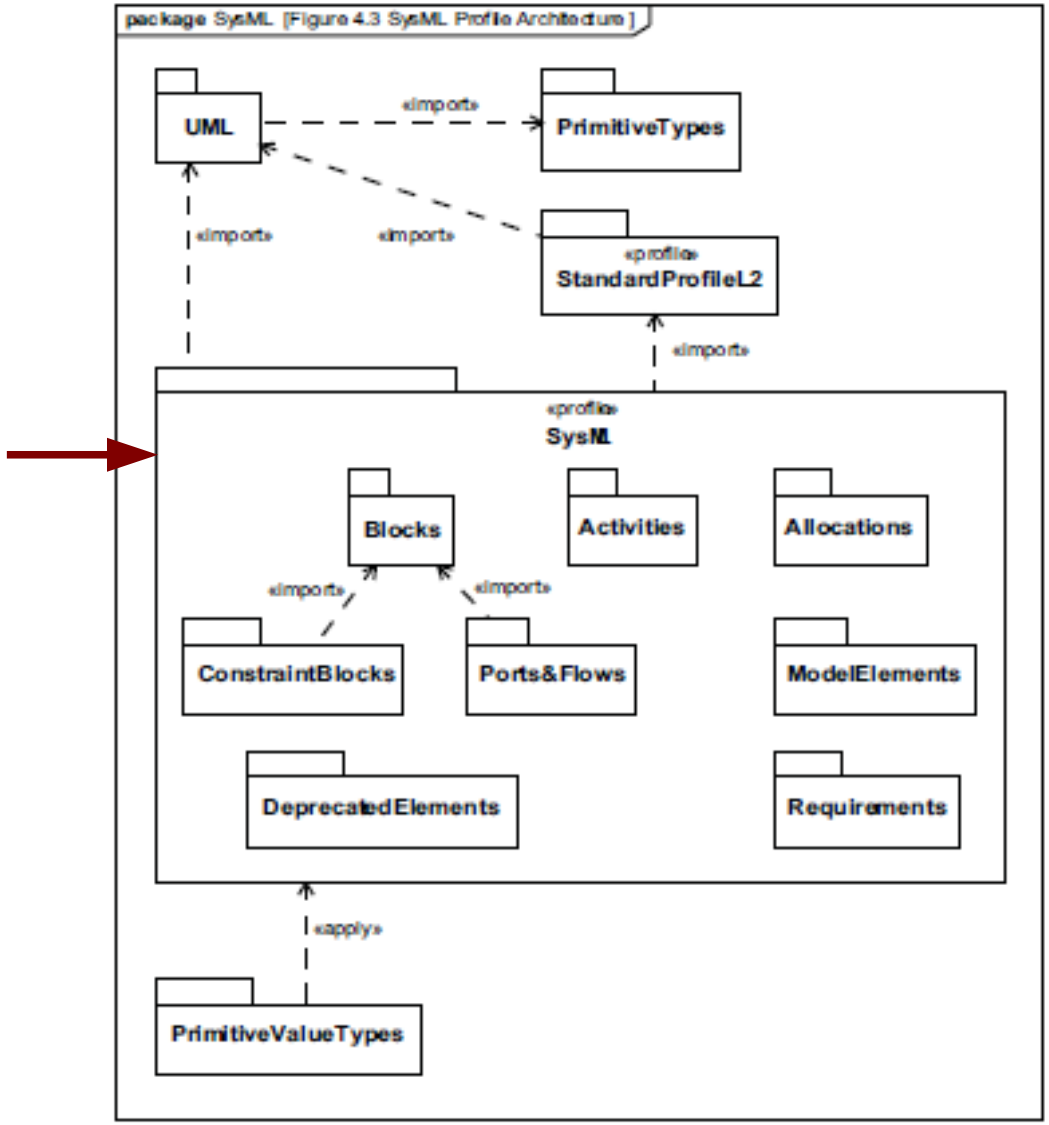
SysML 1.3 OMG 12-06-01.pdf

- **Erweiterungen**



SysML 1.3 OMG 12-06-01.pdf

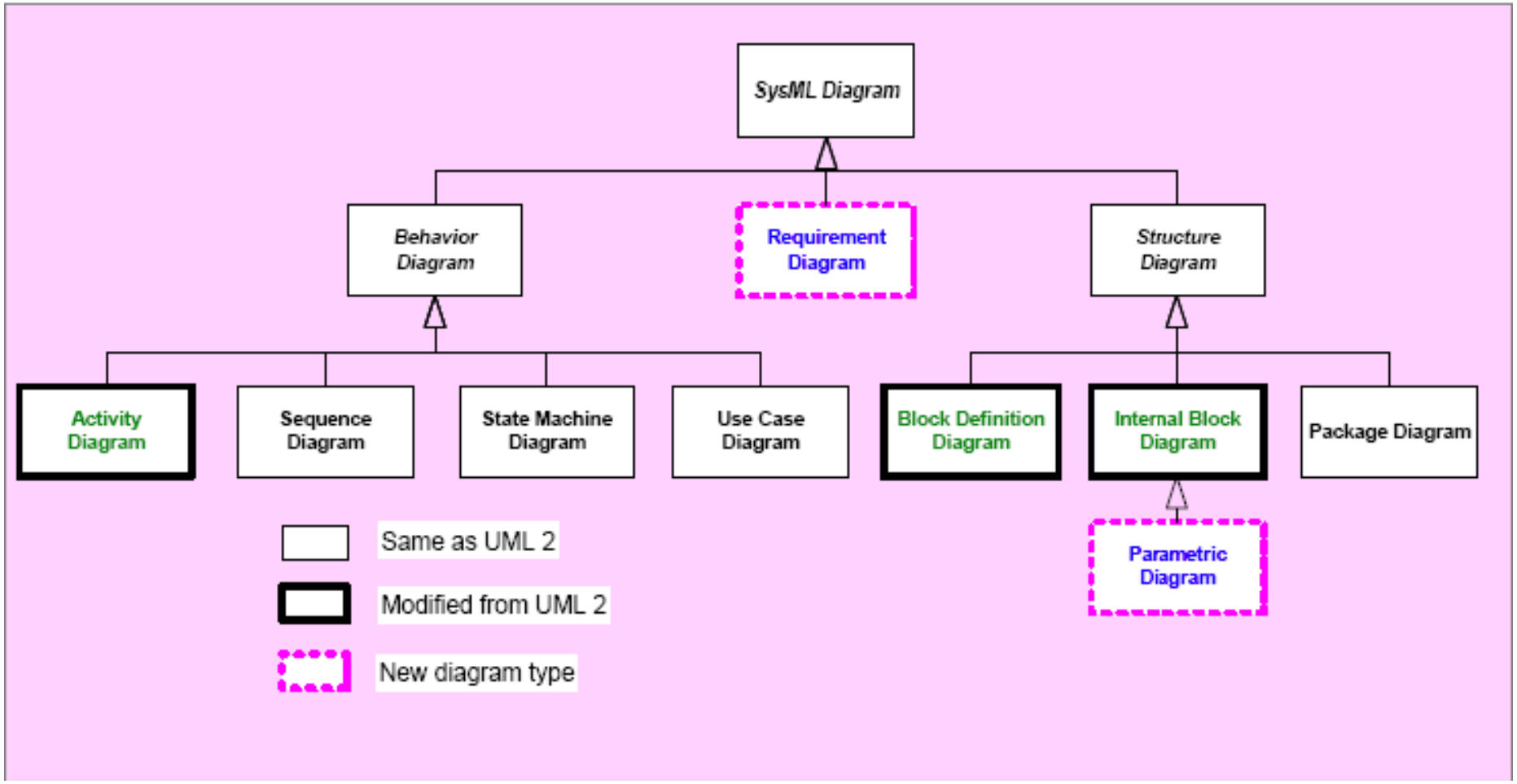
SysML Paketstruktur



SysML 1.3 OMG 12-06-01.pdf

SysML-Diagrammtypen

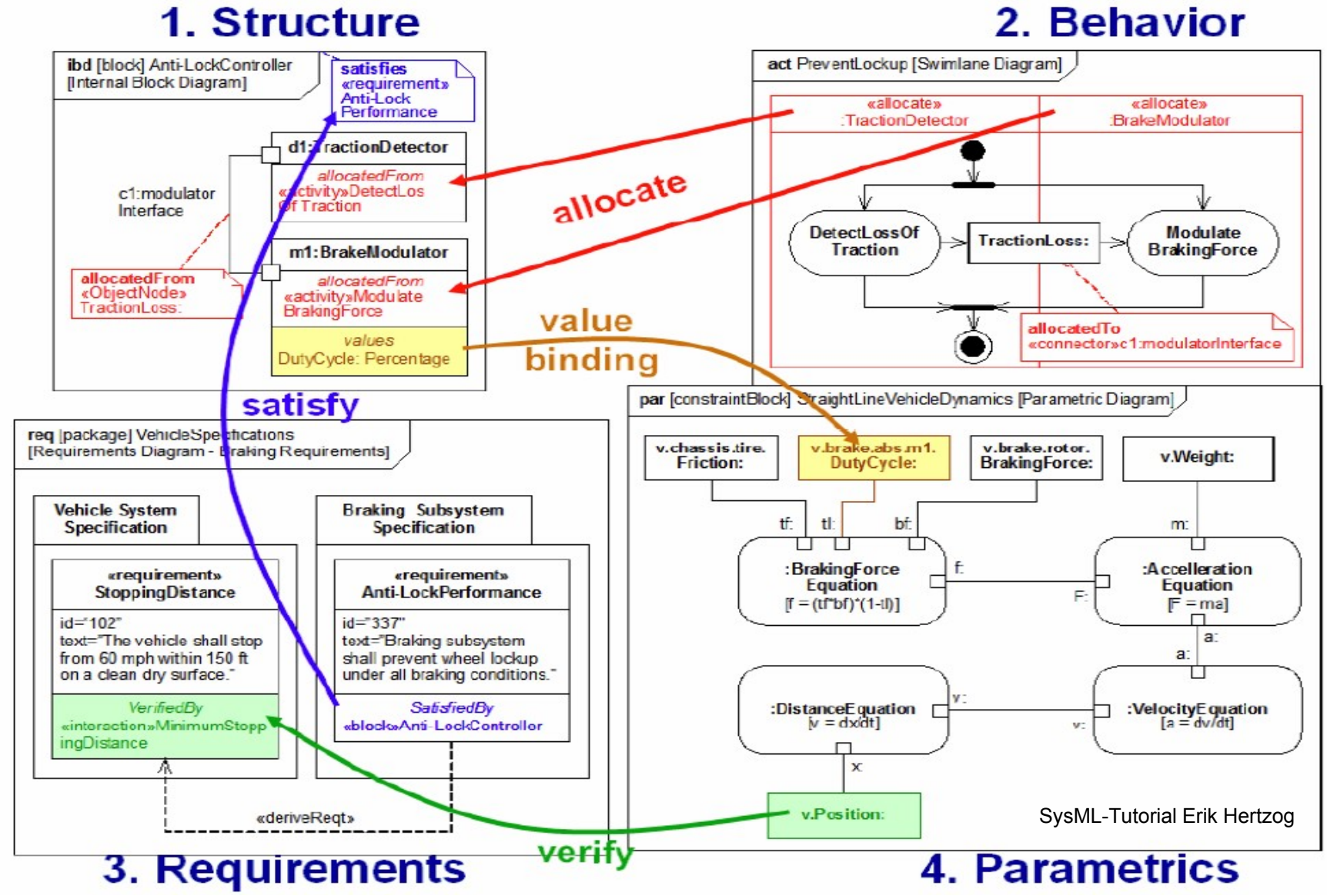
Überblick



SysML-Erweiterungen

- The SysML packages extend UML as follows:
  - SysML::Model Elements refactors and extends the UML kernel portion of UML classes.
  - SysML::Blocks reuses structured classes from composite structures.
  - SysML::ConstraintBlocks extends Blocks to support parametric modeling.
  - SysML::Ports and Flows extends UML ports, UML information flows, and SysML Blocks.
  - SysML::Activities extends UML activities.
  - SysML::Allocations extends UML dependencies.
  - SysML::Requirements extends UML classes and dependencies.
  - SysML::DeprecatedElements extends UML ports, UML interfaces, and SysML Item Flows.

