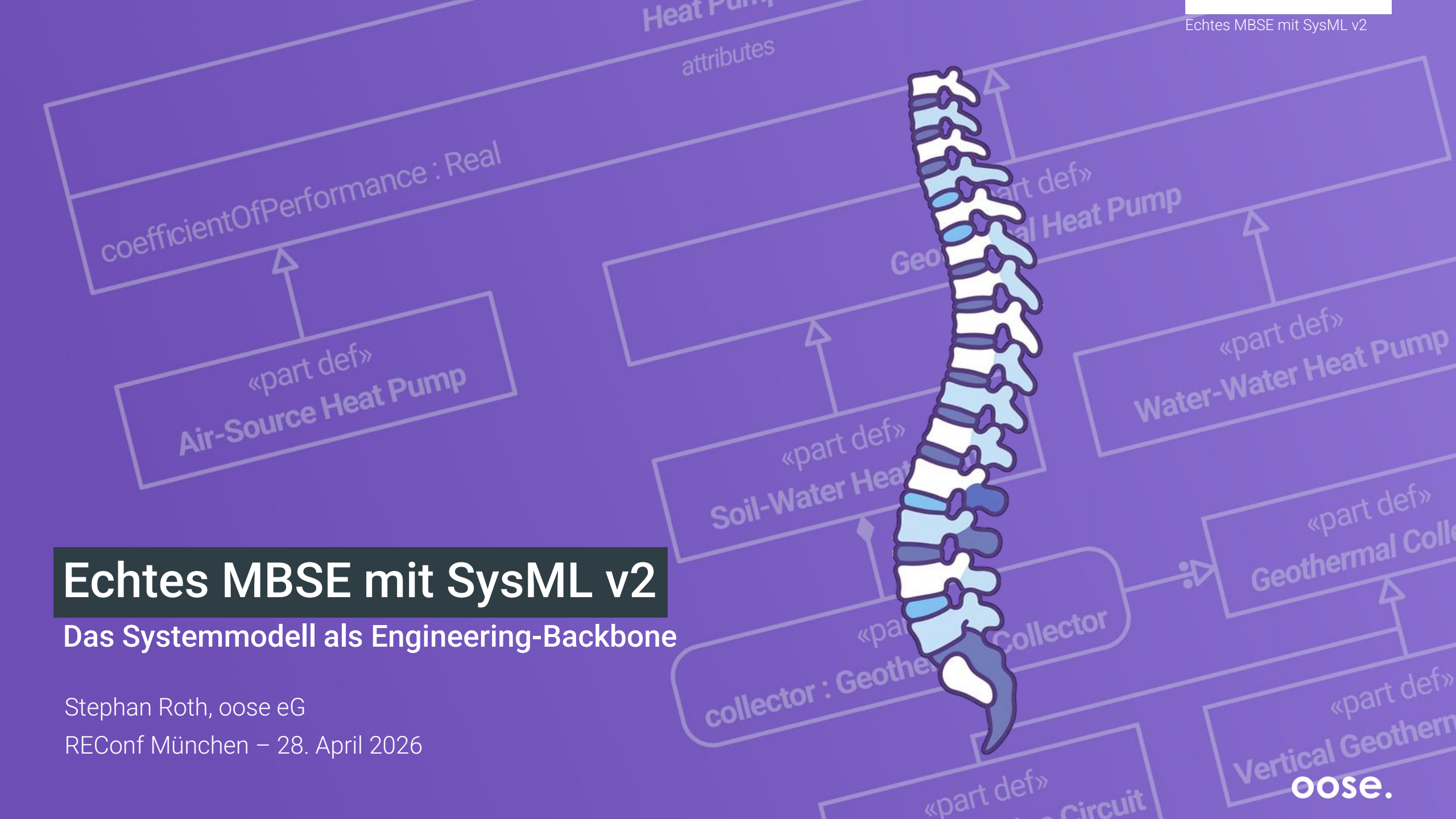


Echtes MBSE mit SysML v2

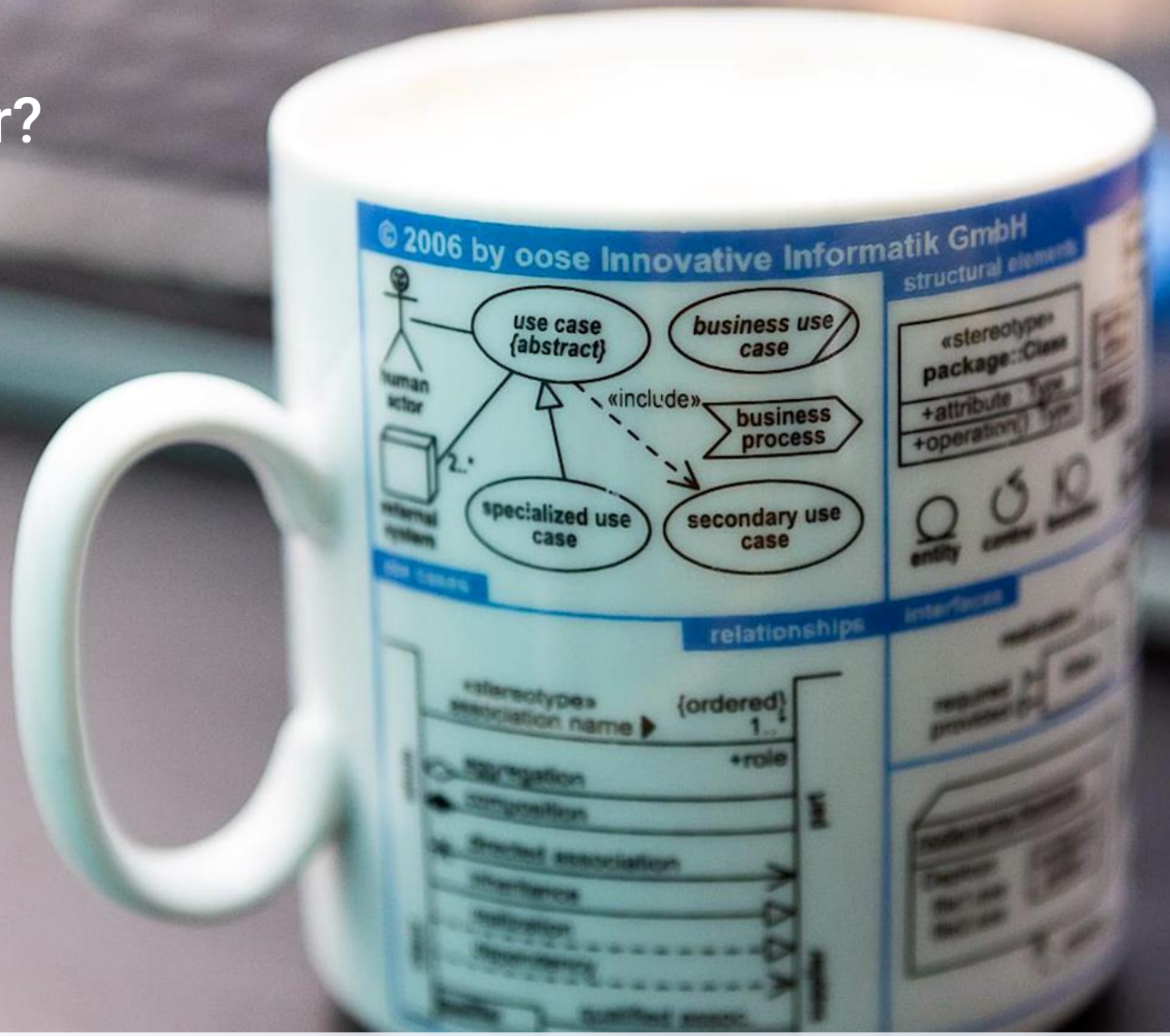
Das Systemmodell als Engineering-Backbone

Stephan Roth, oose eG

REConf München – 28. April 2026

oose.

Was tun wir?



oose bietet Trainings und Beratungen rund um viele Themen der Software-, System- und Organisationsentwicklung an.

Vom Wissen zum Können

20+ Jahre am Markt

120+ verschiedene Seminare

20.000+ Teilnehmer:innen

5.000+ Kund:innen

0 Chef:innen

Stephan Roth

Trainer & Berater

(Model-Based) Systems Engineering
Software Engineering und Architektur
Software Craft und Clean Code
Konferenzsprecher
Fachbuchautor

Mitglied bei GfSE e. V./INCOSE

Gremienmitarbeit bei ISO JTC1/SC7 (WG 4, 7 und 42)

OMG Unified Architecture Framework™ (UAF)

stephan.roth@oose.de

www.linkedin.com/in/steproth

oose.

MBSE – von 2020 bis 2035

„Systems engineers routinely compose task-specific virtual models using ontologically linked, digital twin-based model-assets. These connected models are updated in real-time providing a virtual reality-based, immersive design and exploration space."

—INCOSE SE Vision 2035 (2022)

„Model-based Systems Engineering will become the 'norm' for systems engineering."

—INCOSE SE Vision 2025 (2014)

„In many respects, the future of systems engineering can be said to be 'model-based.'"

—INCOSE SE Vision 2020 (2007)

01

Was ist eigentlich ein Modell?

(im Systems Engineering)

Gemälde: „La trahison des images“, René Magritte (1929); Foto von Caterpillar84 (Wikimedia Commons), Lizenz: CC BY-SA 4.0

Sind das Modelle?

