Kollaboration im Systems Engineering mit ReqIF

REConf 2022

Oskar von Dungern, Dr.-Ing., enso managers GmbH

Die „Story“

1. Bedarf
2. prostep IVIP „Stakeholder Request Clarification“ … auch für Modelle
3. Etwas Theorie: Modellintegration
4. Beispiel: Anforderungen und Systemmodell zusammen führen
5. SpecIF Open Source Program

1 Bedarf

Zusammenarbeit bei der Systementwicklung in der Lieferkette

- Automotive: „ReqIF für Systemmodelle mit verknüpften Anforderungen“
- Luft- und Raumfahrt
- Versorgungsunternehmen: Planung und Wartung von Parks für erneuerbare Energien

Modellaustausch zwischen Systemmodellierungswerkzeugen

- SysML: EA ↔ Cameo ↔ Rhapsody und andere
- Andere Notationen

Integration von Modellen aus verschiedenen Notationen und Werkzeugen

- Navigieren, Suchen und Prüfen von Teilmodellen in einem gemeinsamen Kontext

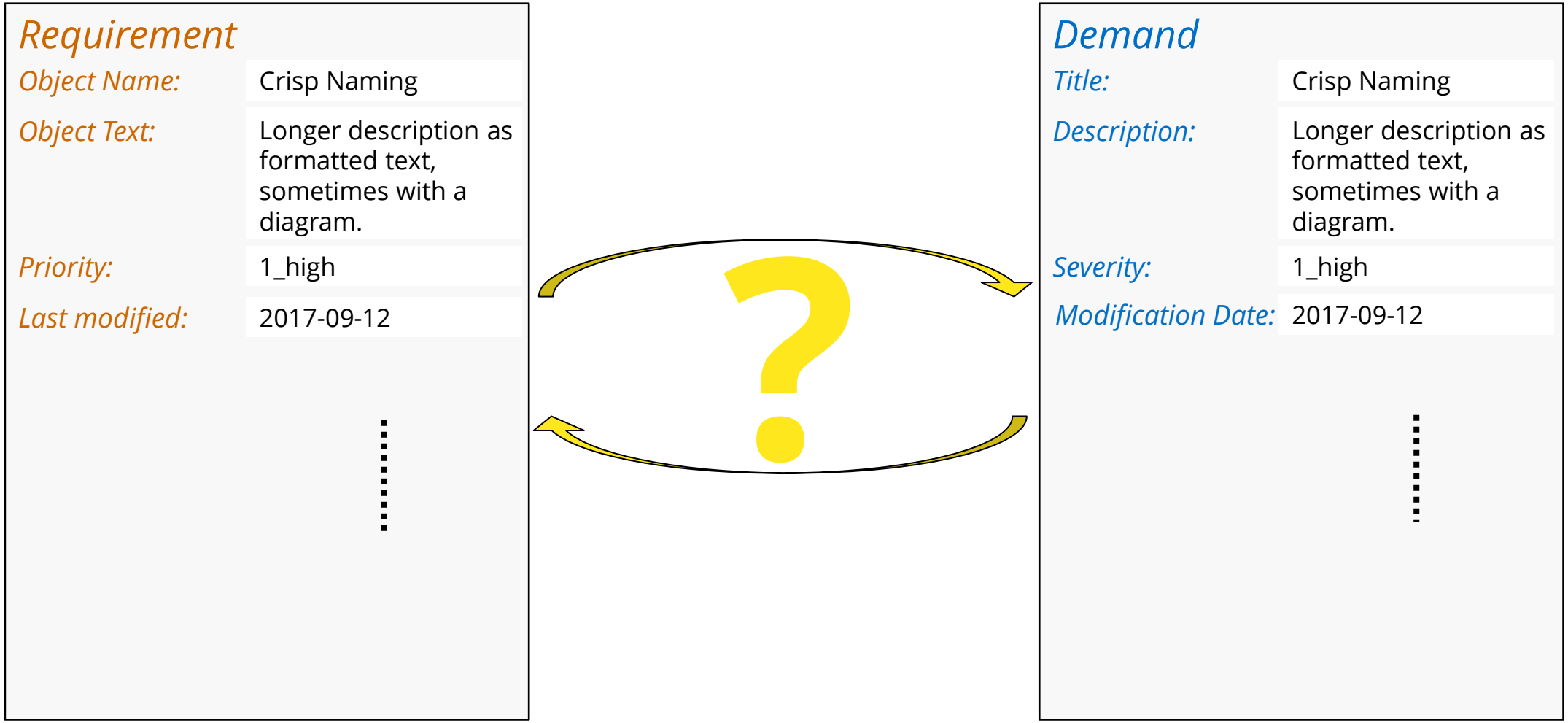
→ SpecIF wurde für diese Anforderungen entwickelt und wird bereits eingesetzt