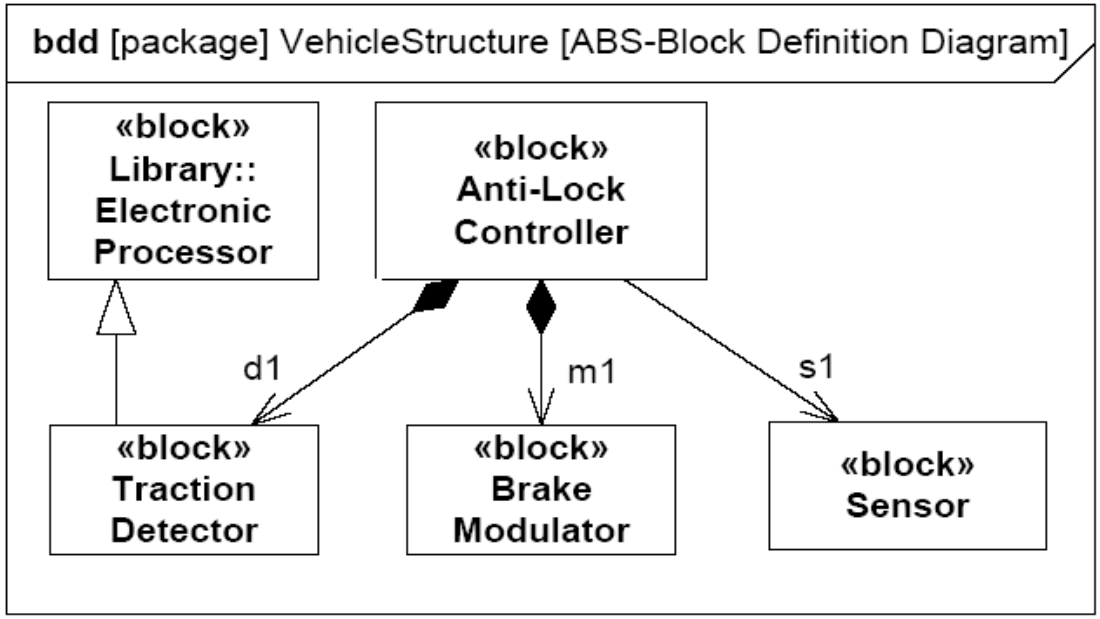
SysML-Säulen



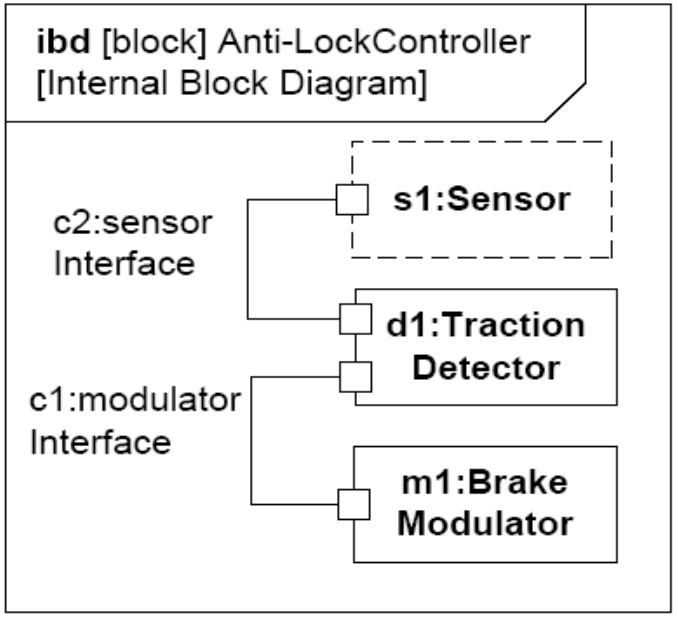
SysML-Tutorial Erik Hertzog

Blockdiagramme

Block Definition Diagram



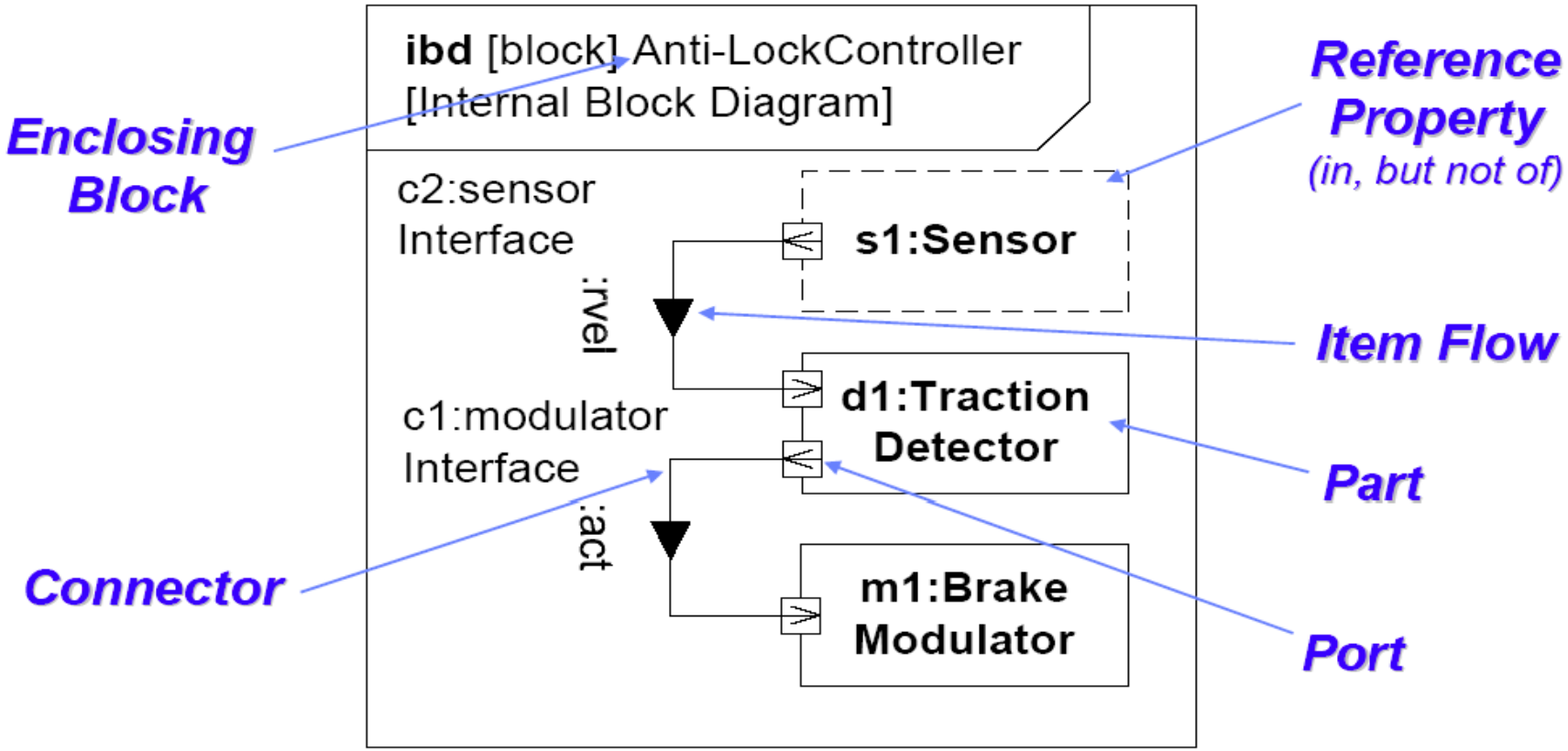
Internal Block Diagram



- **Definition von Bausteinen**
- **Festlegung von Eigenschaften**
- **Beziehungen zwischen Blöcken („part“ = Benutzung eines Blocks)**
- **Sichtbare Schnittstellen**

SysML-Tutorial Erik Hertzog

Blocks, Parts, Ports, Connectors, Flows

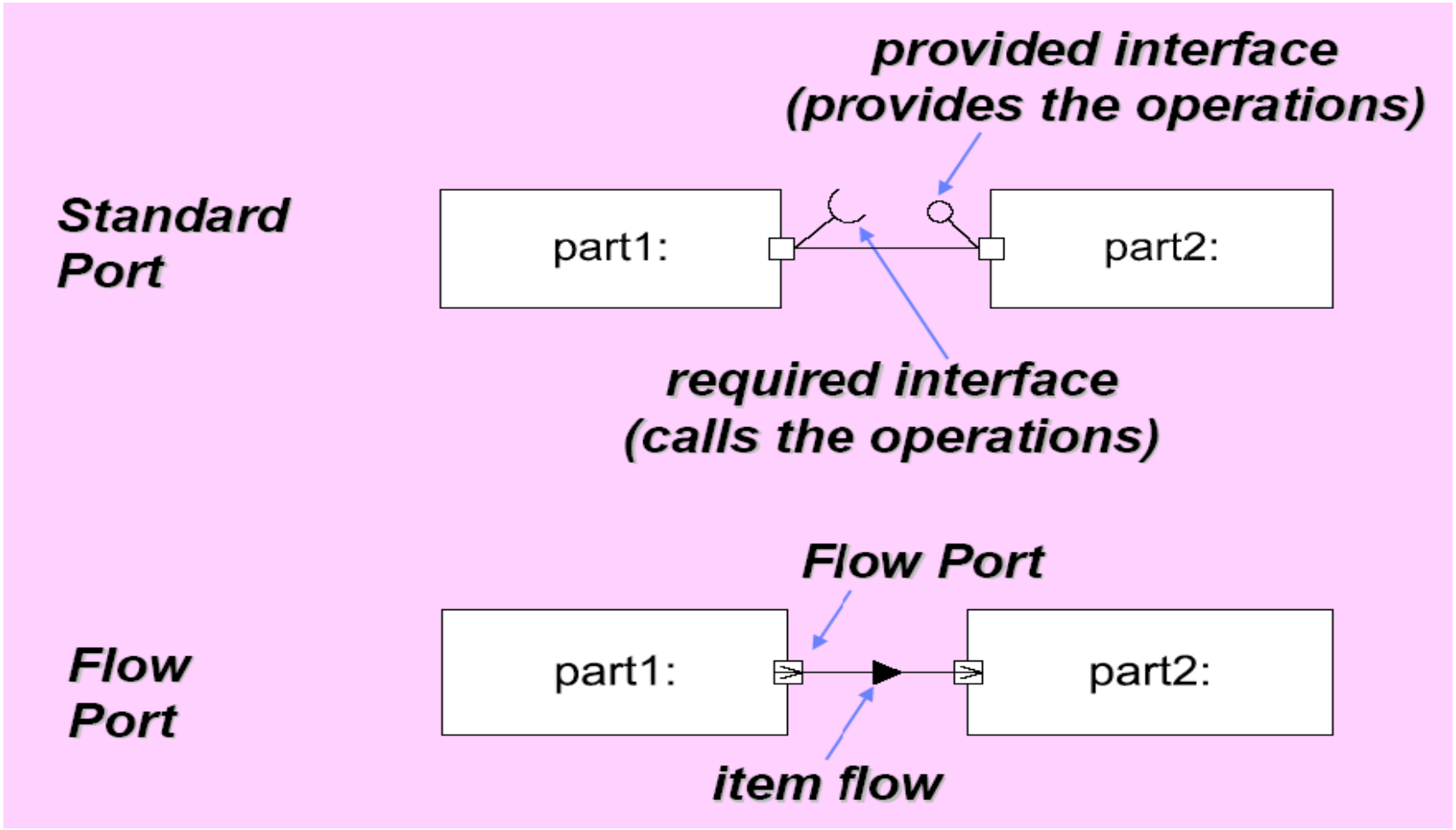


SysML-Tutorial Erik Hertzog

Port-Typen

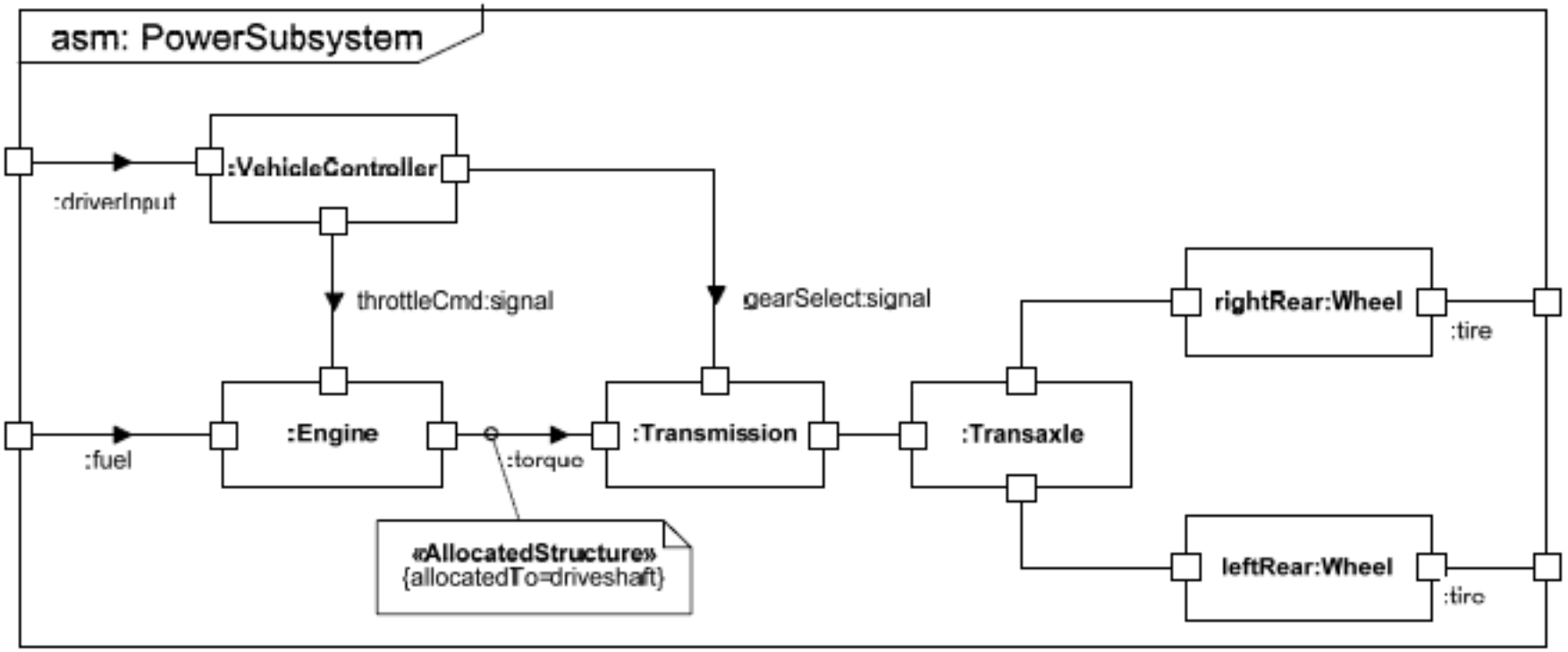
- **Standard (UML) port**
  - **The port indicate the existence of a service interface which external blocks may call (as in software)**
  - **Interaction is as defined for the individual operation made available through the interface**
- **Flow ports**
  - **Specifies what can flow in or out of a component**
  - **Has a specified direction and content**
  - **May be bi-directional**

Port-Typen (2)