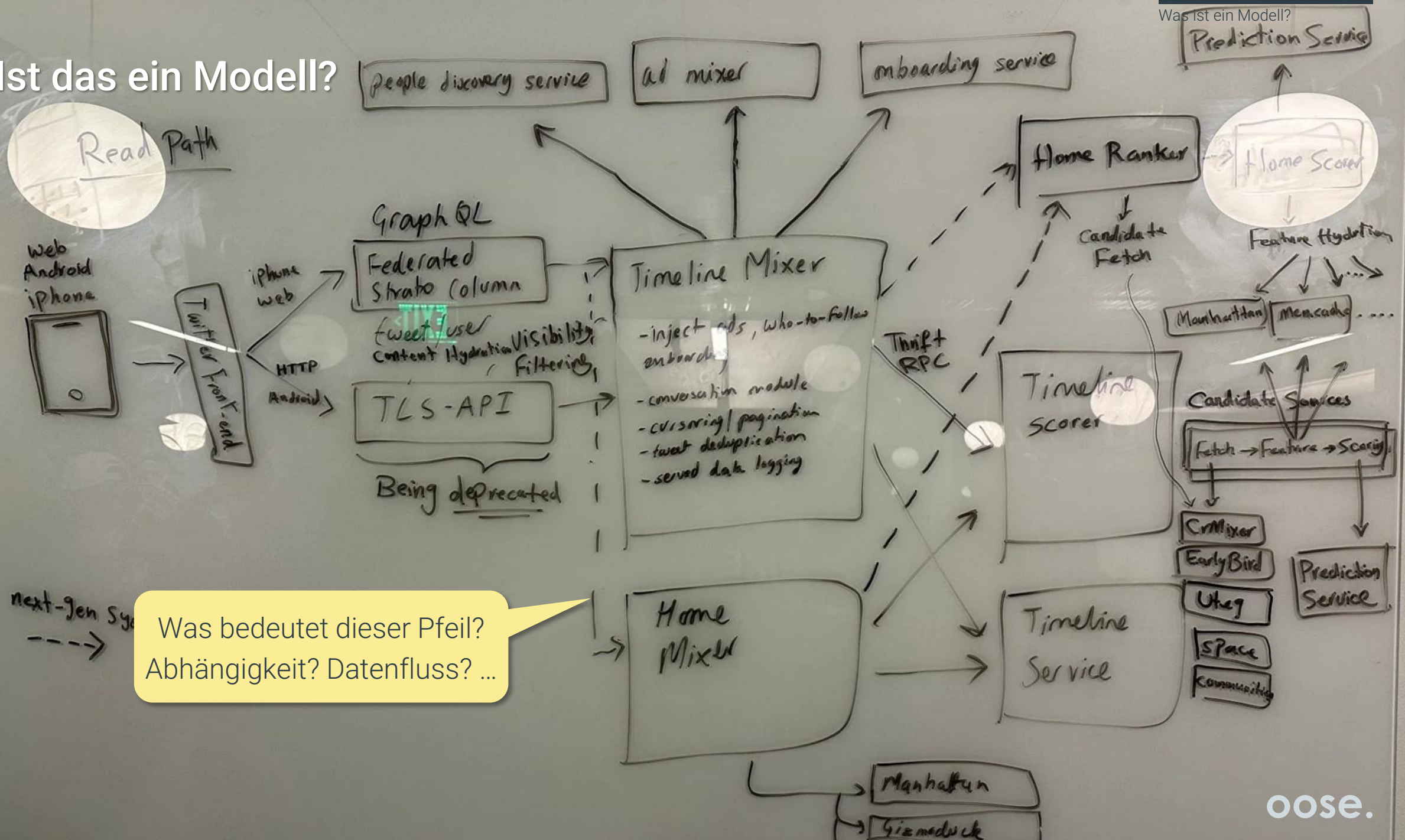
Wie halte ich das aktuell und konsistent?

Ist das noch angemessen in Anbetracht heutiger Systemkomplexität?

Ist das ein Modell?

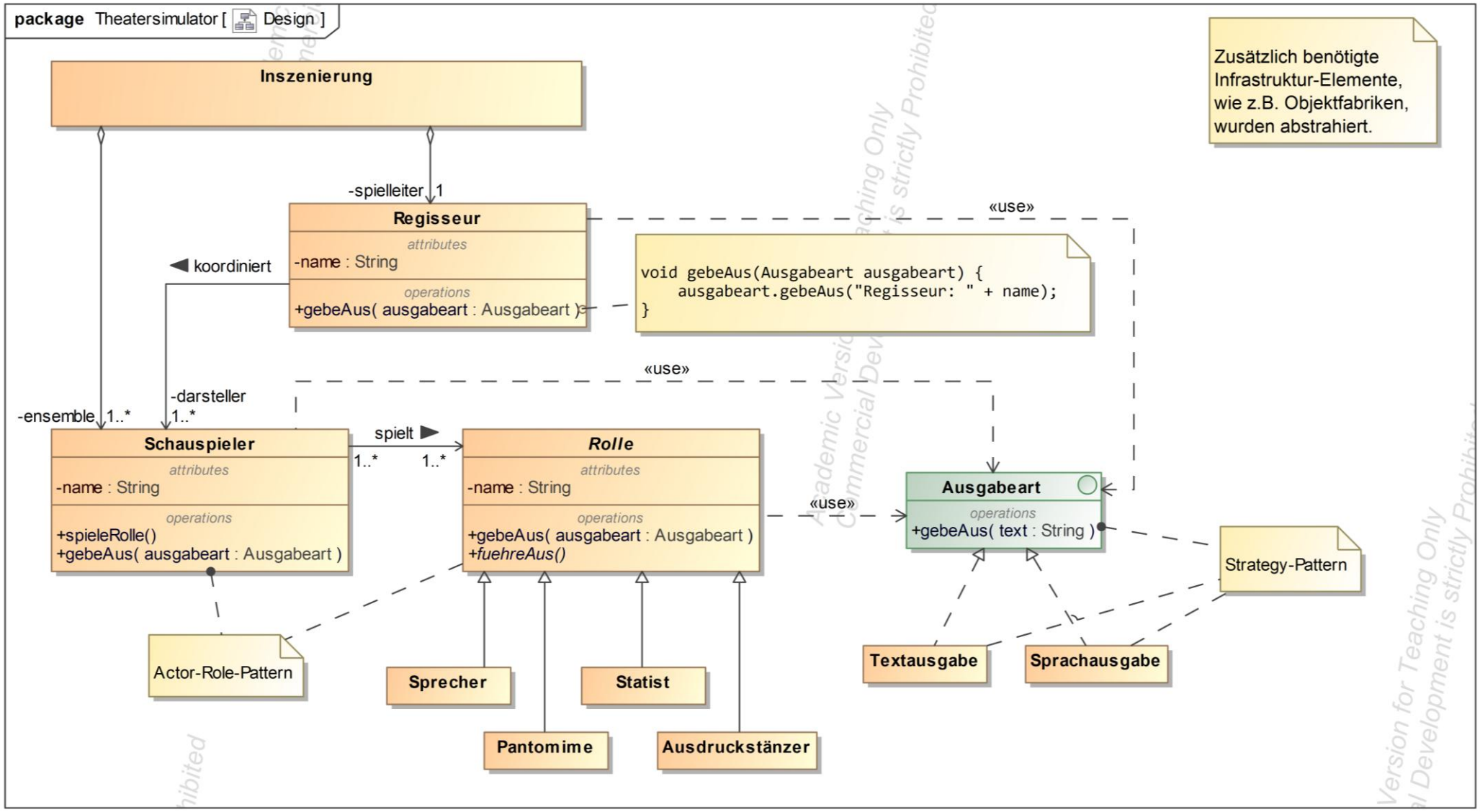
Bild: An example process from an EventStorming; Foto von Henning Schwentner (Wikimedia Commons), Lizenz: CC BY-SA 4.0

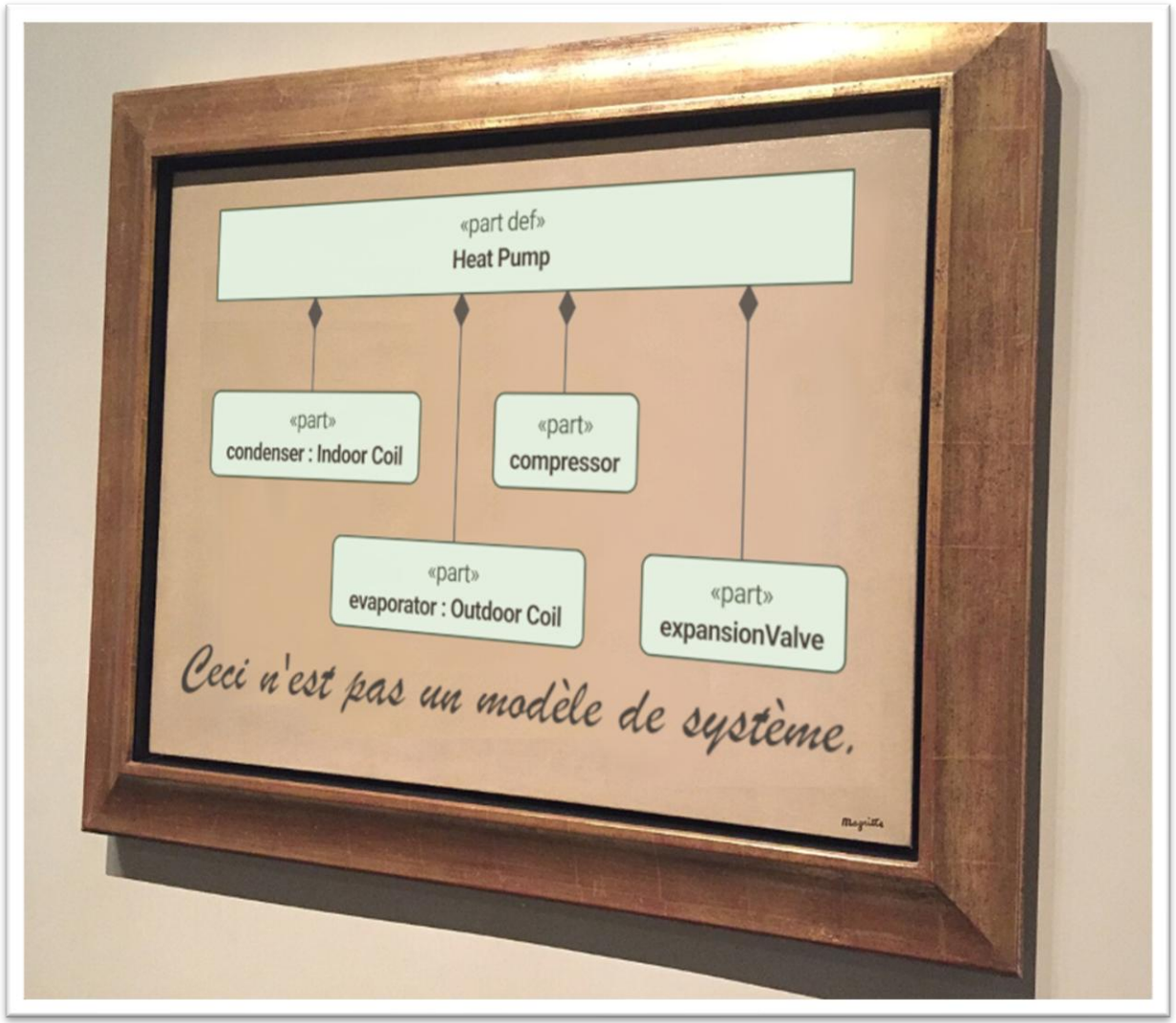
Ist das ein Modell?



Was bedeutet dieser Pfeil?
Abhängigkeit? Datenfluss? …

Ist das ein Modell?




«part def»
Heat Pump
«part»
condenser : Indoor Coil
«part»
compressor
«part»
evaporator : Outdoor Coil
«part»
expansionValve
Ceci n'est pas un modèle de système.