1 prostep IVIP „Stakeholder Request Clarification“

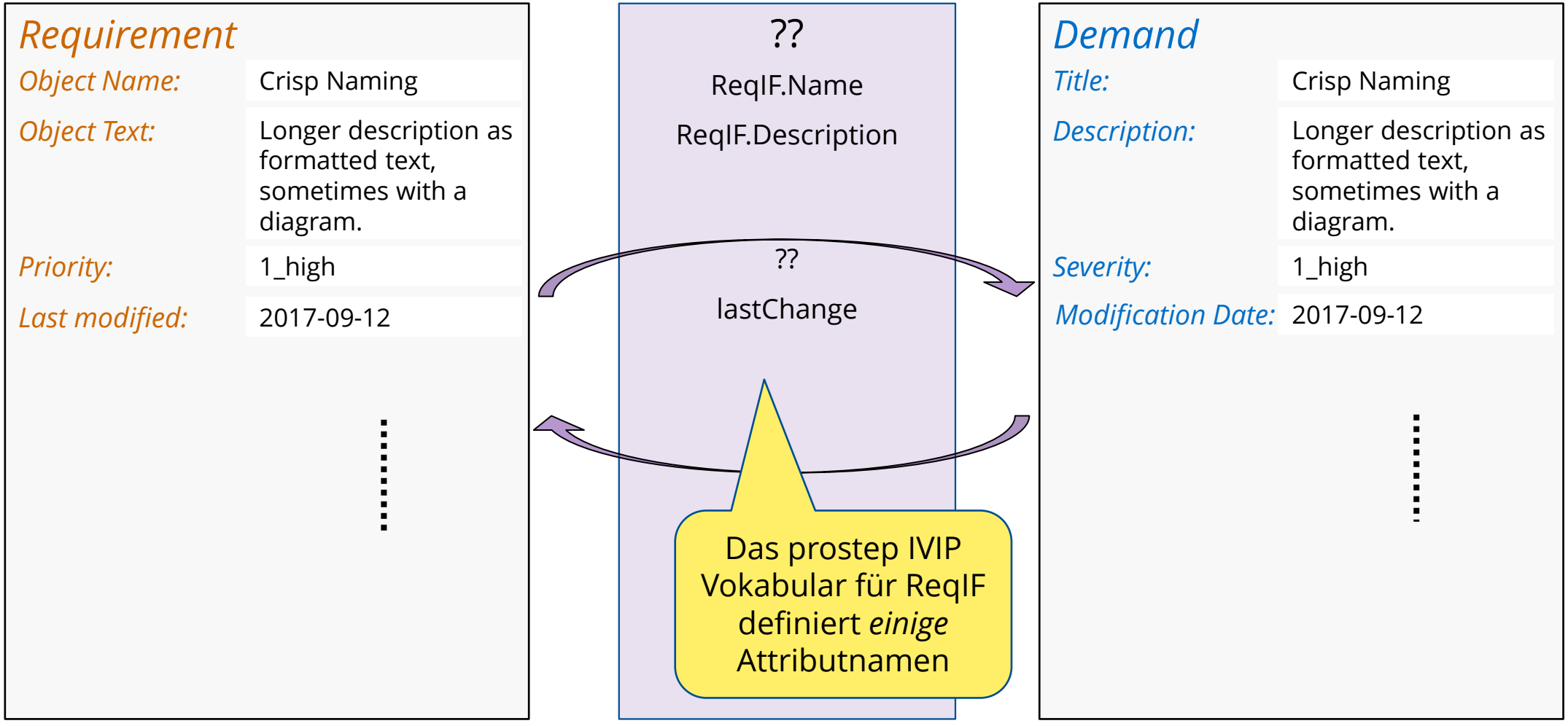
- Anforderungsdokumente zwischen Kunde und Lieferant abstimmen
- Standard-Format OMG ReqIF (XML)
- Definierte Attributnamen nach prostep IVIP Recommendation
- Kunde und Lieferant nutzen jeweils 2 Attribute „Zustand“ und „Kommentar“ zur Abstimmung