SysML-Tutorial Erik Hertzog

Beispiel Internes Blockdiagramm

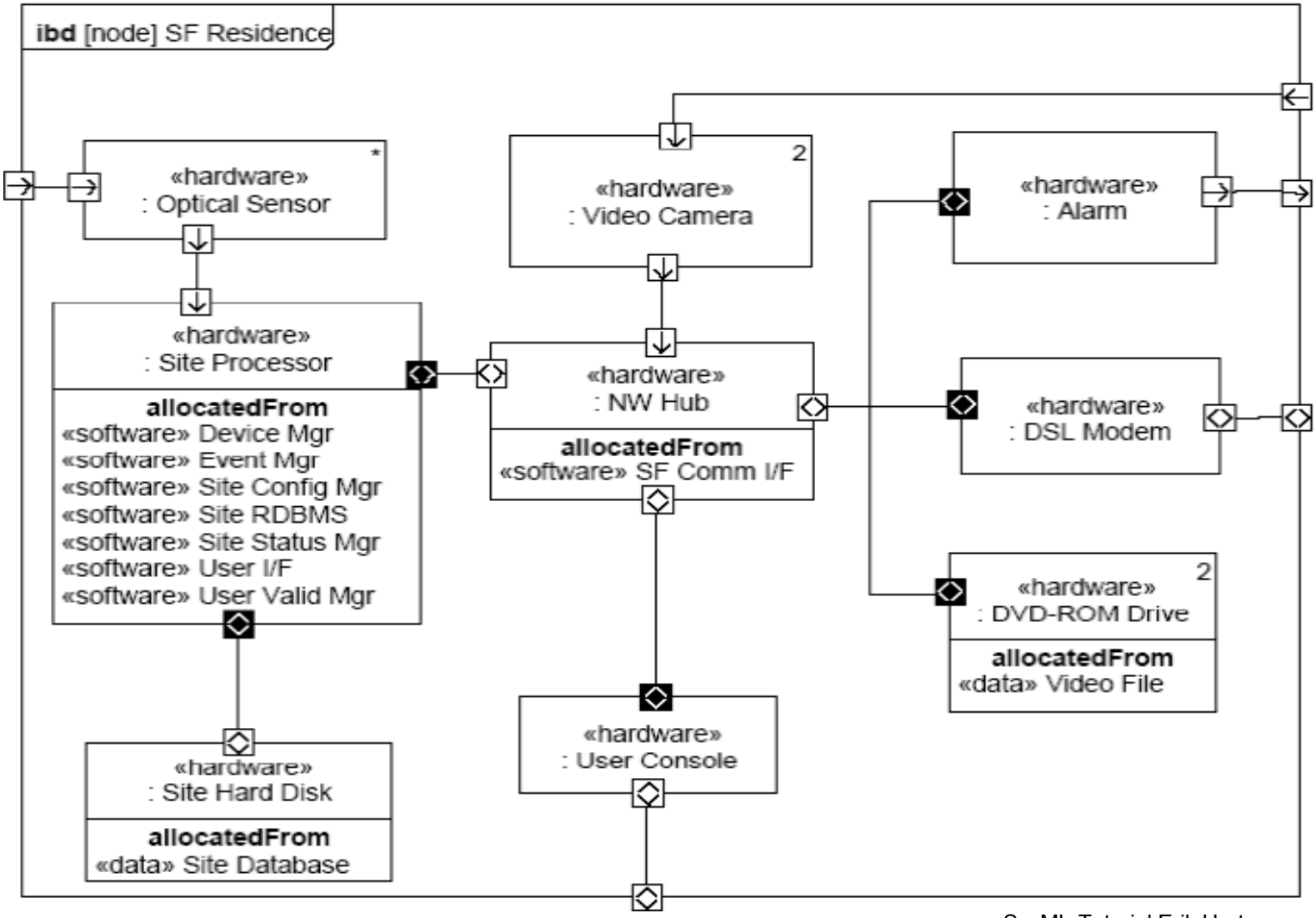


SysML 1.3 OMG 12-06-01.pdf

Allocation

- SysML provides 3 mechanisms for representing the allocation of functional or physical elements to other physical elements
  - Via Swimlanes in activity diagrams
    - » Elegant
  - Via the addition of a separate compartment in the block structure
  - Via relationships directly on diagrams

Beispiel Allocation

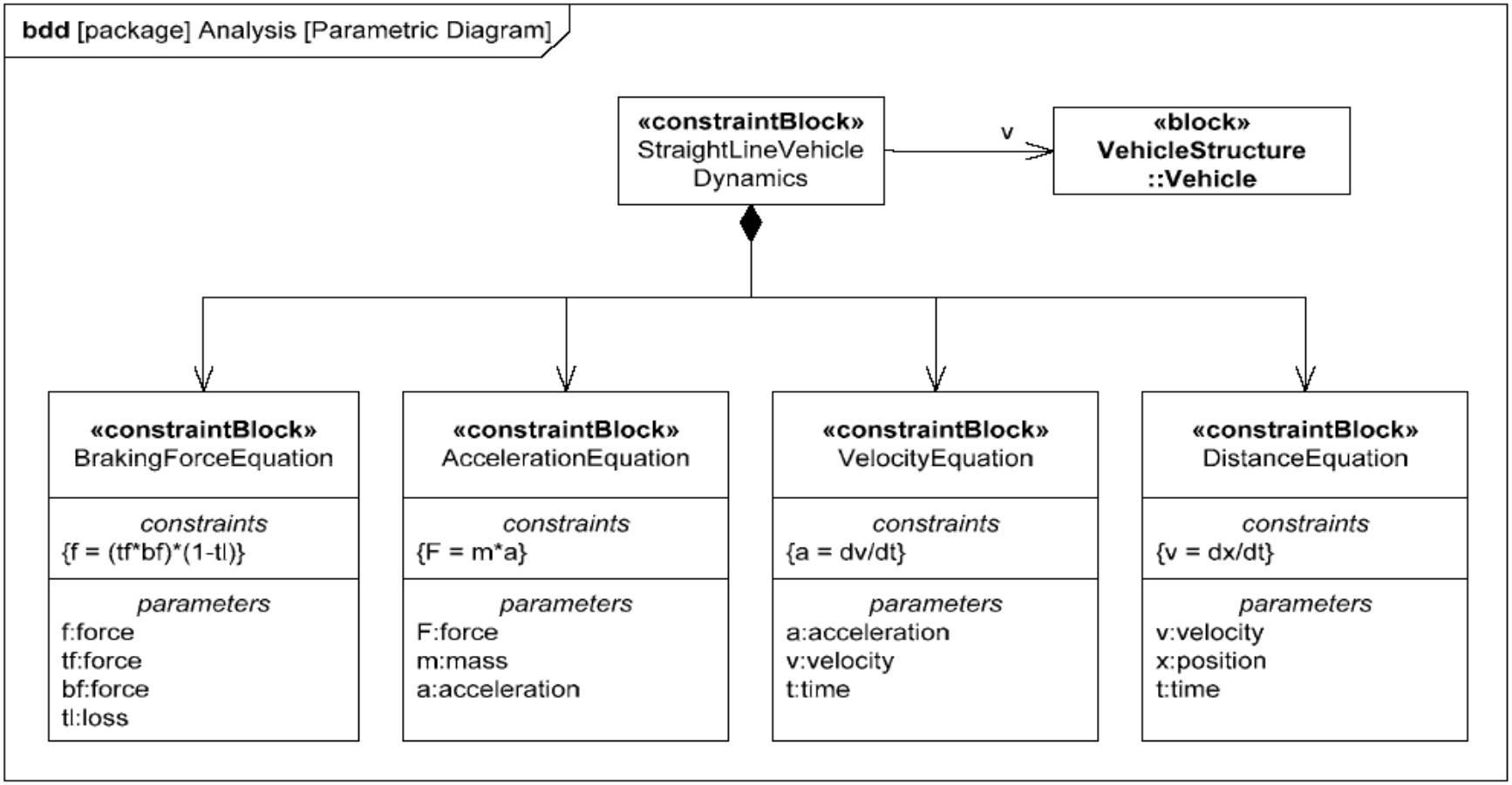


SysML-Tutorial Erik Hertzog

Parametrische Constraints

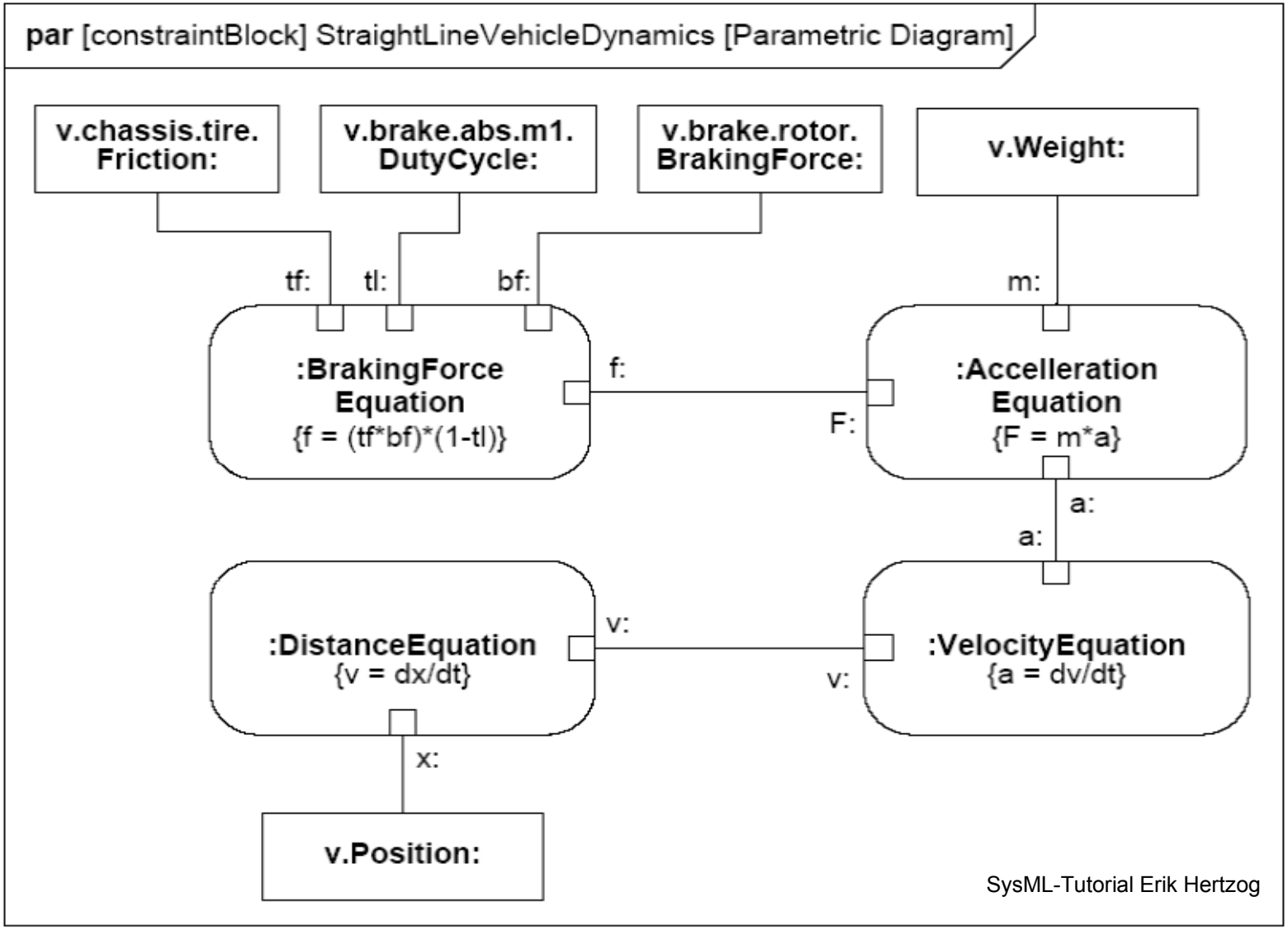
- Genutzt für Constraints zwischen quantifizierbaren Eigenschaften
- Definiert als Stereotype
  - Expression = Text String
  - Sprache für Expressions offen (informell, OCL, MathML)
  - Auswertung durch externe Tools (nicht durch SysML)