Modell (im Systems bzw. Software Engineering)

Ein Modell ist eine zur Beherrschung von Komplexität bewusste Abstraktion eines realen Systems, das einen Zweck* erfüllt und Stakeholder-orientierte Sichten auf seine Modelldaten bereitstellen kann.

Wichtig: Ein Modell ist nicht das System selbst, sondern ein Denk- und Arbeitswerkzeug.

*) Beispiele für Zweck: Verständnis, Entwurf, Simulation, frühe Verifikation, …

Modell (im Systems bzw. Software Engineering)

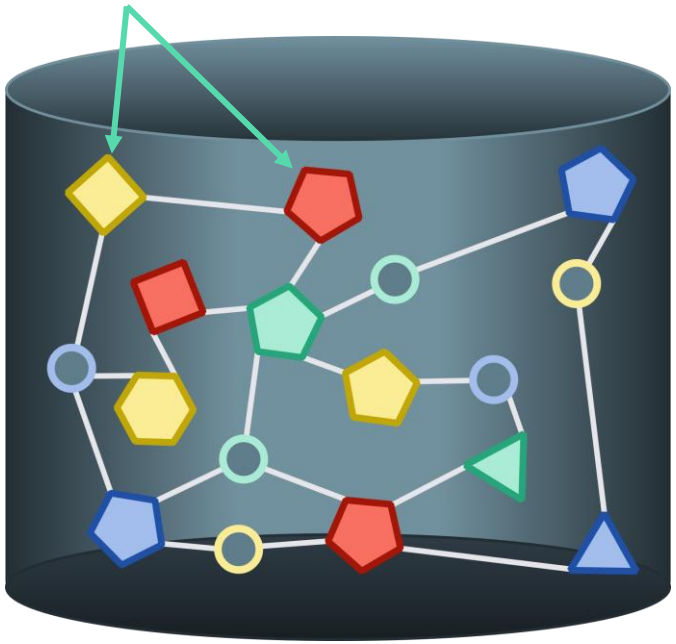
Ein Modell ist eine zur Beherrschung von Komplexität bewusste Abstraktion eines realen Systems, das einen Zweck erfüllt und **Stakeholder-orientierte Sichten** auf seine **Modelldaten** bereitstellen kann.

Sichten (engl. Views) dienen dazu, bestimmte Aspekte, Eigenschaften oder Teilbereiche eines Modells für bestimmte Zielgruppen (Stakeholder) verständlich und übersichtlich aufzubereiten.

Modelle bestehen aus, im Idealfall maschinell auswertbaren Daten bzw. Datenstrukturen.

SysML v2 Modell vs. Sichten

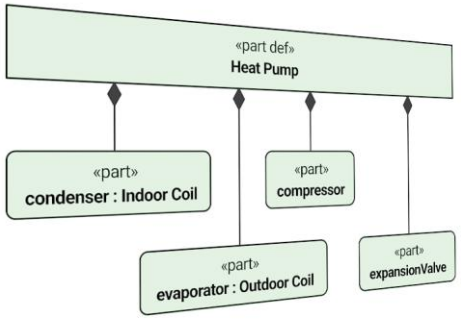
Datenstruktur der Modelldaten; Sprache: SysML v2



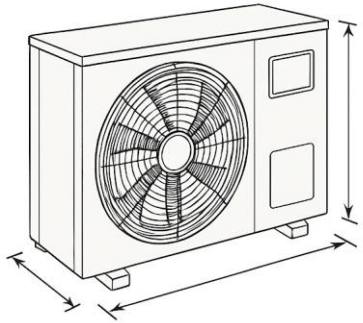
Model-Repository

```
part def 'Outdoor Coil';
part def 'Indoor Coil';
part def 'Heat Pump' {
  part compressor;
  part expansionValve;
  part evaporator : 'Outdoor Coil';
  part condenser : 'Indoor Coil';
}
```

SysML v2 Textual Notation



SysML v2 Graphical Notation



3D Wire Frame Darstellung
(SysML v2 Geometry View)

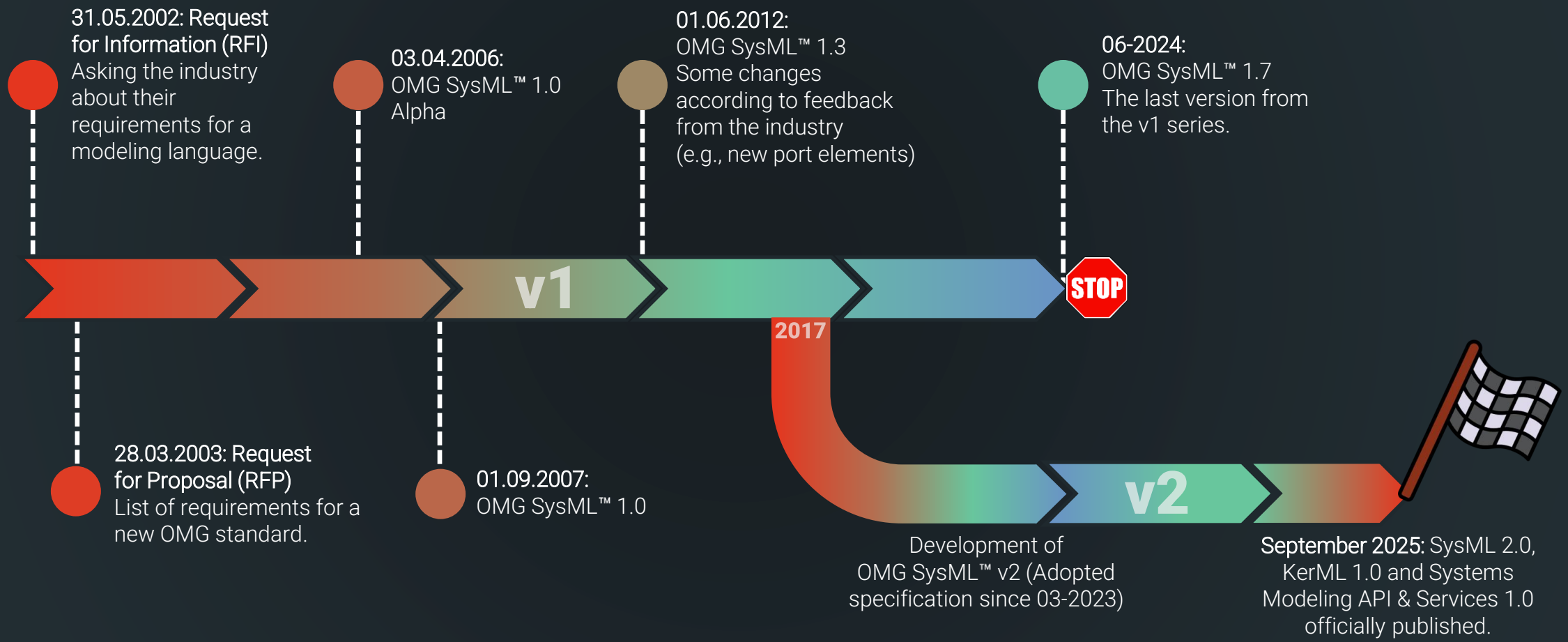
oose.

02

Der neue Stern am Himmel:
OMG Systems Modeling Language™ 2.0

AI-generiertes Bild (Night Café Studio); Prompt: „A SysML v2 lettering in the sky, bewondered by happy people.“

Entwicklung der OMG SysML™



SysML v2 Submission Team (SST)

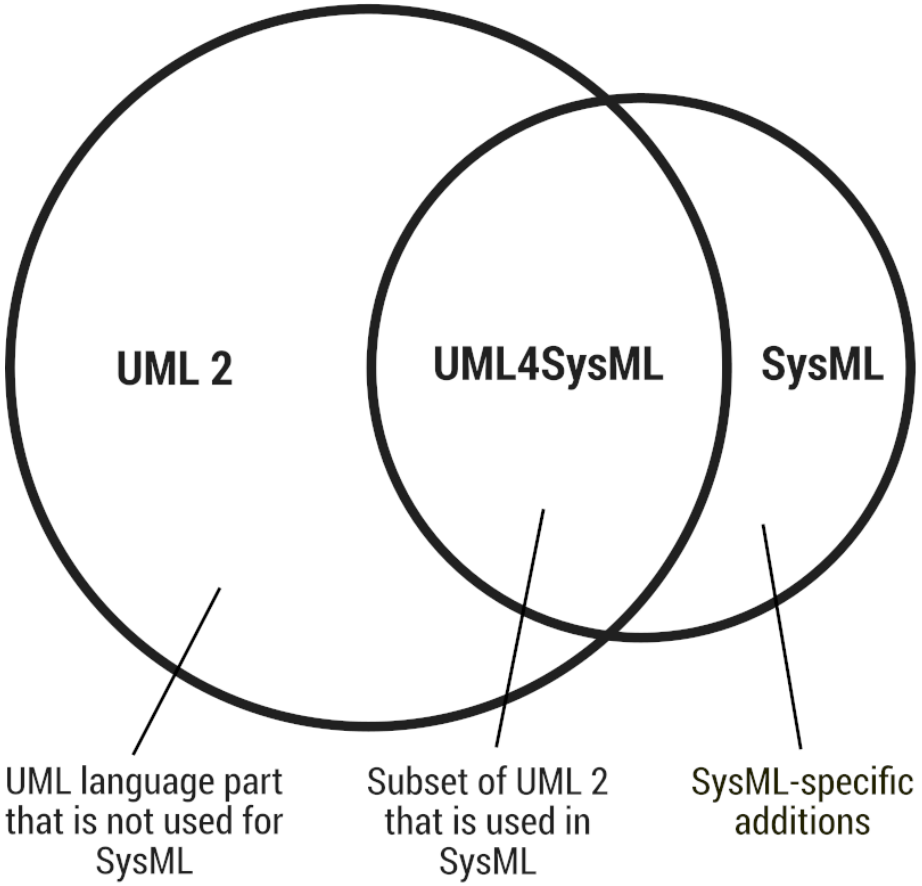
SST-Meeting in Long Beach (Los Angeles, Kalifornien), 2019. Credit: SST