- … oder auch Modellentwicklung durch Ingenieur-Dienstleister wie im CAD

Systeme benennen Informationselemente unterschiedlich

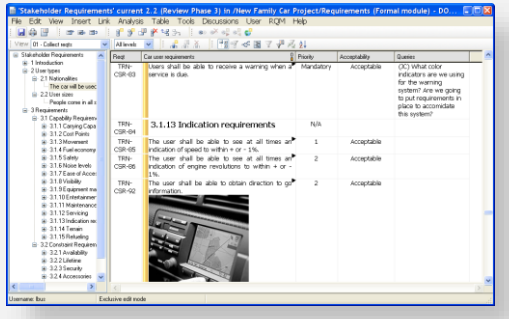


Erst ein Vokabular gibt Informationselementen eine Bedeutung und ermöglicht einen Informationsaustausch ohne Erläuterung



Use-Case:
Anforderungsdokument zwischen Kunde und Lieferant abstimmen

Kunde (OEM)



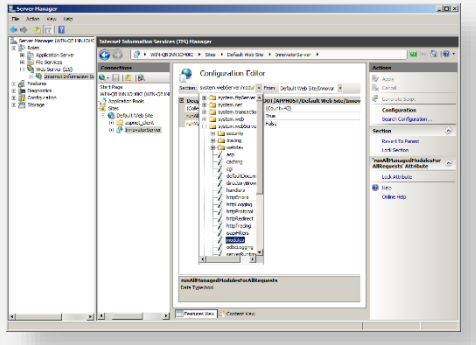
- Requirement-Management

ReqIF

ReqIF

ReqIF mit
- 1 OBJECT-TYPE
- prostep IVIP Vokabular

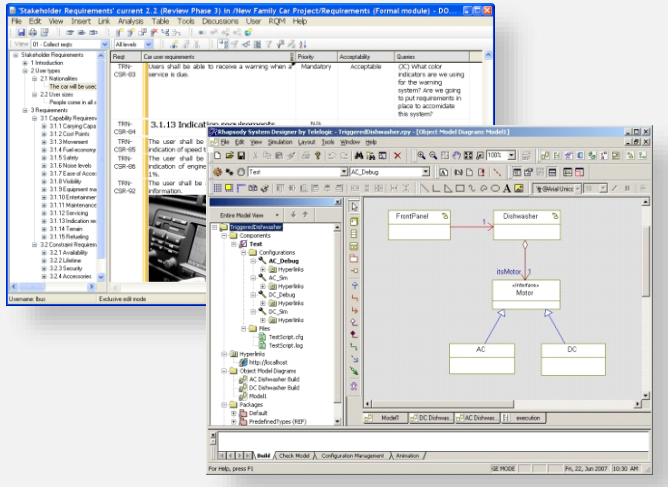
Lieferant



- PLM/ALM

Use-Case:
Modellbasierte Spezifikation zwischen Kunde und Lieferant abstimmen

Kunde (OEM)



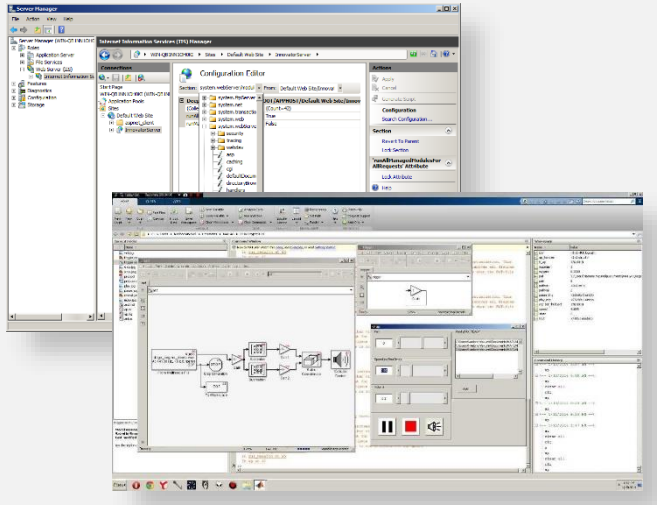
- Requirement-Management
- System-Modelling