Beispiel Parametrische Constraints

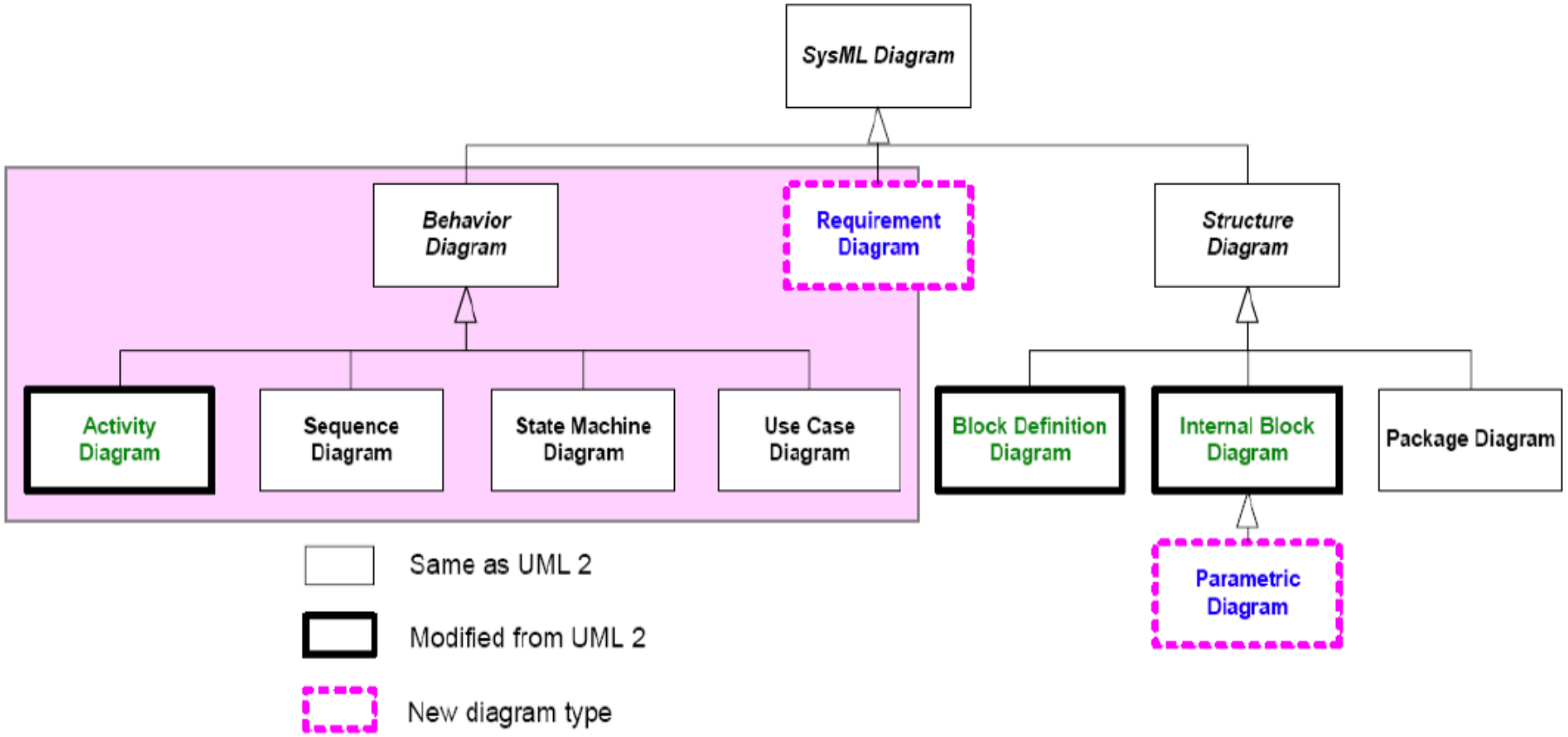


SysML-Tutorial Erik Hertzog

Beispiel Variablen-Bindung

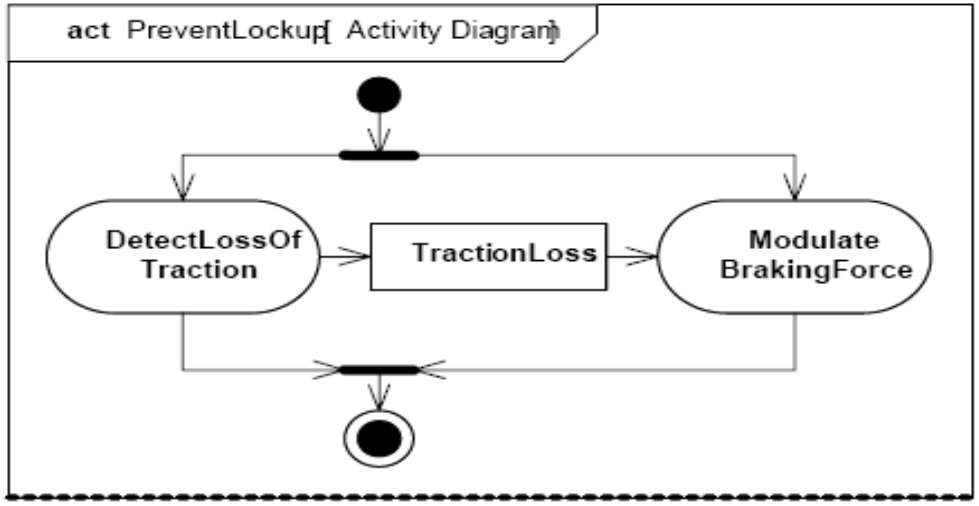


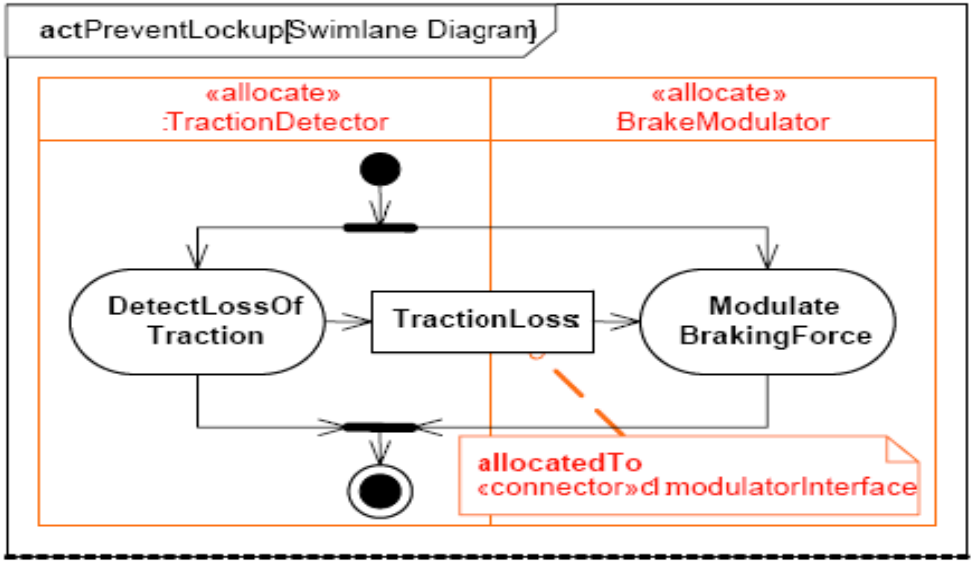
SysML Verhaltensdiagramme



SysML-Tutorial Erik Hertzog

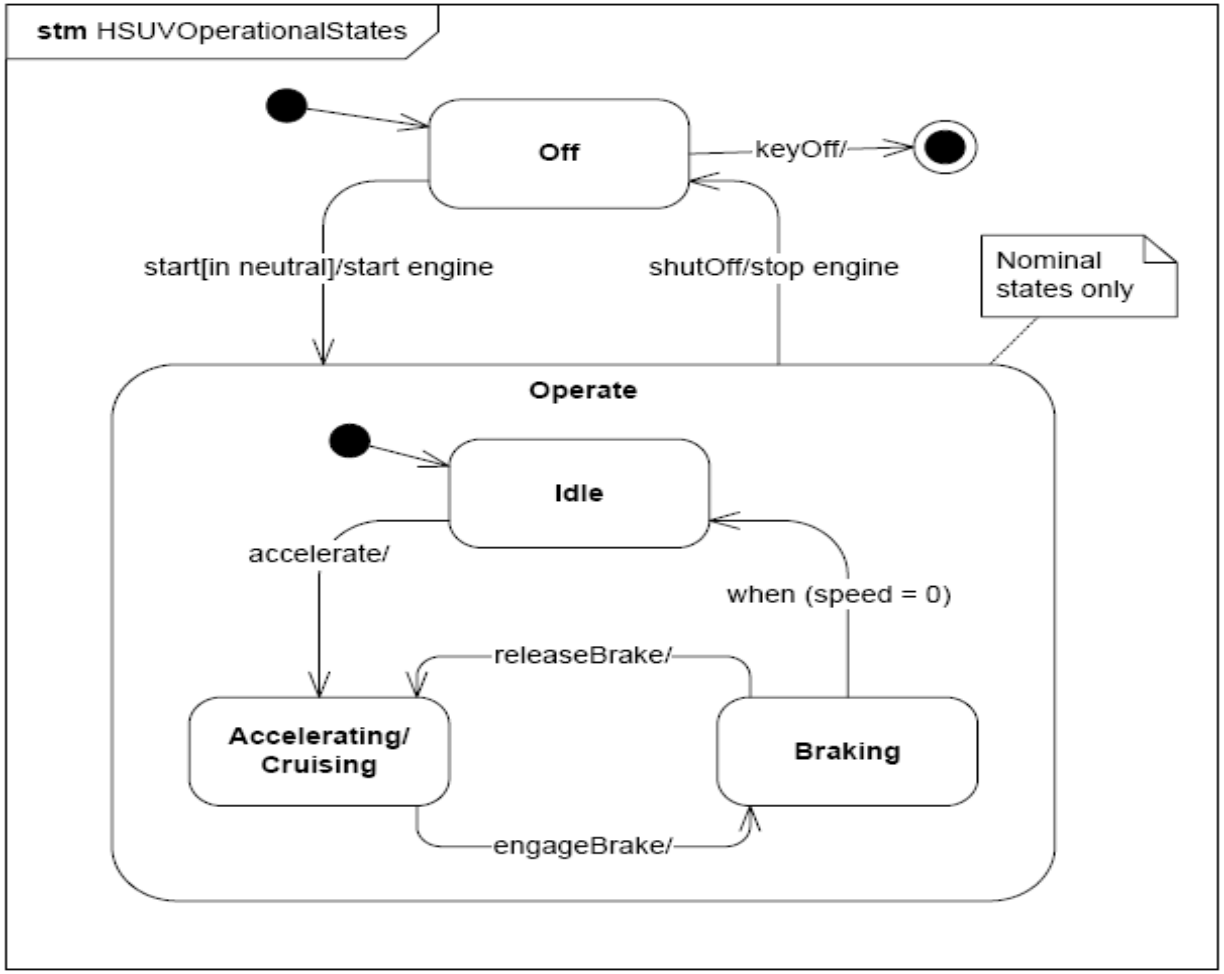
Aktivitätsdiagramme



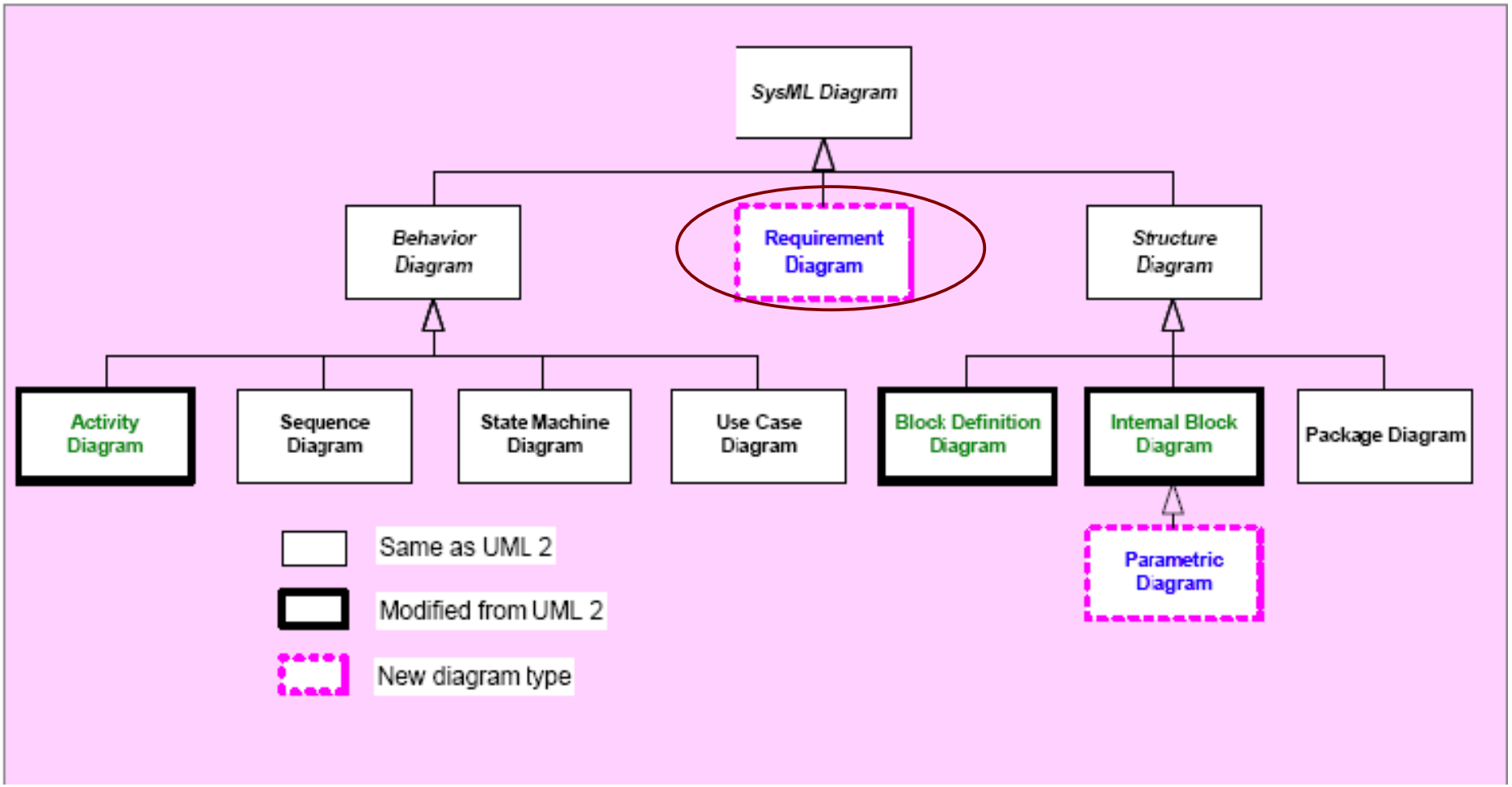


SysML-Tutorial Erik Hertzog

- **Beispiel**

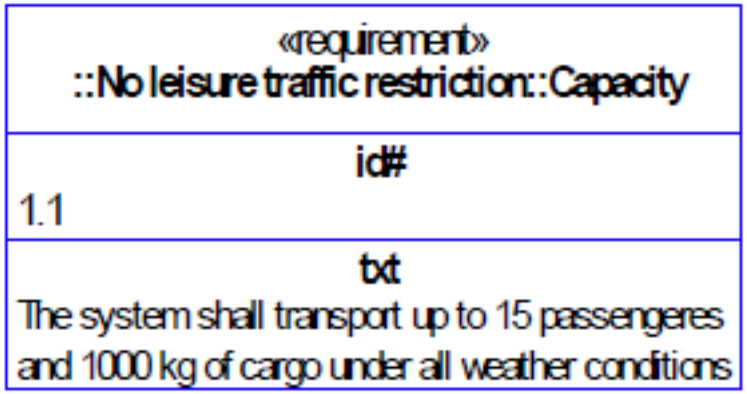

stm HSUVOperationalStates
Off
keyOff/
start[in neutral]/start engine
shutOff/stop engine
Nominal states only
Operate
Idle
accelerate/
when (speed = 0)
releaseBrake/
Accelerating/ Cruising
Braking
engageBrake/