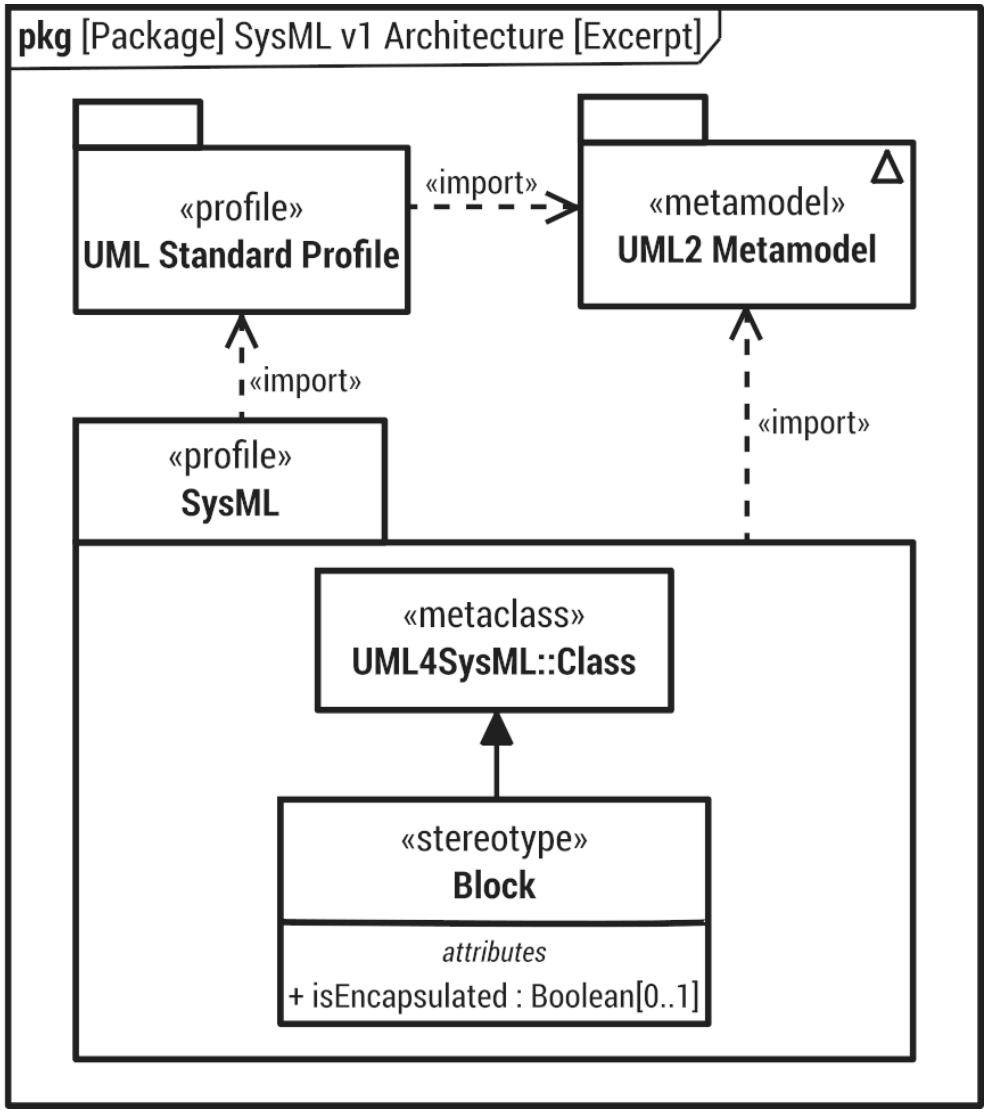
Knapp 200 Mitglieder von ~80 Organisationen (u.a. oose)
Leitung: Sanford Friedenthal und Ed Seidewitz

The Systems Modeling Language (SysML) is a **general-purpose modeling language** for **modeling systems** that is intended to facilitate a **model-based systems engineering** (MBSE) approach to **engineer systems**.

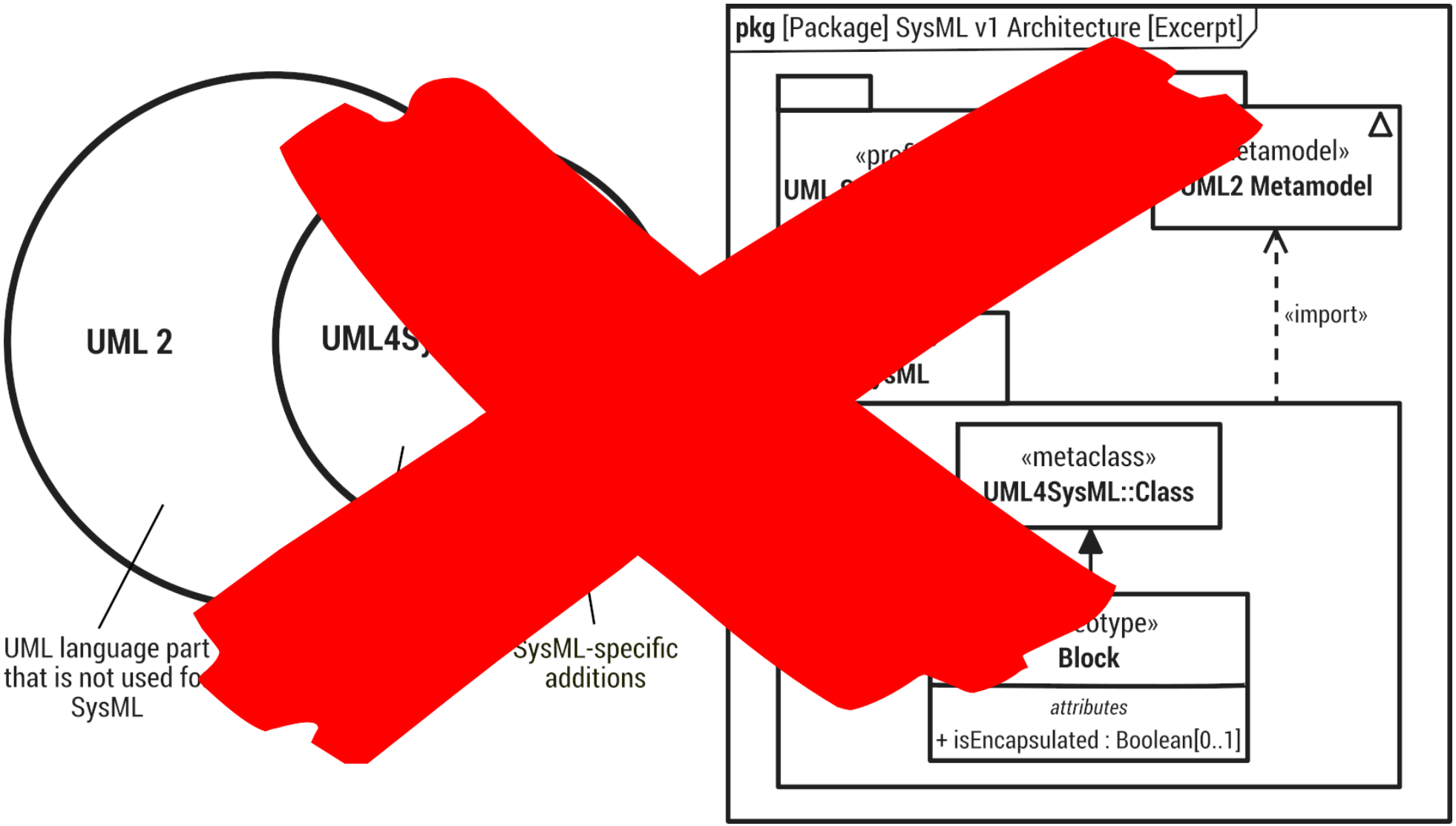
— OMG SysML™ 2.0 Specification, Part 1, page 11

Architektur des Vorgängers SysML 1.x

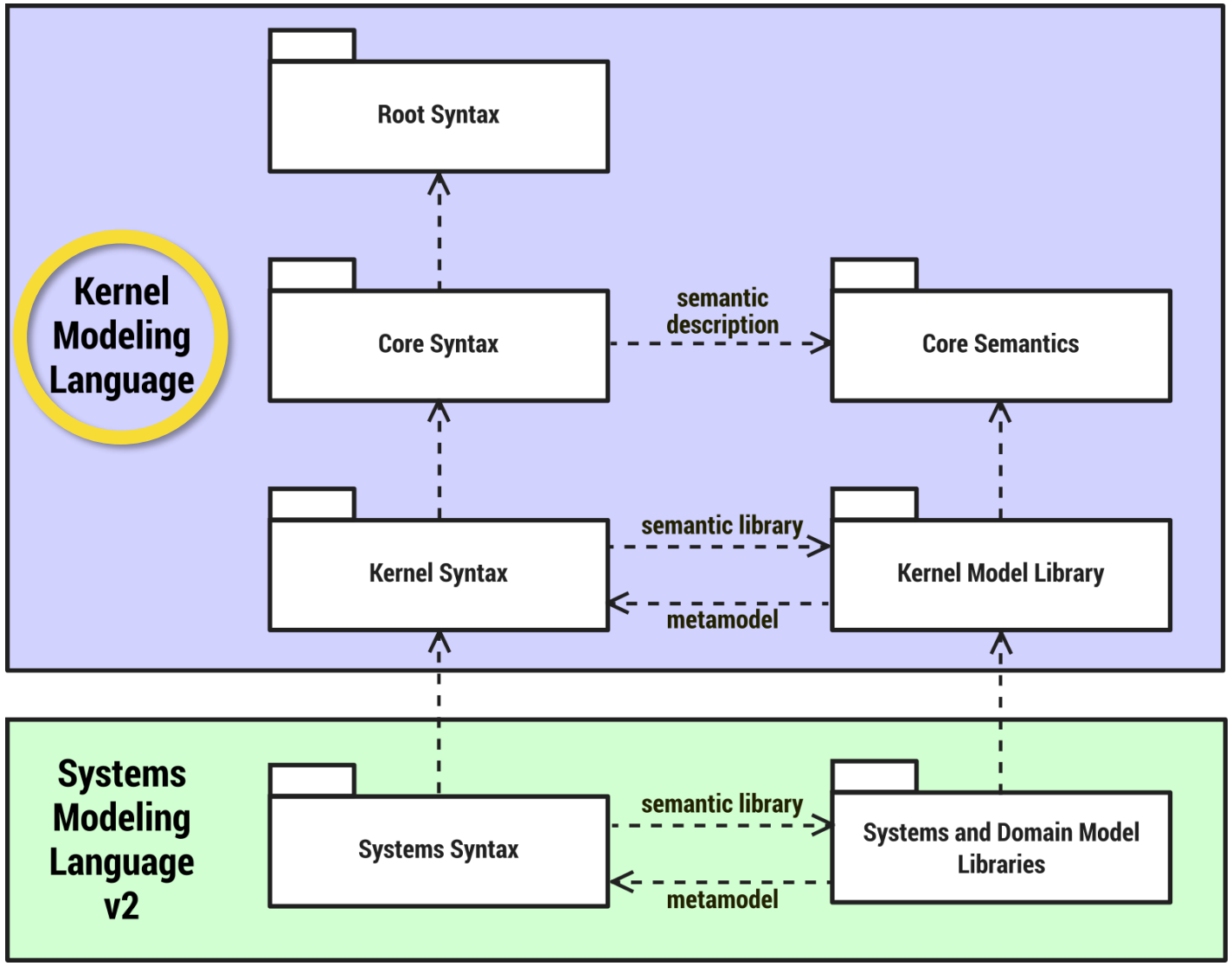




SysML v2 basiert nicht mehr auf UML!