ReqIF

ReqIF

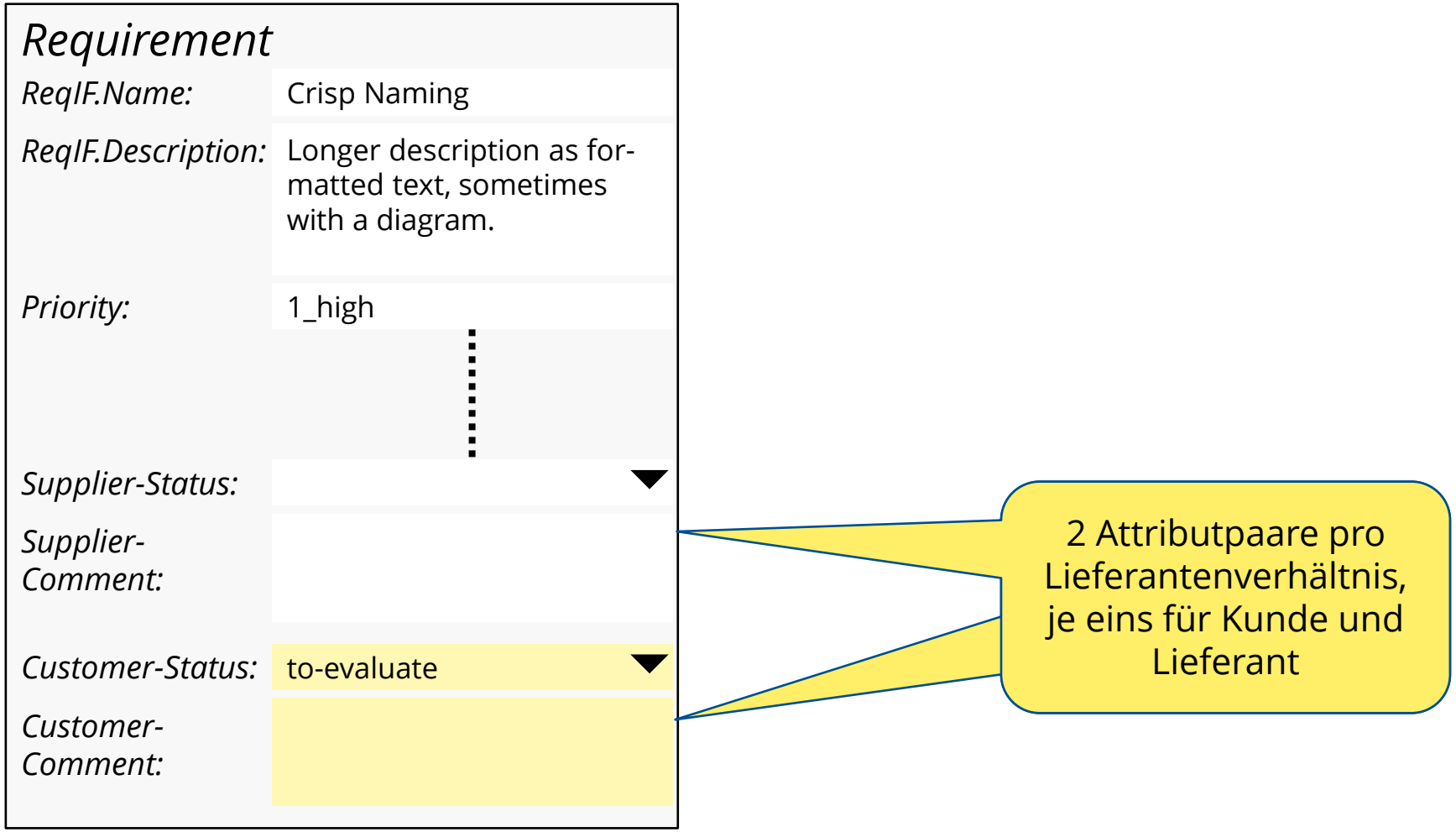
ReqIF mit
- ~7 OBJECT-TYPES
- ~10 RELATION-TYPES
- SpecIF Vocabulary