Requirements-Diagramm



Requirement

- In SysML ist Requirement eine Zeichenkette (String)
- Keine Annahmen darüber, wann Requirement Elemente im Entwurfsprozess zu verwenden sind
- Notation:



Zusammenfassung Requirements

- **SysML provides the following features**
  - **Representation of requirements**
    - » **Representation of individual *requirements***
    - » **Requirement *composition***
    - » **Requirements can be sub-classed using specialization**
  - **Requirement relationships**
    - » ***derive* relationship between derived and source requirements**
    - » ***satisfy* relationship between design models and requirements**
    - » ***verify* relationship between requirements and test cases**
    - » **generalized *trace* relationship between requirements and other model elements**
    - » ***rationale* for requirements traceability, satisfaction, etc**
  - **Alternative graphical, tabular and tree representations**
    - » **Supported by the standard, but currently not implemented in any tools**

Stereotypes

- **Überblick**

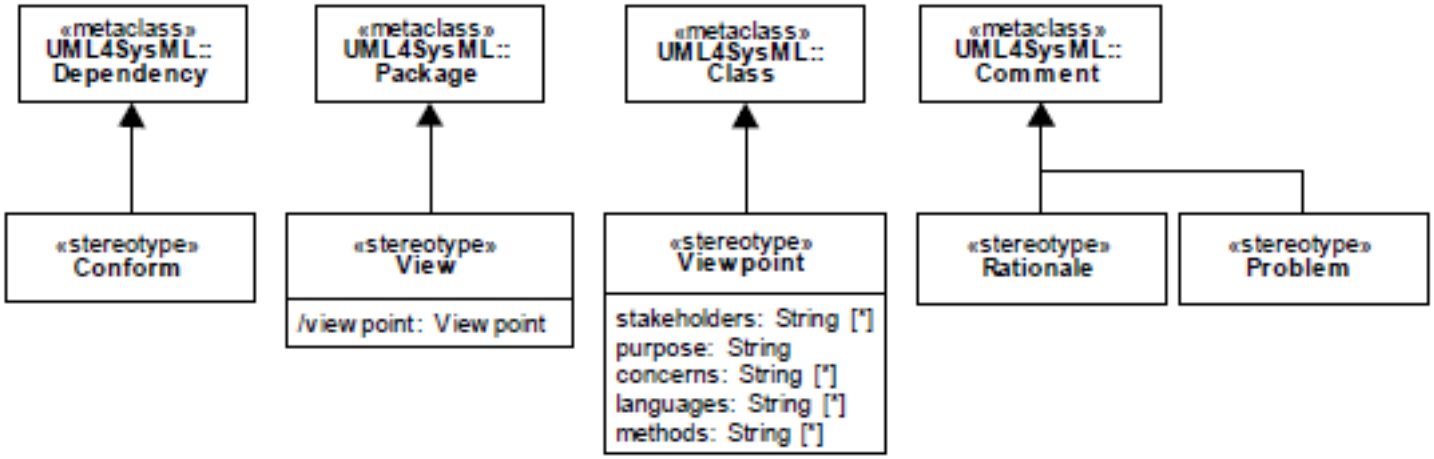


Figure 7.1 - Stereotypes defined in package ModelElements

SysML 1.3 OMG 12-06-01.pdf

Beispiel

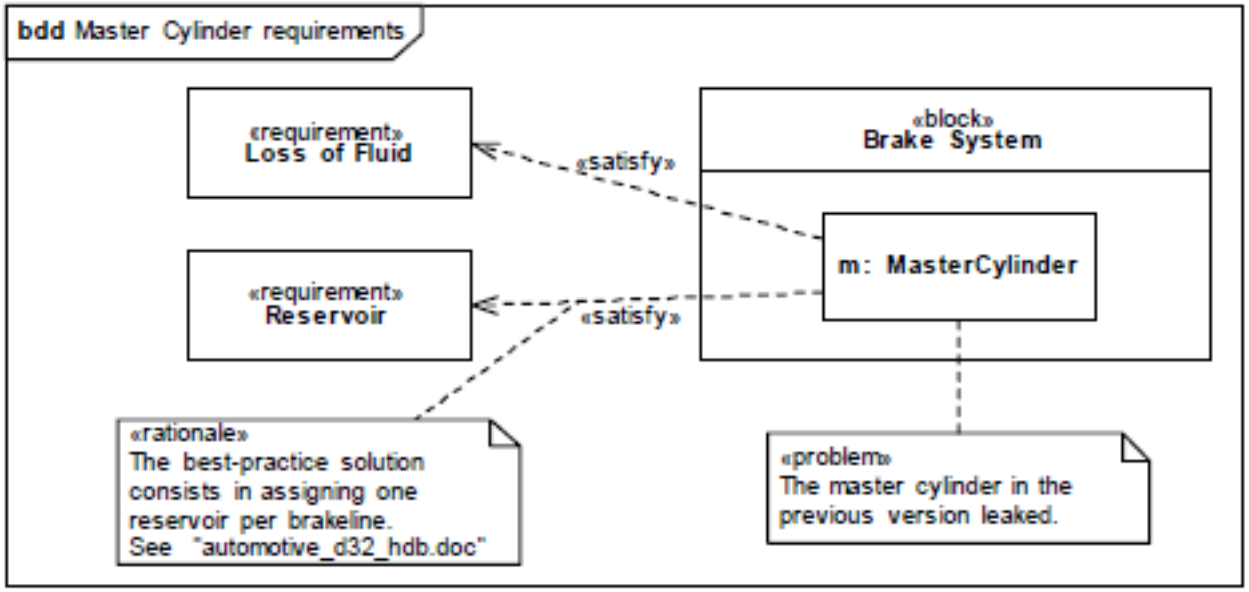


Figure 7.2 - Rationale and Problem examples

SysML 1.3 OMG 12-06-01.pdf

7.4.3 MARTE

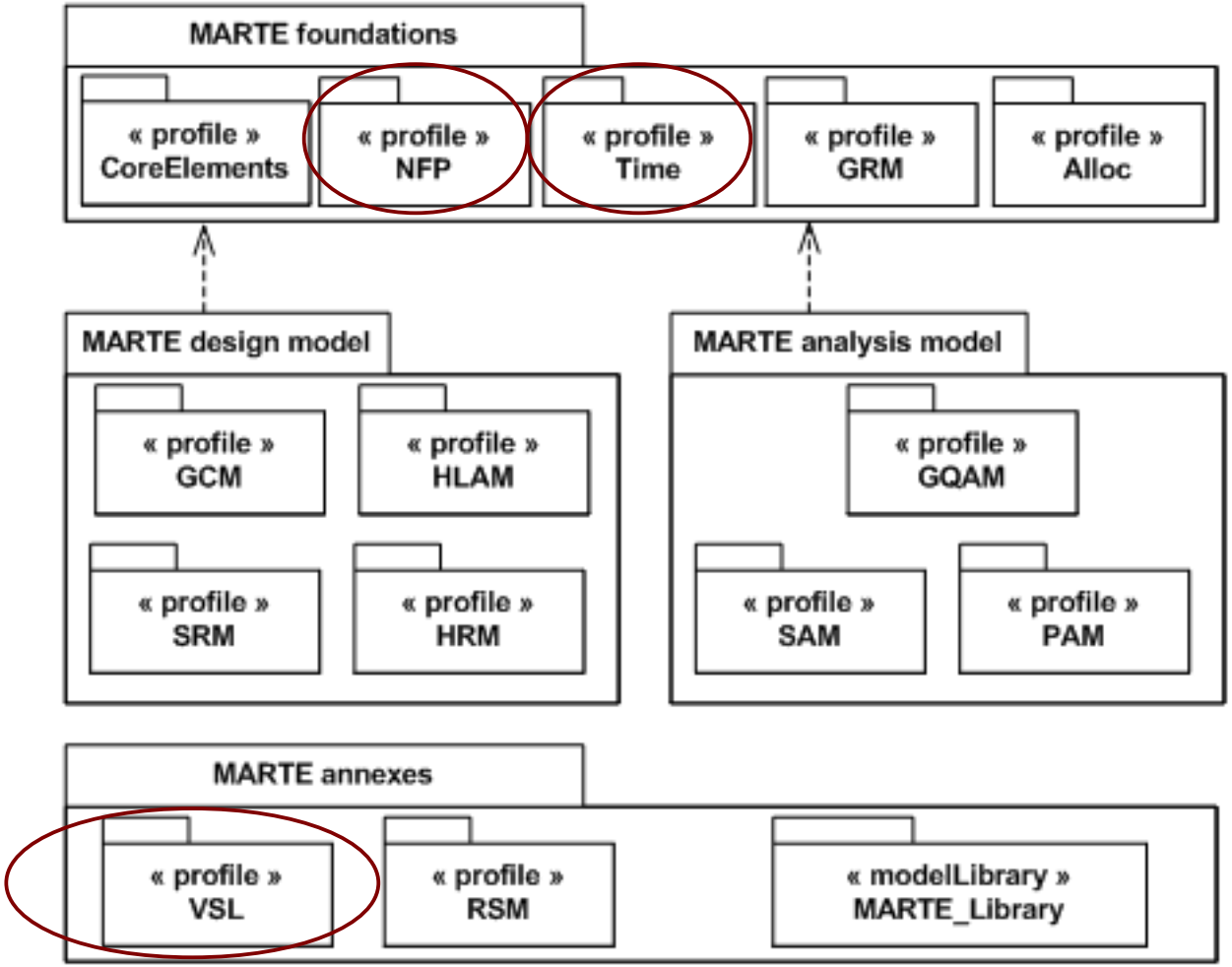
- MARTE: Modeling and Analysis of Real Time and Embedded systems
- OMG Standard formal/2011-06-02 (Version 1.1)
- UML 2 Profil
- Unterstützung für Modellierung, Simulation und statische Analyse
- Zahlreiche MARTE Modell-Bibliotheken
  - Primitive Typen
  - Erweiterte Datentypen
  - Maßeinheiten
  - Vordefinierte NFP-Typen
  - Time
  - Echtzeitbetriebssysteme (OSEK/VDX, Arinc-653 (Avionik))
  - ...

MARTE Hauptkomponenten

- Core Framework zur Definition der Basiskonzepte
  - Core Elements
  - Non-Functional Properties (NFP) Modeling
  - Time Modeling (Time)
  - Generic Resource Modeling (GRM)
  - Allocation Modelling (Alloc)
- Refinement 1: Reine Applikationsmodellierung (z.B. Hardware and Software Plattform Modellierung)
- Refinement 2: Unterstützung für quantitative Analyse of UML2-Modellen, insb. Schedulability und Performance-Analyse
- Anhänge
  - insb. Value Specification Language (VSL)
  - Sprache zum Zusammenfügen der Modellbibliotheken