Spracharchitektur von SysML v2



Die SysML v2 hat eine völlig überarbeitete Spracharchitektur.

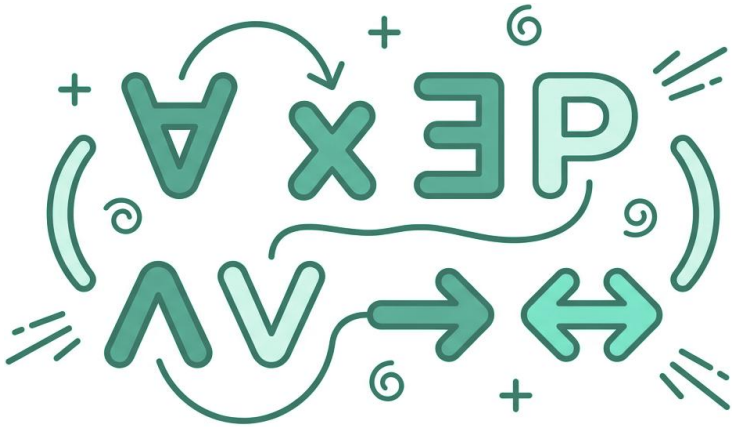
Die Basis bildet nun die **Kernel Modeling Language** (KerML).

„The **Kernel Modeling Language (KerML)** provides an application-independent syntax and semantics for creating more specific modeling languages.”

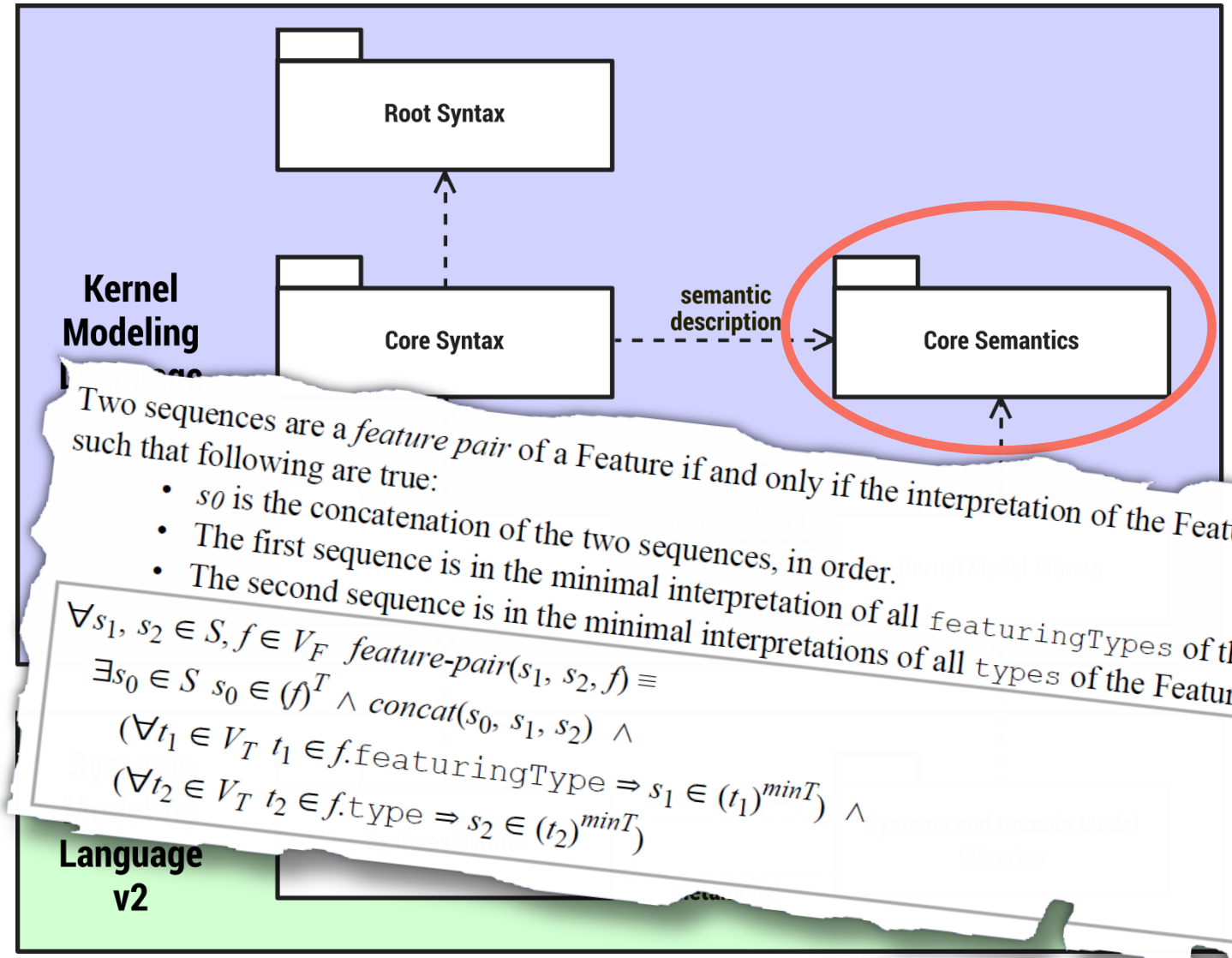
— OMG Kernel Modeling Language™ (KerML™), Version 1.0

Formale Semantik – Was bedeutet das?

- Eine formale Sprache ist eine künstliche (abstrakte) Sprache mit klar definierten und strengen Regeln.
- In KerML ist die Sprachsemantik definiert mit Hilfe von Prädikatenlogik erster Stufe:
  - Ein formales logisches System, das Aussagen über Objekte einer Domäne mithilfe von **Quantoren**, **Operatoren** und **Prädikaten** ausdrückt.
    - Quantoren dienen dazu, den Gültigkeitsbereich einer Aussage anzugeben. Die beiden gebräuchlichsten Quantoren sind der Existenzquantor (∃ bzw. ∃!) und der Allquantor (∀).
    - Operatoren sind die Konjunktion (∧), Disjunktion (∨), Negation (¬) und Implikation (⇒).
    - Prädikate sind Aussagen, die wahr oder falsch sein können.



Formale Semantik



Two sequences are a *feature pair* of a Feature if and only if the interpretation of the Feature includes a sequence $s0$ such that following are true:

- $s0$ is the concatenation of the two sequences, in order.
- The first sequence is in the minimal interpretation of all `featuringTypes` of the Feature.
- The second sequence is in the minimal interpretations of all `types` of the Feature.

$$\forall s_1, s_2 \in S, f \in V_F \ \ \textit{feature-pair}(s_1, s_2, f) \equiv$$
$$\exists s_0 \in S \ \ s_0 \in (f)^T \wedge \textit{concat}(s_0, s_1, s_2) \ \wedge$$
$$(\forall t_1 \in V_T \ \ t_1 \in f.\texttt{featuringType} \Rightarrow s_1 \in (t_1)^{minT}) \ \wedge$$
$$(\forall t_2 \in V_T \ \ t_2 \in f.\texttt{type} \Rightarrow s_2 \in (t_2)^{minT})$$

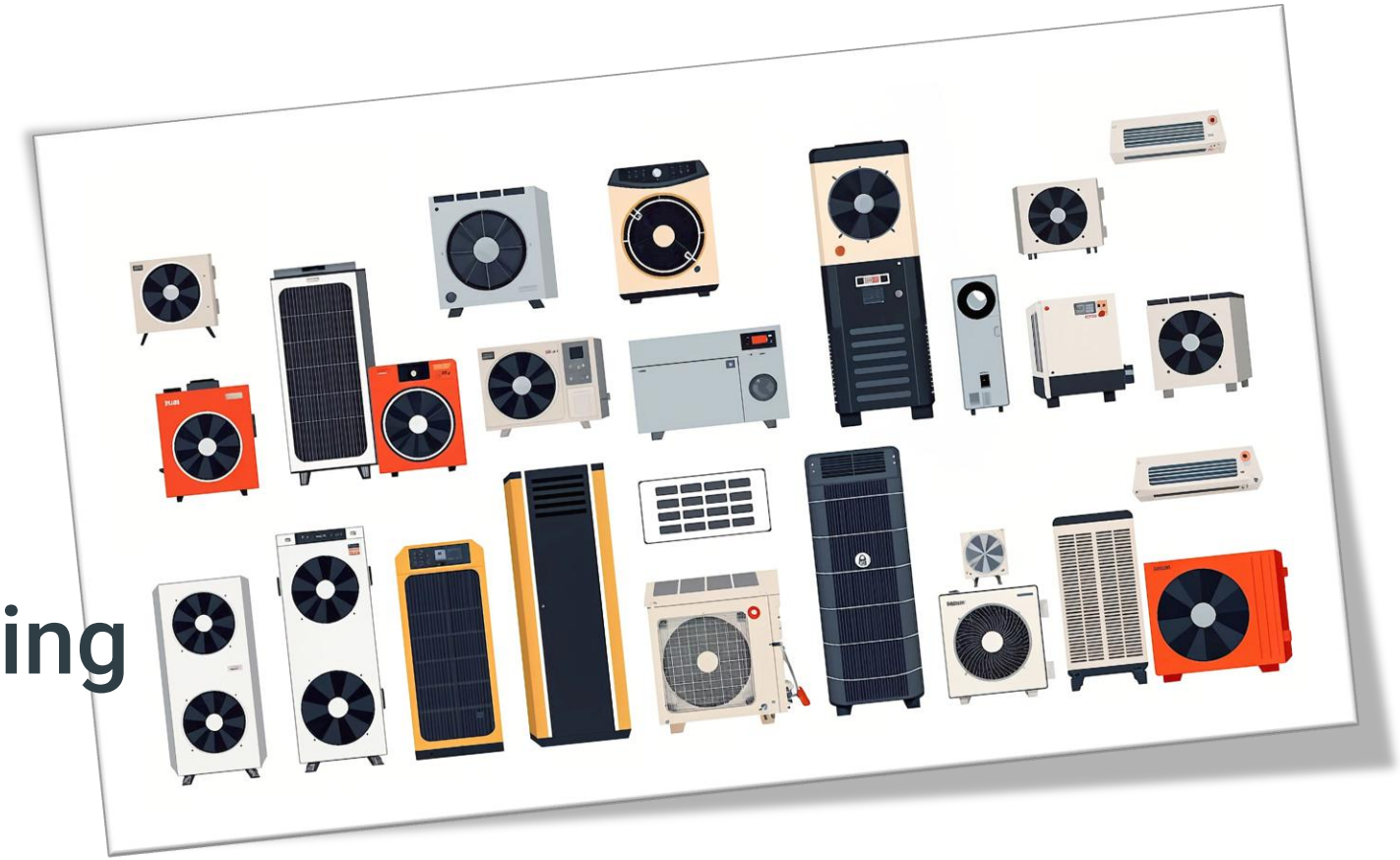
Ein Beispiel aus der KerML Spezifikation.