Lieferant

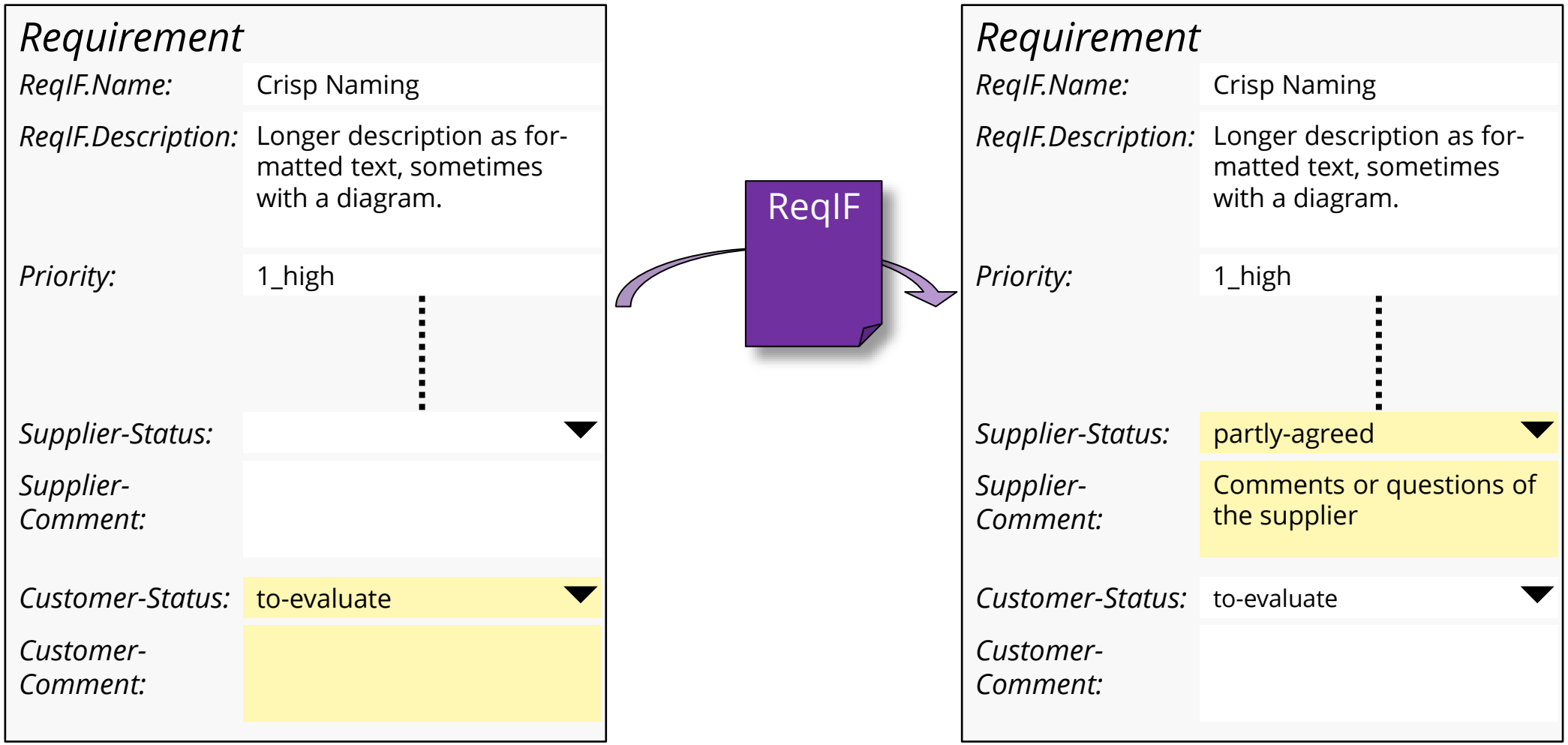


- PLM/ALM
- Simulation