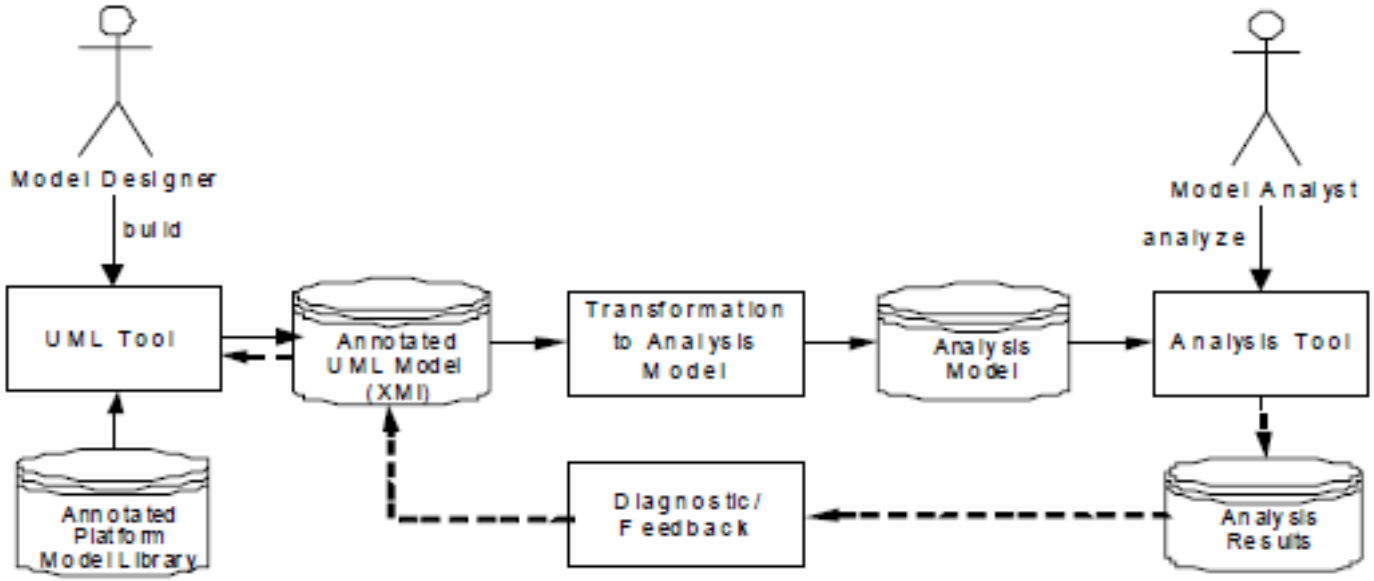
MARTE Paketstruktur

Überblick



UML MARTE OMG 11-06-02.pdf

MARTE Toolchain für Modellanalyse



UML MARTE OMG 11-06-02.pdf

Value Specification Language Framework 7.4.3

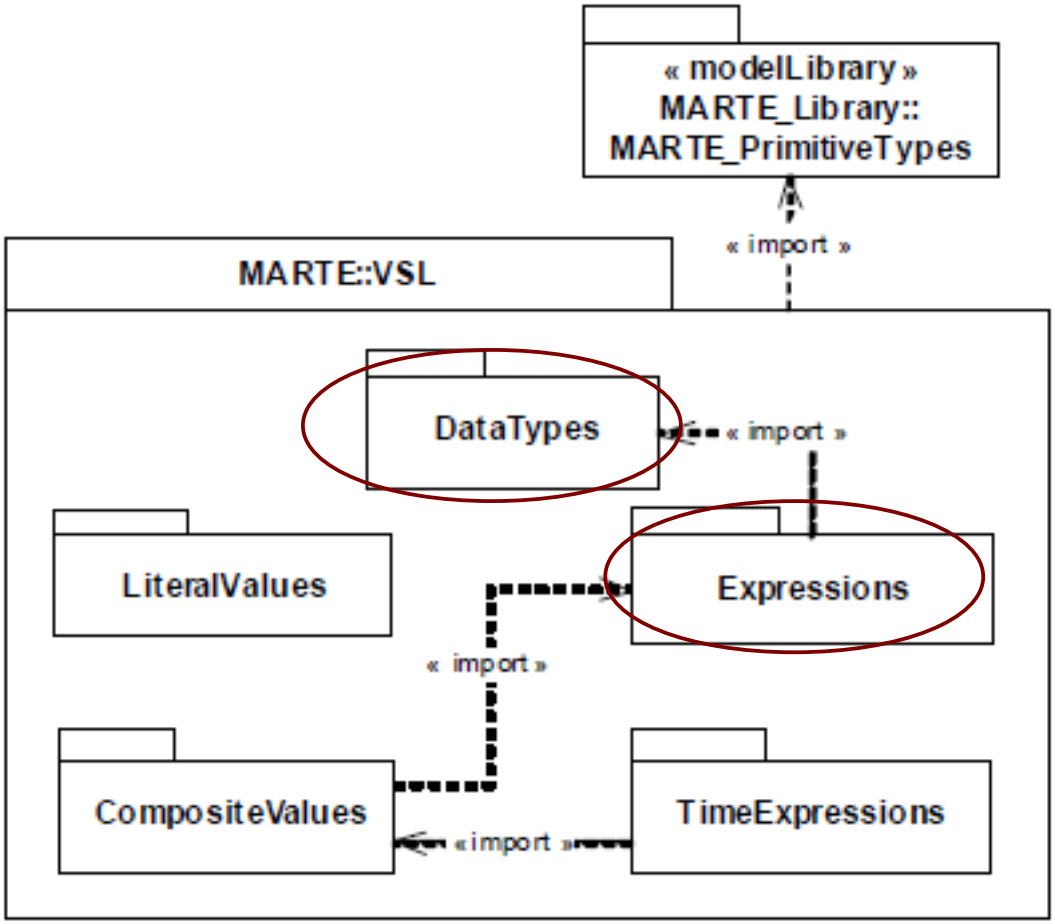


Figure B.1 - Structure of the VSL framework

UML MARTE OMG 11-06-02.pdf

VSL-Datentypen

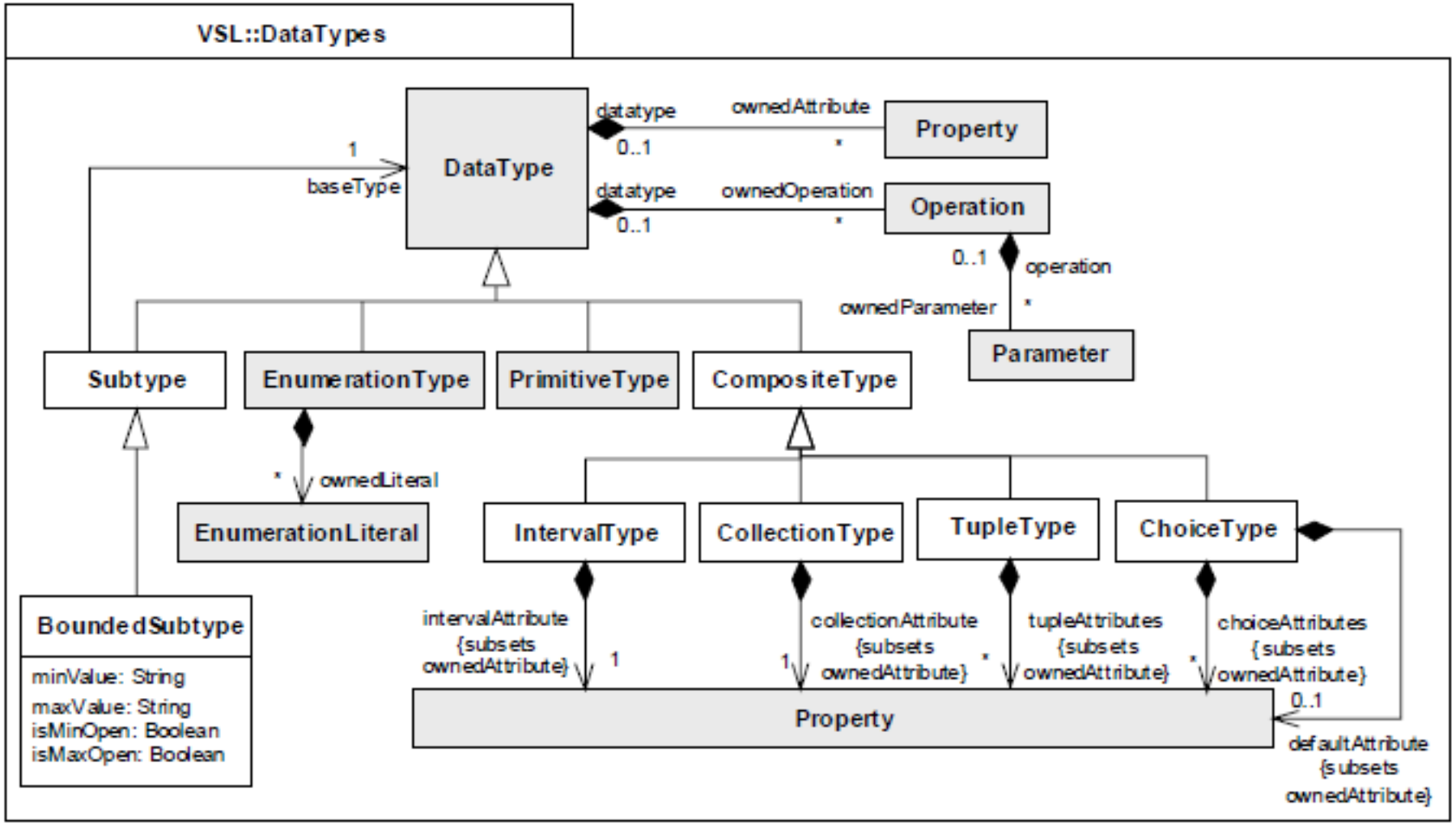


Figure B.2 - VSL::DataTypes package

UML MARTE OMG 11-06-02.pdf

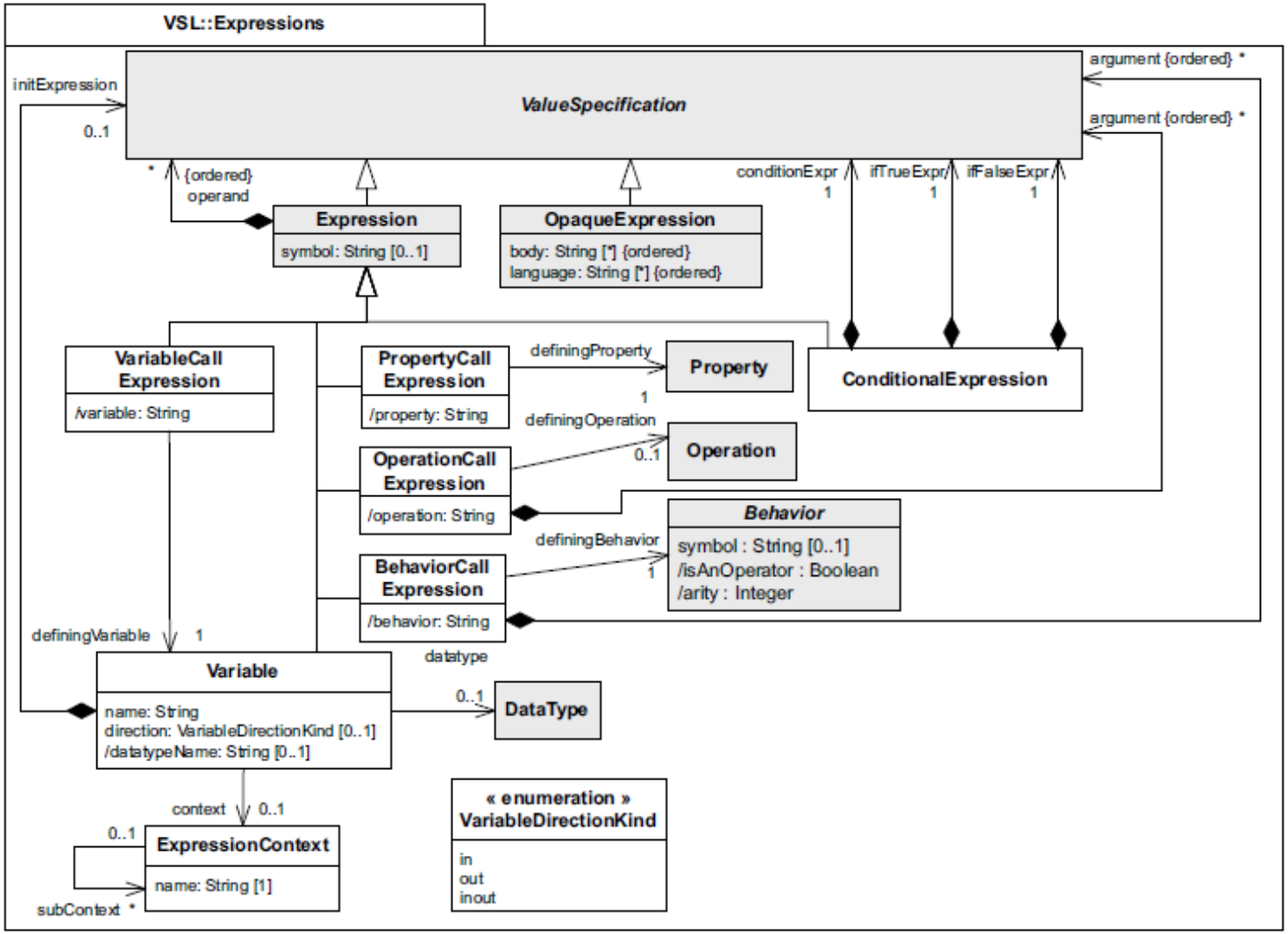


Figure B.4 - VSL::Expressions package

Non-Functional Properties (NFC) Modelling 7.4.3

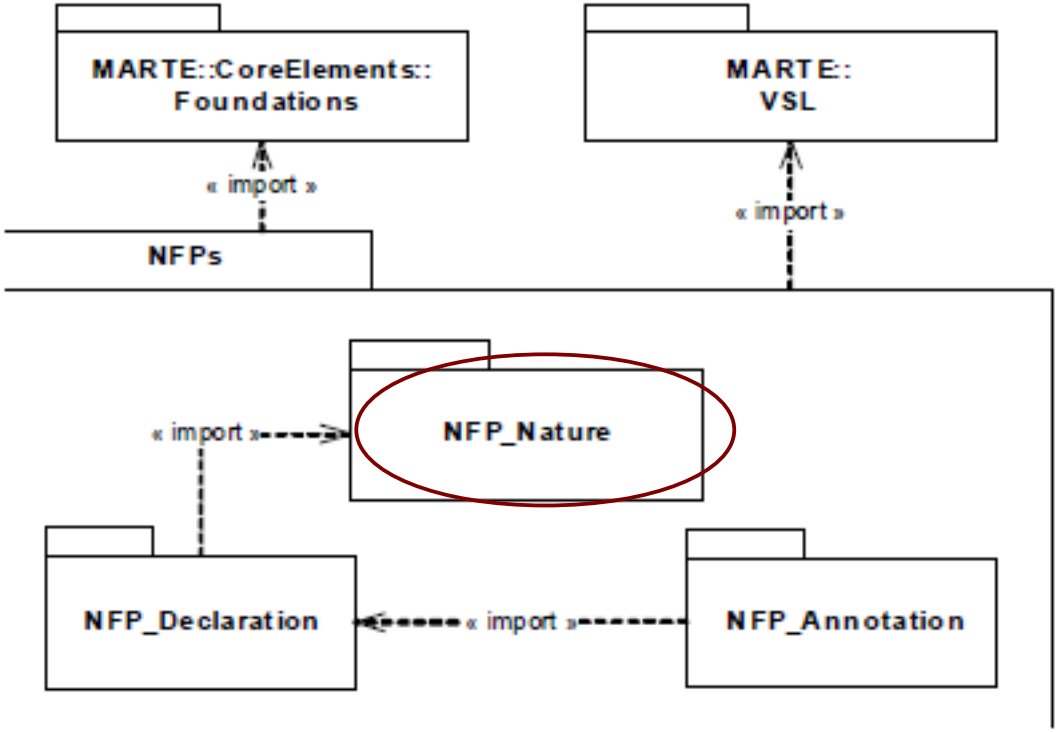


Figure 8.1 - Structure and dependencies of the NFPs modeling package