Und was bringt mir das?

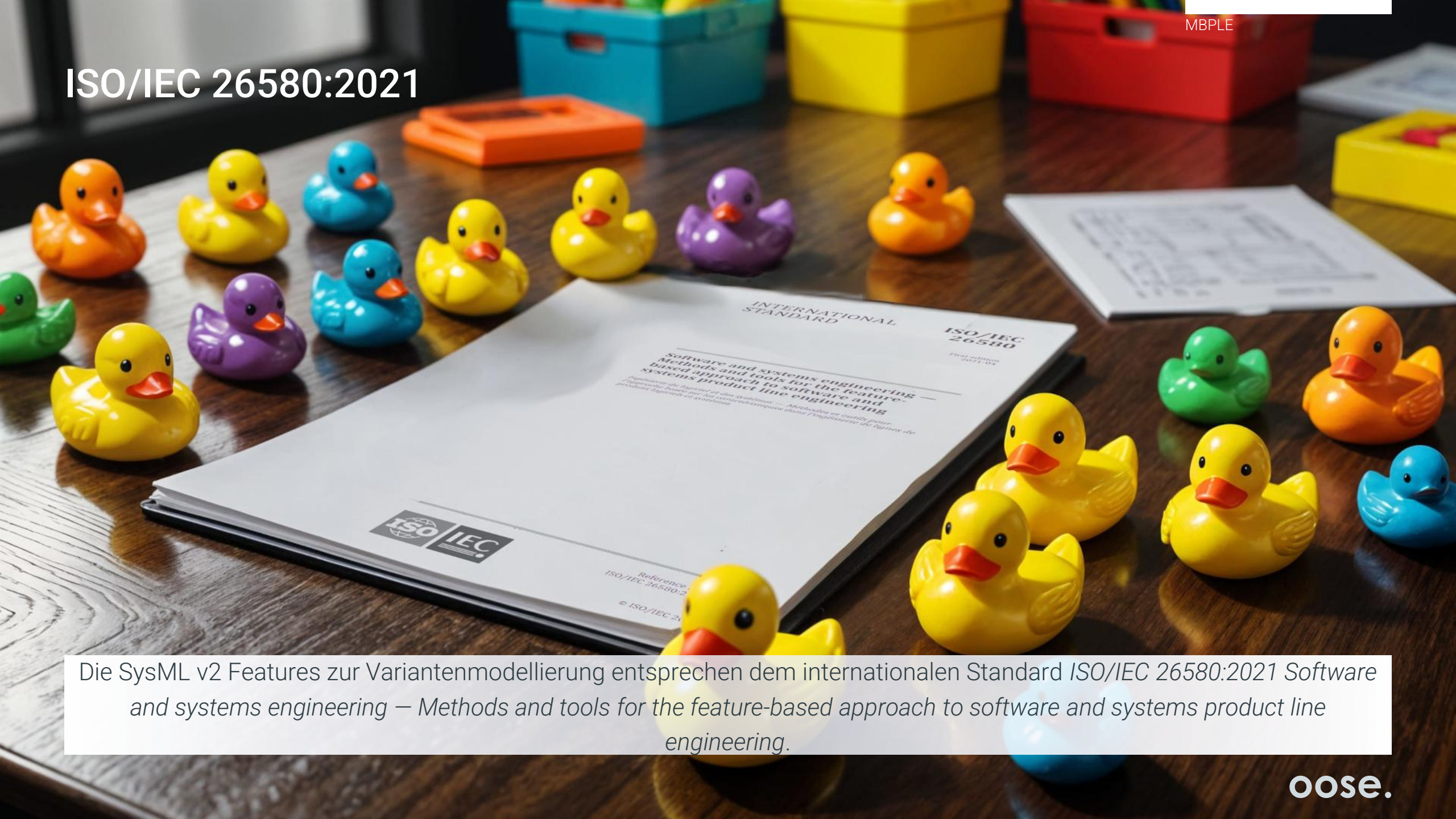
- Formale Regeln sorgen dafür, dass wohlgeformte Modelle entsprechend den Regeln der Sprache erstellt werden.
  - Toolhersteller können die Regeln in Form von Validierungen in ihren Werkzeugen implementieren.
- Modellierungswerkzeuge verschiedener Hersteller behandeln die Modelle identisch und machen Modelldaten austauschbar.
- Ausführungssemantik: Keine Interpretationsspielräume mehr bei Modellsimulationen.
  - Unterschiedliche Simulationswerkzeuge kommen zu denselben Ergebnissen.

03

Model-Based Product Line Engineering (PLE)

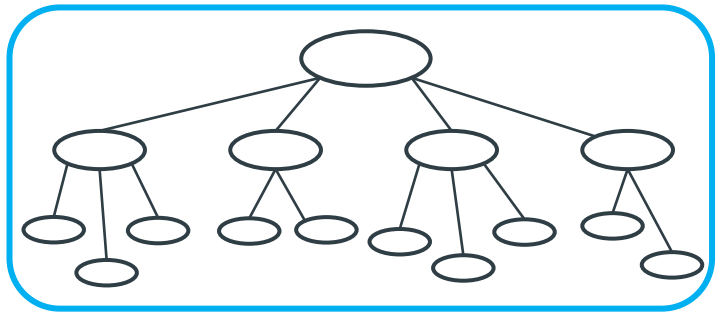


ISO/IEC 26580:2021

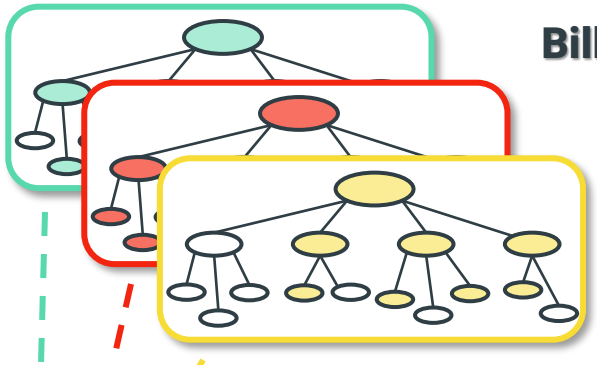


Die SysML v2 Features zur Variantenmodellierung entsprechen dem internationalen Standard *ISO/IEC 26580:2021 Software and systems engineering — Methods and tools for the feature-based approach to software and systems product line engineering*.

Feature-Based PLE entsprechend ISO/IEC 26580:2021

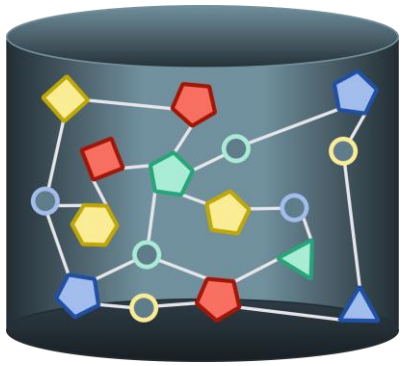


Feature Catalogue



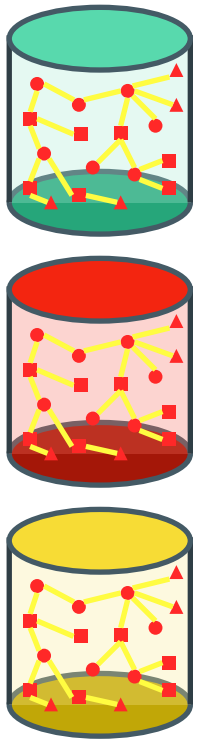
Bill of Features (Portfolio)

Selected feature sets from the Feature Catalogue for each system in a system family portfolio.



Shared Assets Superset („150% Model“)