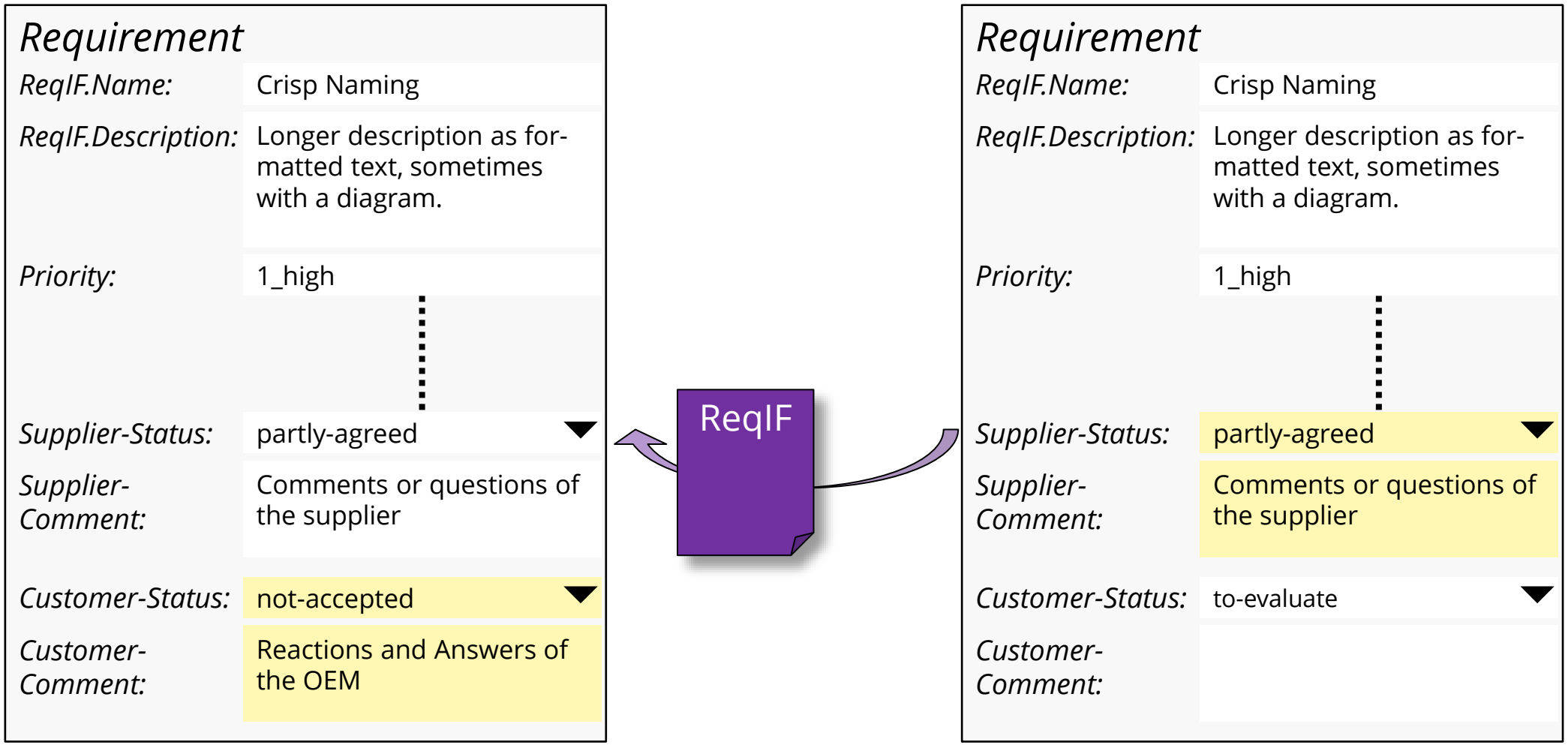
Use-Case: Anforderungen abstimmen mit dem SRC-Prozess