UML MARTE OMG 11-06-02.pdf

Non-Functional Properties (NFC) Modelling (2) 7.4.3

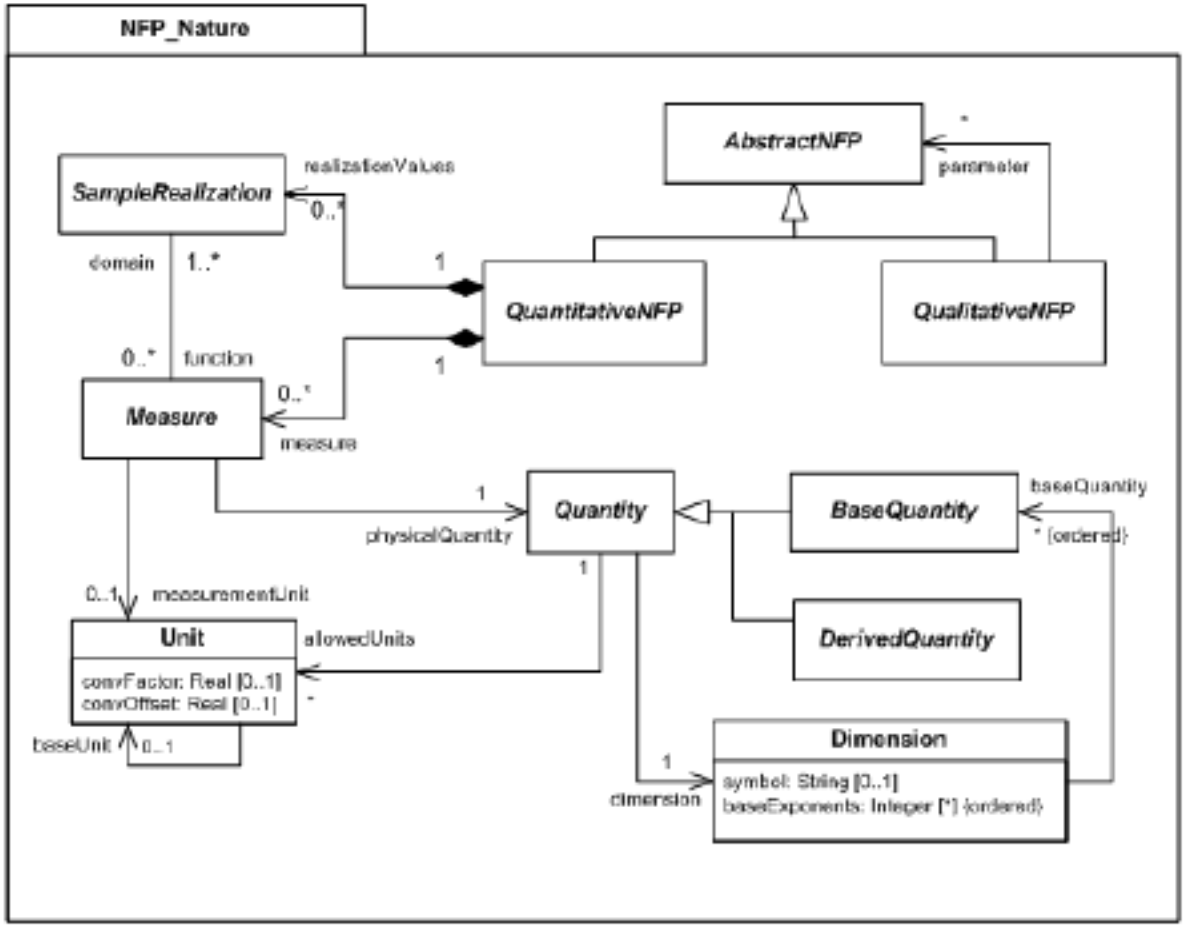


Figure 8.2 - Domain Model for NFP Nature

UML MARTE OMG 11-06-02.pdf

Zeitmodell (Time)

- **Überblick**

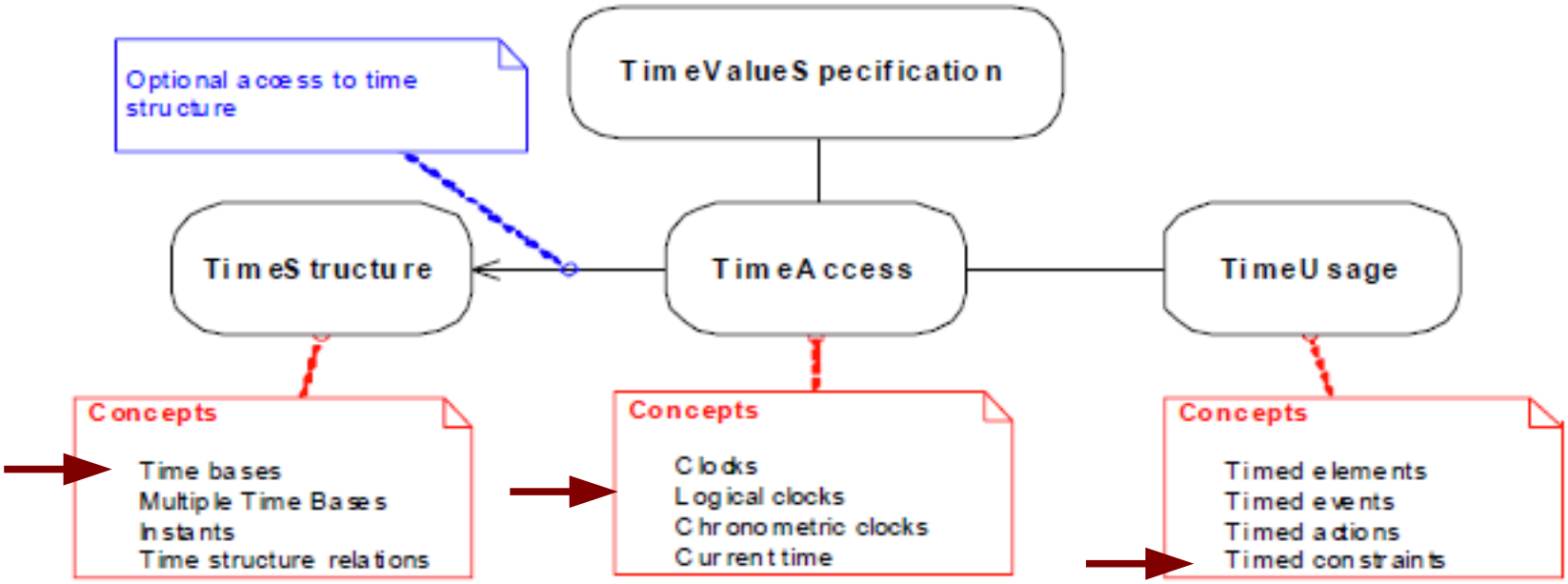


Figure 9.1 - Overview of the time model concerns

UML MARTE OMG 11-06-02.pdf

Basic Time Models

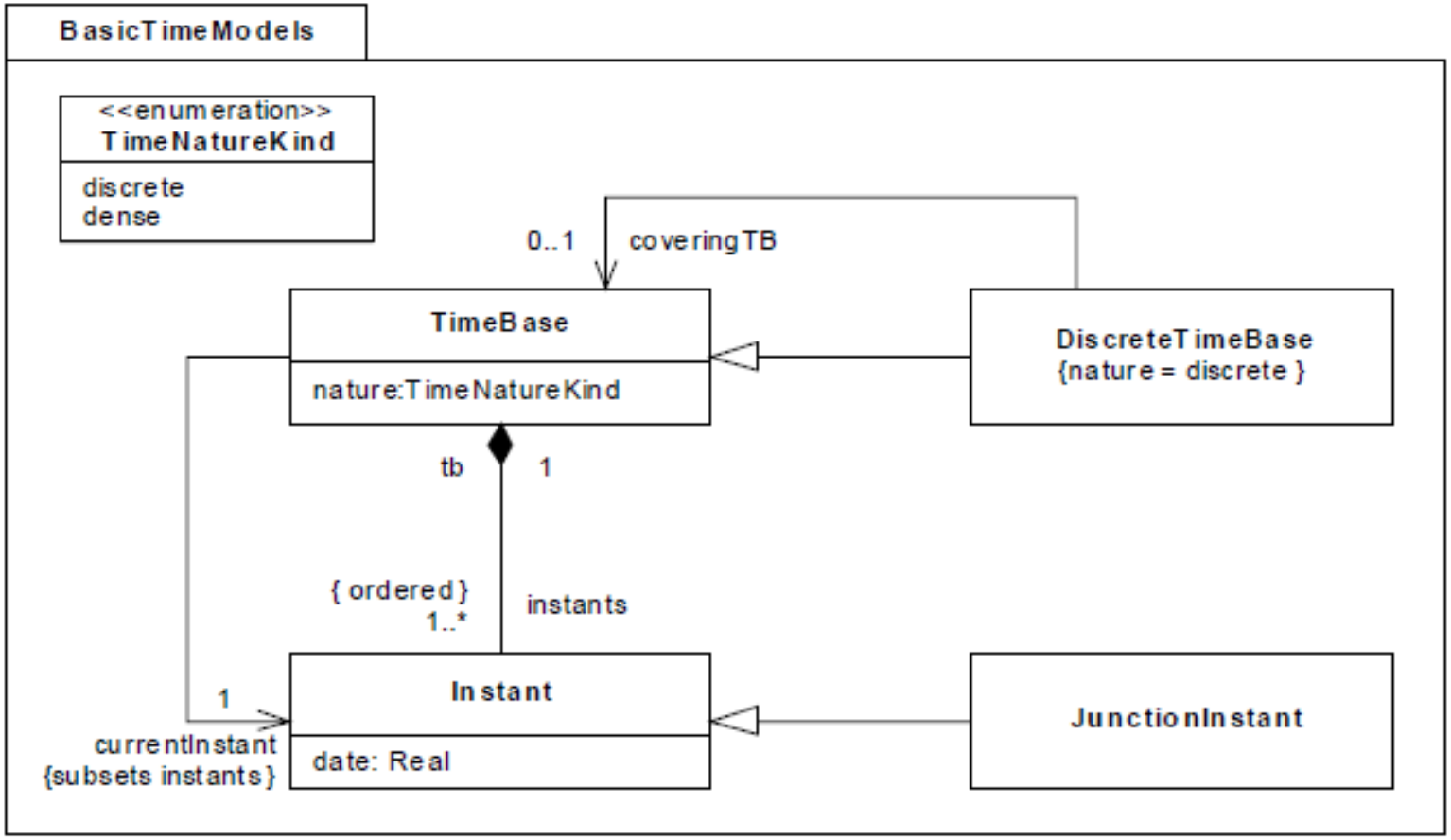


Figure 9.3 - Basic time diagram of the time model

UML MARTE OMG 11-06-02.pdf

Uhren (Clocks)

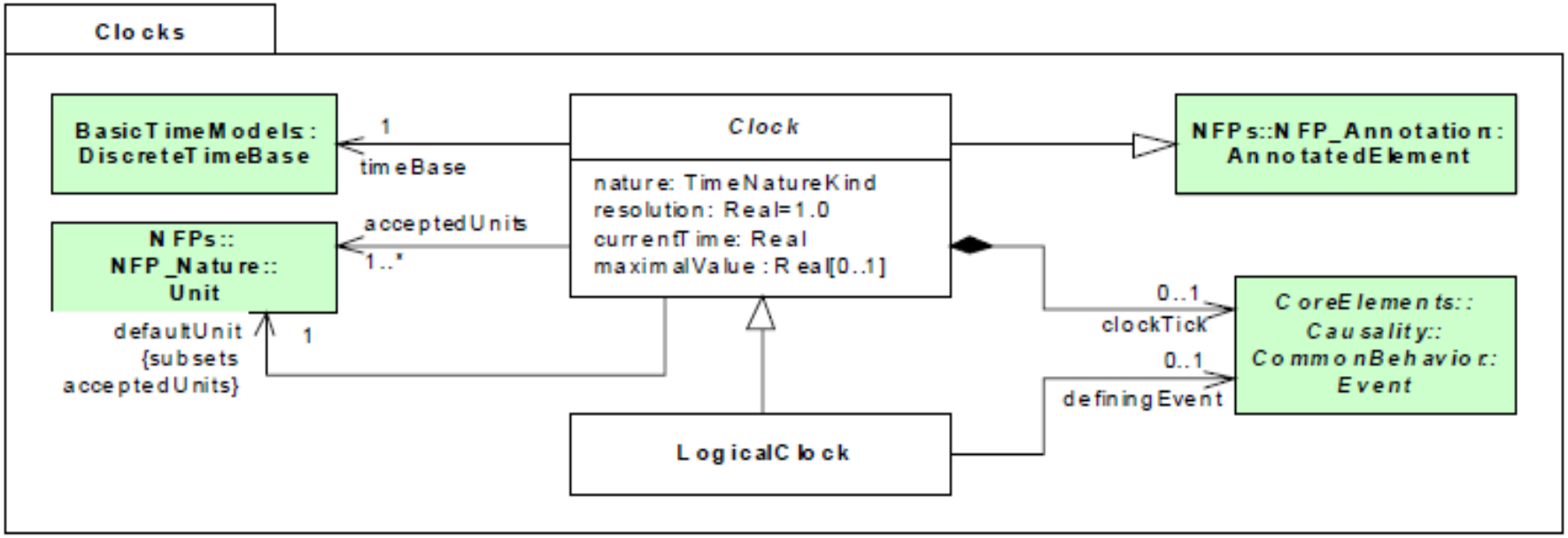


Figure 9.9 - Clocks diagram of the time model

UML MARTE OMG 11-06-02.pdf

Uhren (Clocks) (2)