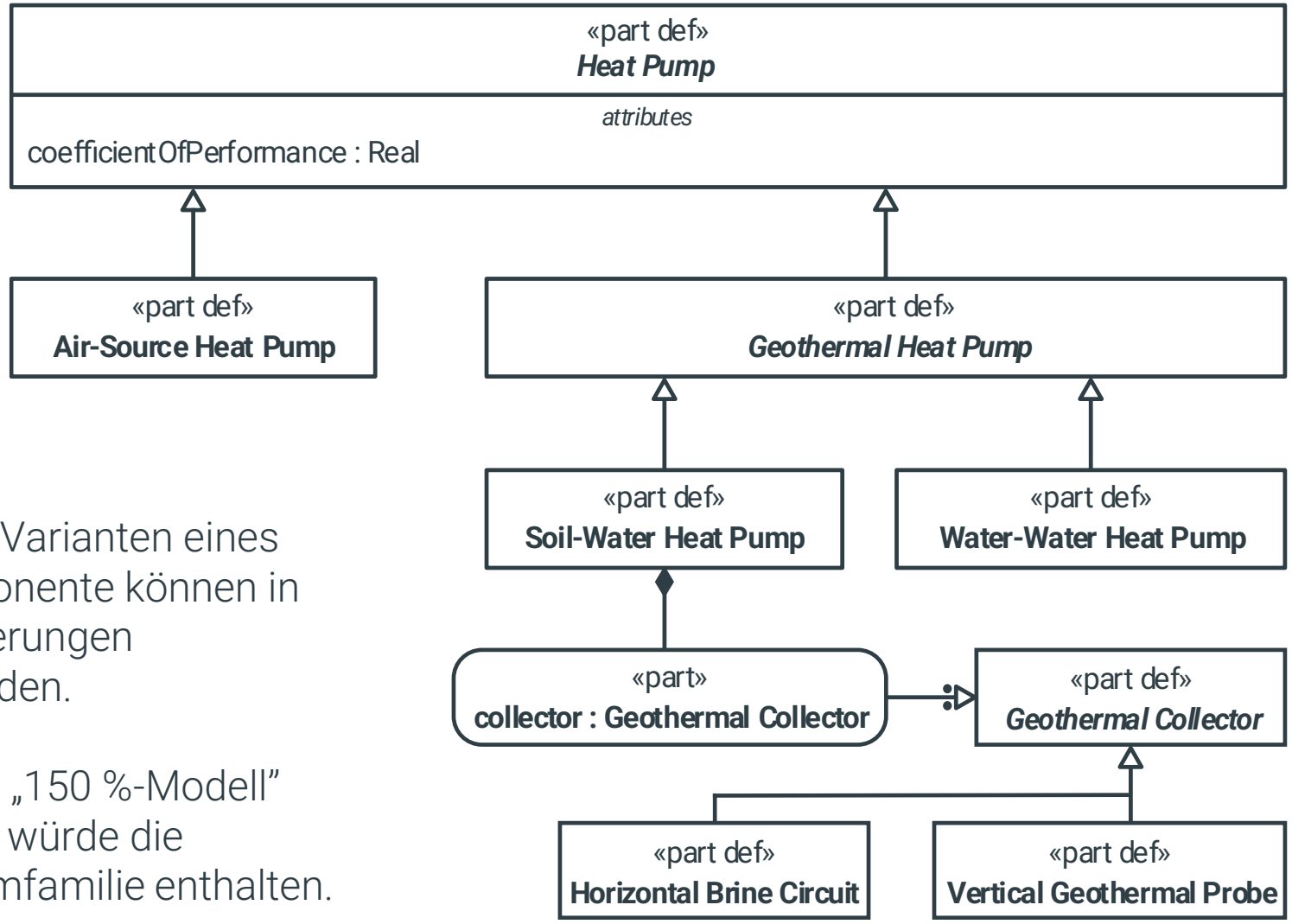
PLE Factory Configurator



Product Asset Instances

Extracted models that contain only the shared asset content that is intended for the respective system of the system family.

Specialization (Subclassification)

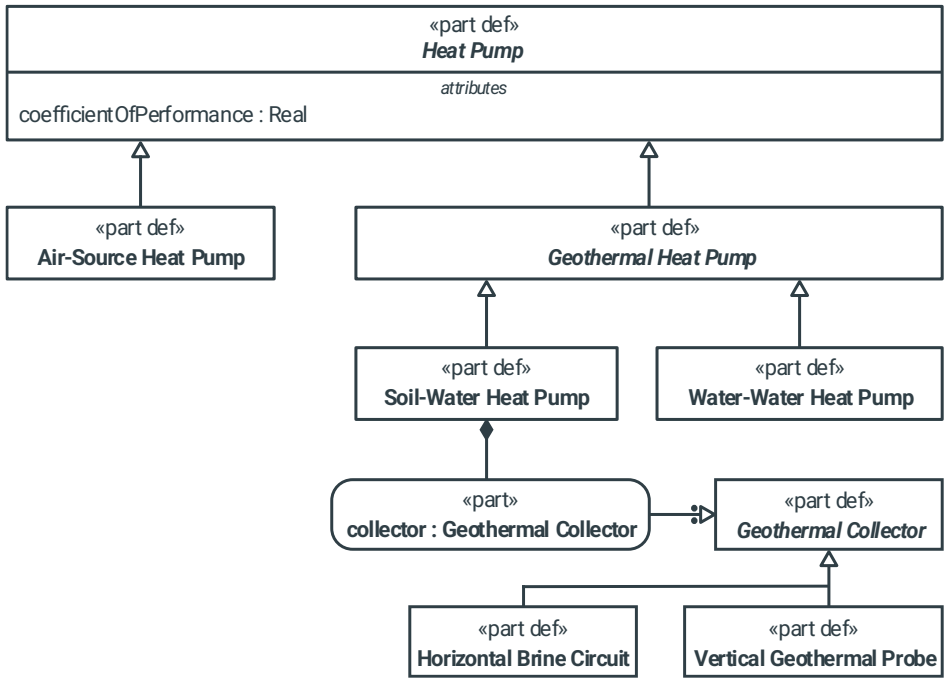


Verschiedene Ausprägungen und Varianten eines Systems oder einer Systemkomponente können in SysML v2 mit Hilfe von Spezialisierungen (Subclassification) modelliert werden.

Das resultierende Modell wäre ein „150 %-Modell” (Shared Assets Superset), d. h. es würde die Modelldaten der gesamten Systemfamilie enthalten.

Shared Assets Superset

```
01  package 'Heat Pump Product Family' {
02    abstract part def 'Geothermal Collector';
03    part def 'Horizontal Brine Circuit' :>
        'Geothermal Collector';
04    part def 'Vertical Geothermal Probe' :>
        'Geothermal Collector';
05
06    abstract part def 'Heat Pump' {
07      attribute coefficientOfPerformance :
          ScalarValues::Real;
08    }
09    part def 'Air-Source Heat Pump' :> 'Heat Pump';
10    abstract part def 'Geothermal Heat Pump' :> 'Heat Pump';
11    part def 'Soil-Water Heat Pump' :>
        'Geothermal Heat Pump' {
12      part collector : 'Geothermal Collector';
13    }
14    part def 'Water-Water Heat Pump' :>
        'Geothermal Heat Pump';
15  }
```



Variation Definitions

Modellelemente können als **Variationsmöglichkeiten** gekennzeichnet werden, indem das keyword **`variation`** vor dem kind keyword platziert wird.

Die **Variante** (keyword: **`variant`**) repräsentiert dann eine zulässige Design-Auswahl innerhalb einer Variation.

```
01  abstract part def 'Heat Pump‘ {
02    // Stuff that is the same for all heat pumps
03    // in a product family
04    attribute coefficientOfPerformance : ScalarValues::Real;
05  }
06
07  part def Compressor;
08  part reciprocatingCompressor : Compressor;
09  part scrollCompressor : Compressor;
10
11  // Variability model:
12  variation part def CompressorChoices :> Compressor {
13    variant reciprocatingCompressor;
14    variant scrollCompressor;
15  }
```

Variation Usages

Zusicherungen (**`constraint`**) können verwendet werden, um Regeln zu definieren und unzulässige Konfigurationen auszuschließen.

xor = Exclusive-Or

```
01  abstract part heatPumpProductFamily : 'Heat Pump' {
02    // A variation definition (see previous slide) can be used
03    // like a ordinary definition. The valid values of the
04    // usage are restricted to the allowed variants.
05    part compressor : CompressorChoices [1];
06
07    // Note: Definitions of collectors not shown here
08    variation part collector : 'Geothermal Collector' [1] {
09      variant 'Horizontal Brine Circuit';
10      variant 'Vertical Geothermal Probe';
11    }
12
13    assert constraint {
14      (compressor == compressor::reciprocatingCompressor and
15      collector == collector::'Horizontal Brine Circuit') xor
16      (compressor == compressor::scrollCompressor and
17      collector == collector::'Vertical Geothermal Probe')
18    }
19  }
```

Variation Configuration

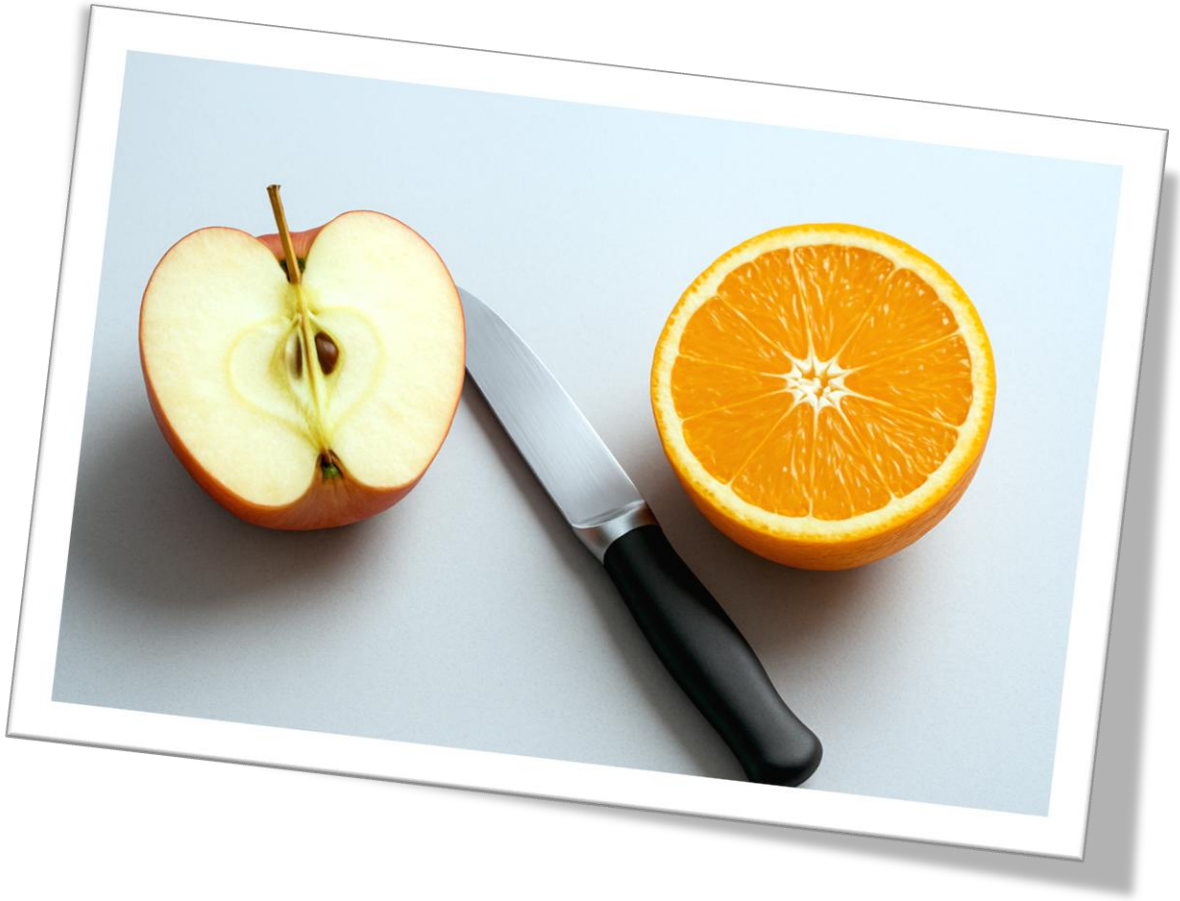
```
01  part heatPumpA :> heatPumpProductFamily {
02    part redefines compressor =
        compressor::reciprocatingCompressor;
03    part redefines collector =
        collector::'Horizontal Brine Circuit';
04  }
05
06  part heatPumpB :> heatPumpProductFamily {
07    part redefines compressor = compressor::scrollCompressor;
08    part redefines collector =
        collector::'Vertical Geothermal Probe';
09  }
```

Eine konkrete Produktkonfiguration kann wie folgt definiert werden:

1. Erstelle eine Spezialisierung des Produktfamilien-Elements.
2. Wähle für jeden Variationspunkt die gewünschte Variante aus (Redefinition).