Use-Case: Anforderungen abstimmen mit dem SRC-Prozess



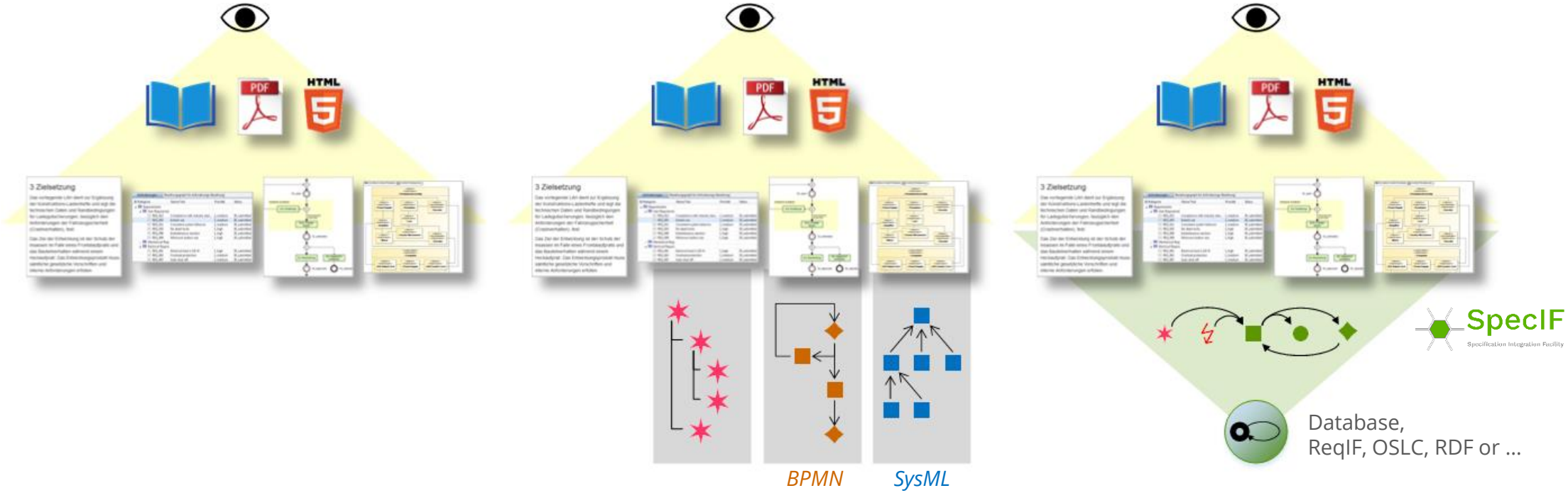
Use-Case: Anforderungen abstimmen mit dem SRC-Prozess



2 Etwas Theorie: Fünf Prinzipien der Modellintegration

1. Sicht und Modell trennen
2. Modellelement-Typen abstrahieren
3. Vokabular benutzen
4. Identische Modellelemente auf mehreren Sichten identifizieren
5. Modellelemente mit ihren Relationen zu einem semantischen Netz verknüpfen

Das Auge sieht dasselbe – dahinter wird‘s interessant



Das „Sichtbare“ erzeugen

- Texte erstellen und Diagramme zeichnen
- Erfordert Gedankenkraft und Disziplin

Partielles Modellieren

- Texte erstellen und Modellieren pro Notation
- Werkzeugunterstützung innerhalb der Notationen