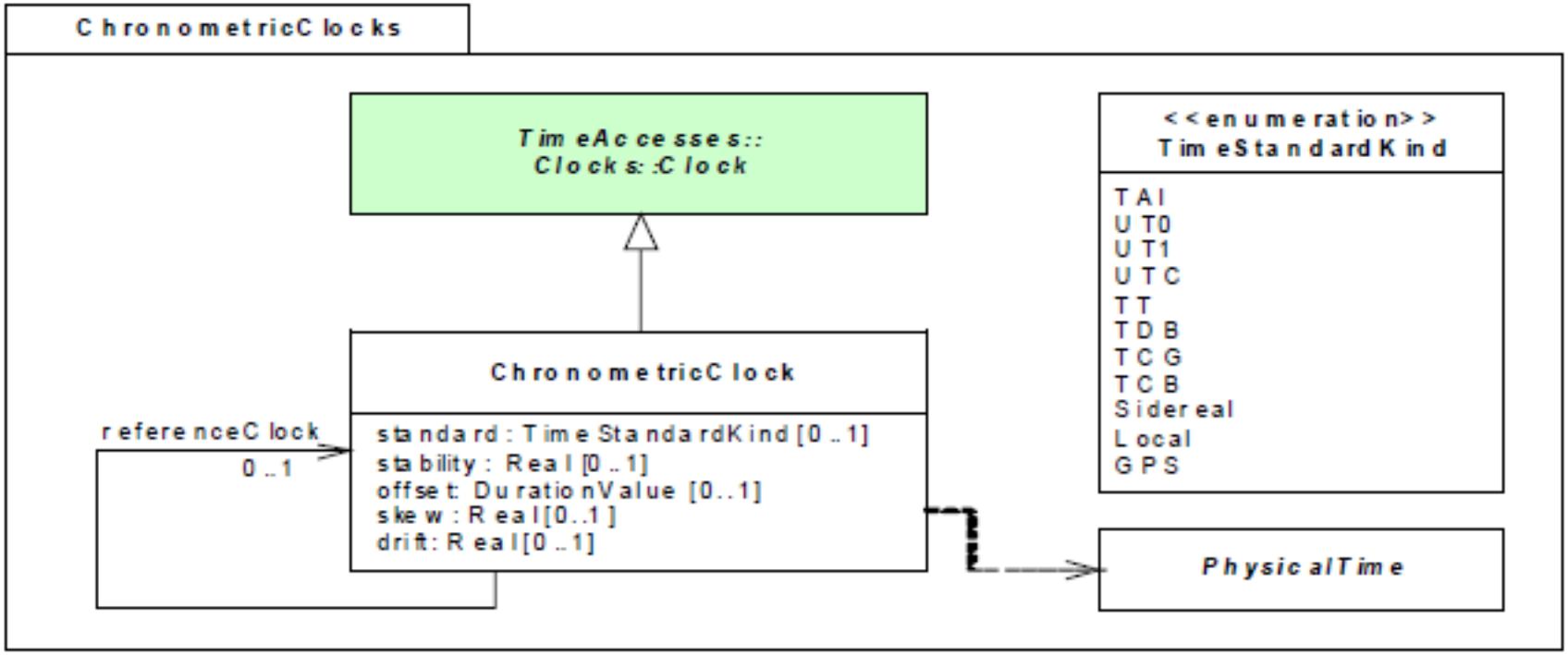


Figure 9.12 - ChronometricClocks diagram of the time model

UML MARTE OMG 11-06-02.pdf

Uhren (Clocks) (3)

- **Beispiel**

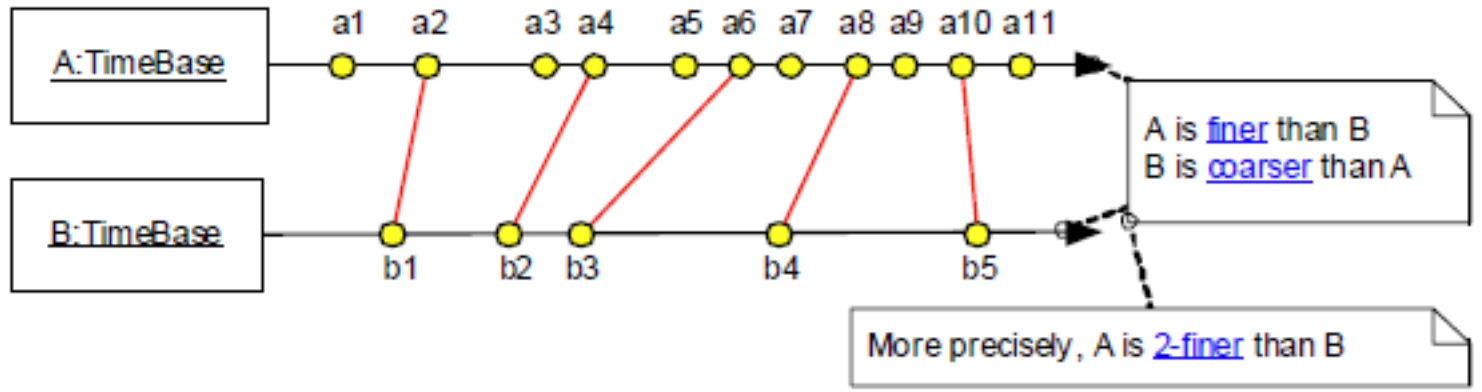


Figure 9.7 - Example of time relations between two time bases

UML MARTE OMG 11-06-02.pdf

Timed Constraints

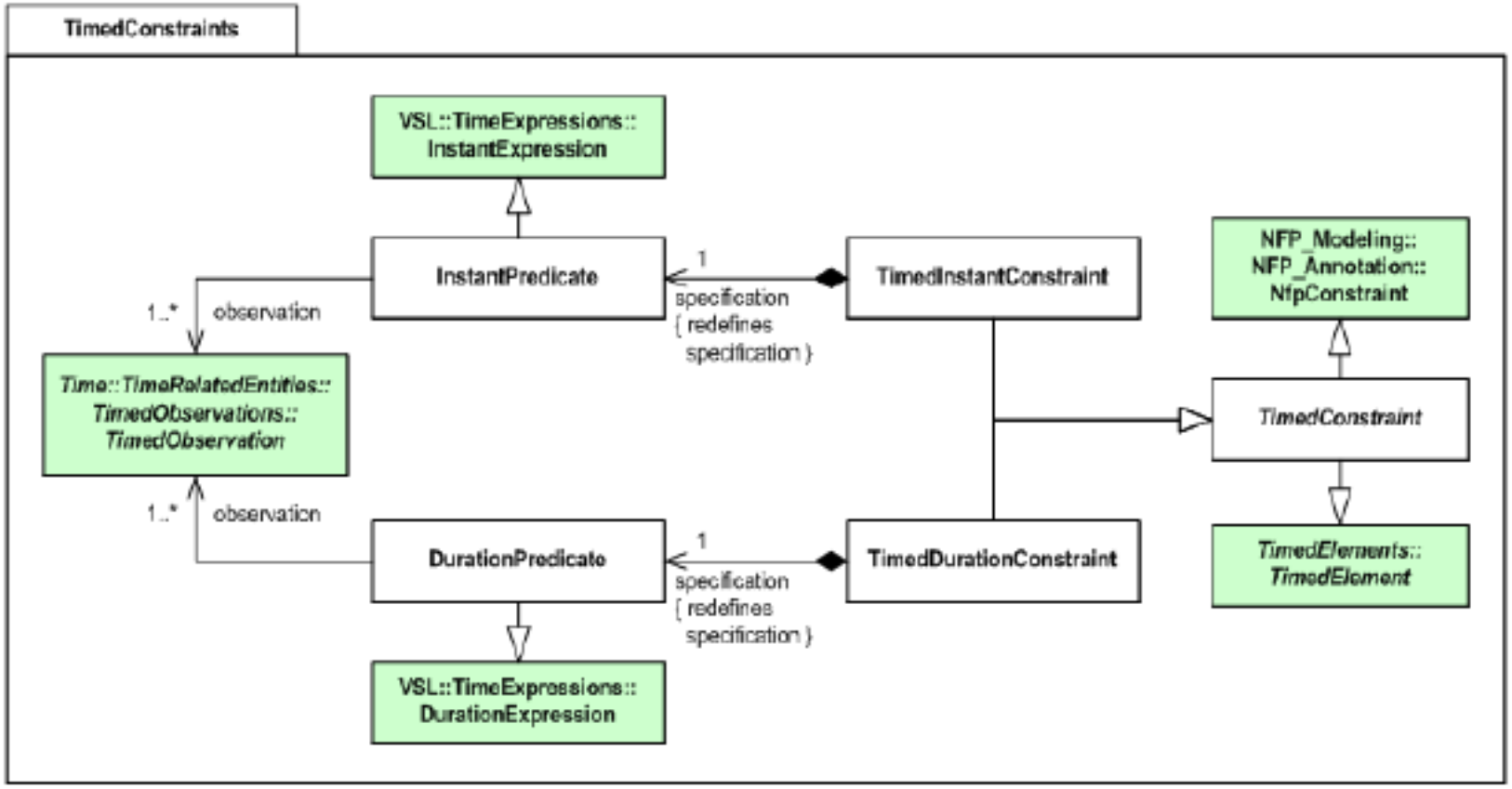


Figure 9.18 - TimedConstraints diagram of the time model

UML MARTE OMG 11-06-02.pdf

Timed Constraints (Beispiel)

- **Zusammenfassendes Beispiel**

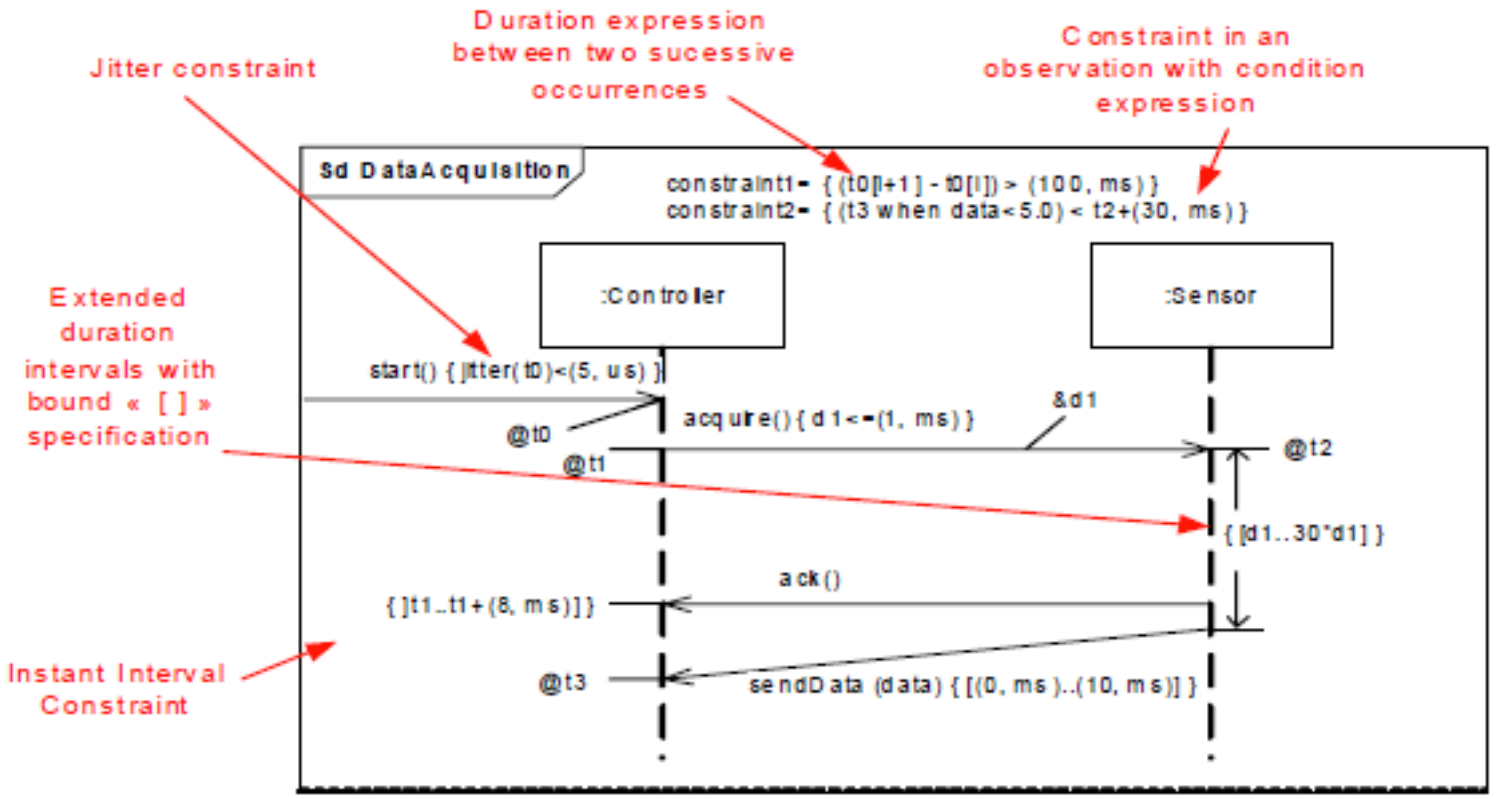


Figure B.14 - Time Expressions and Constraints in Sequence Diagrams with VSL

UML MARTE OMG 11-06-02.pdf