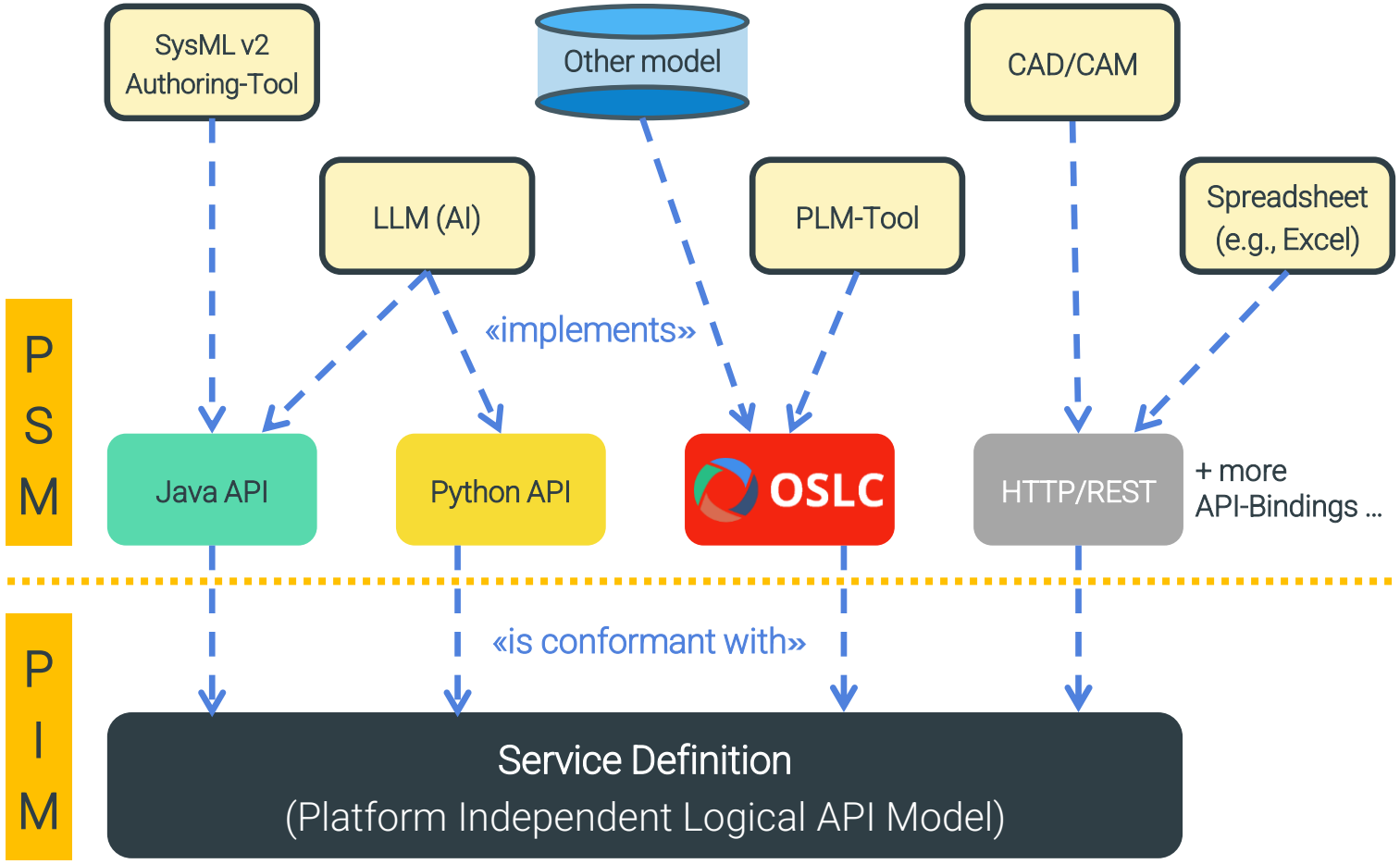
04

Systems Modeling API & Services



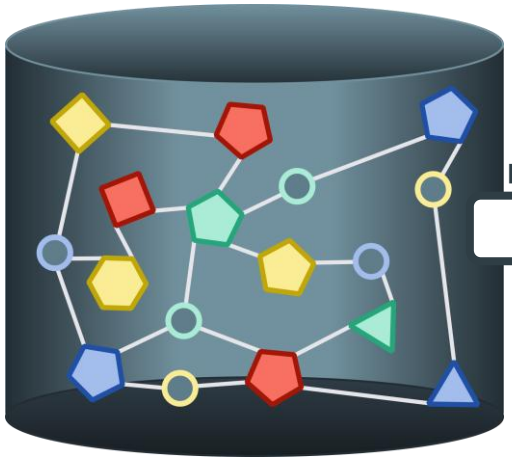
Systems Modeling API & Services

Teil des Standards ist auch eine plattformunabhängige Service-Spezifikation (PIM) zum Abfragen und Bearbeiten der Daten eines SysML v2-Modells sowie zwei plattformspezifische API-Bindungen der PIM unter Verwendung bestimmter Technologien (HTTP/REST und OSLC).



oose.

Use Case: Kollaborativ mit Stakeholdern arbeiten



Model-Repository

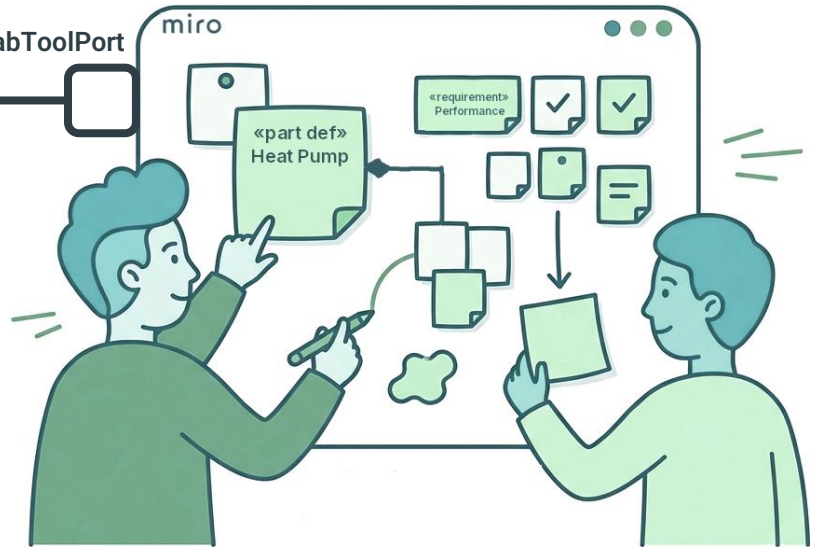
p1 : SysMLv2APIPort

p2 : ~SysMLv2APIPort

«part»
SysML v2 API to Collaboration Platform Protocol Converter

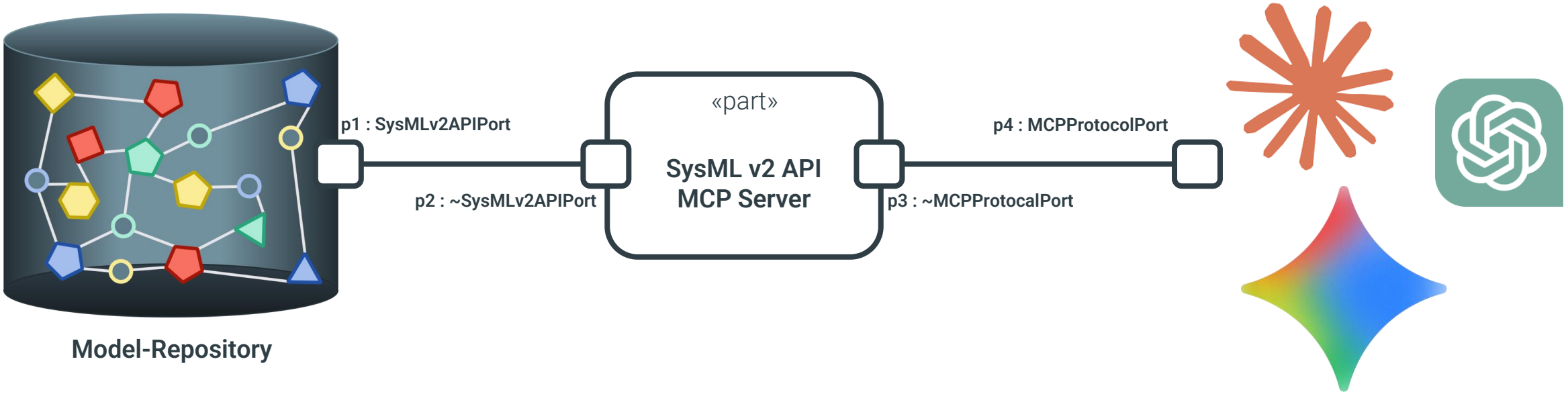
p3 : ~CollabToolPort

p4 : CollabToolPort



Alle Stakeholder – auch die, die keine SysML-Experten sind – können gemeinschaftlich am Systemmodell arbeiten.

Use Case: MCP-Server



Digital Engineering: Das SysML-Modell als integrierendes Element

Requirements Management
IBM DOORS · Polarion · reqSuite® RM
Anforderungen

Simulation & Analyse
Matlab · Simulink · Modelica
Simulation

CAD & 3D-Modellierung
CATIA · Siemens NX · Ansys
CAD/CAE/CAM

PLM & Varianten
Teamcenter · Windchill
PLM

MBSE-Modellierung
CATIA Magic SoSA · Rhapsody · Sparx EA
SysML v2

Qualitätssicherung & Test
ALM · TestBench
Verifikation

Dokumentation
Confluence · Nuclino · ADOC · Office
Docs

AI-Support
Claude · Gemini · GPT · Llama · Mistral
LLM

SysML v2 – Enabler für *echtes* MBSE

Formale Semantik (KerML)

Offen für Erweiterungen (Libraries & Metadata)

„Systems engineers routinely compose task-specific virtual models using **ontologically linked, digital twin-based model-assets**. These connected models are **updated in real-time** providing a **virtual reality-based, immersive design and exploration space**."

—INCOSE SE Vision 2035 (2022)

Digital Twin/Digital Thread

Stakeholder-spezifische Sichten

Varianten und MBPLE

Standardisierte API

Besuche uns auch im Ausstellungsbereich!

Hat Dir der Vortrag gefallen?
Dann freue ich mich über
Deine Bewertung in der
REConf-App.

oose.de