Modellintegration

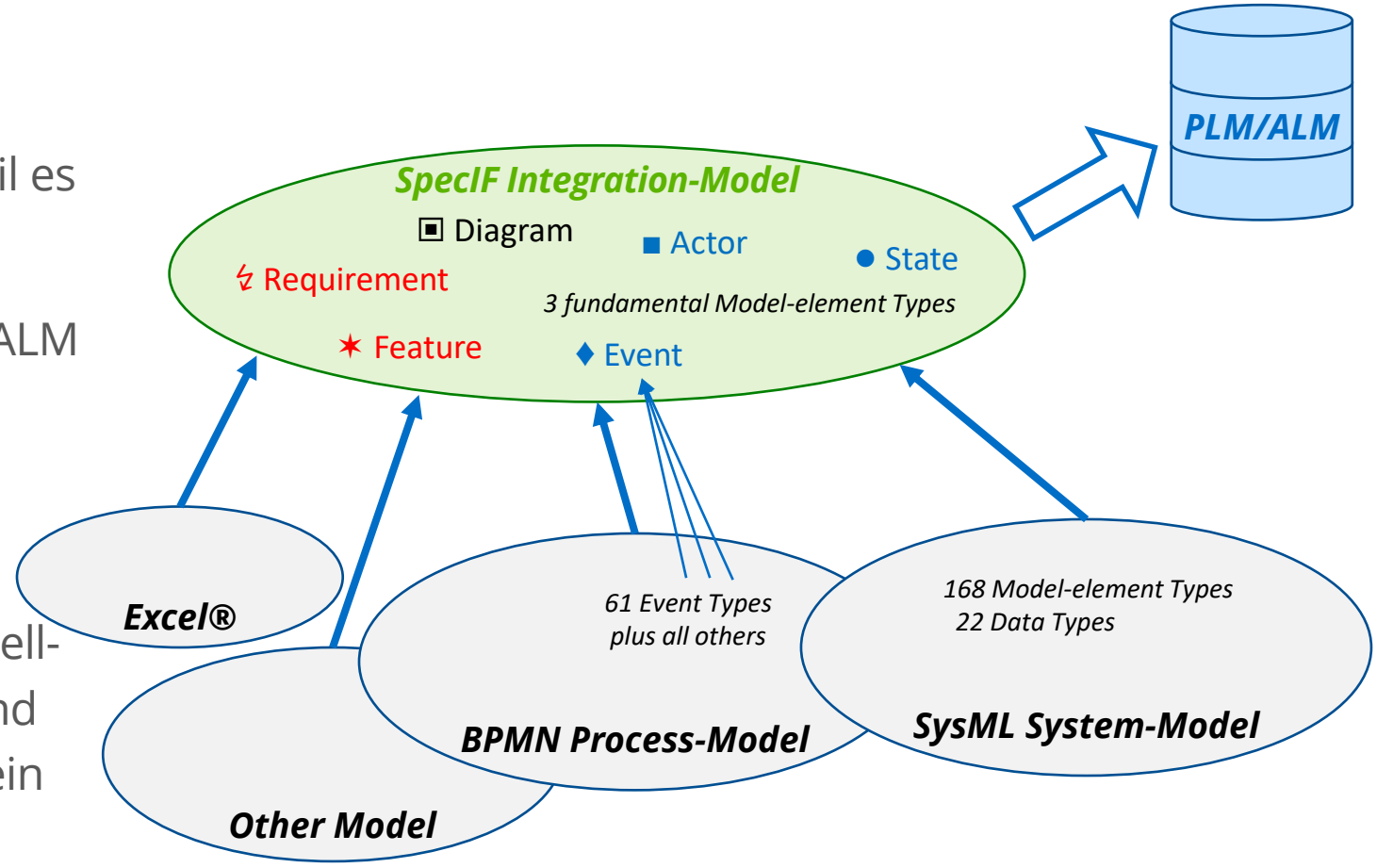
- Texte erstellen und Modellieren pro Notation
- Elemente aller Sichten sind über ein semantisches Netz verknüpft

Eine nützliche Abstraktionsebene finden …

- Modell-Integration: Original Modell-element-Typen sind ungeeignet, weil es viel zu viele gibt.
- Konfigurations-Management: PLM/ALM soll nur „wenige“ Artefakt-/CI-Typen unterscheiden

→ Abbildung auf 3 fundamentale Modell-element-Typen ■ Akteur, ● Zustand und ♦ Ereignis, die allen Notationen gemein sind.

(see Fundamental Modelling Concepts by S.Wendt)



Vokabular benutzen

- Begriffen Bedeutung geben
- Einigung zu Begriffen erzielen
- Bedeutung wird über Begriffe transportiert
- Begriffe können übersetzt werden:
  - In Nationalsprachen
  - In Spezialterminologie eines Arbeitsfelds

→ Bitte nutzen und beitragen zum
SpecIF System Engineering Vocabulary

Resource Class Names
- FMC:Actor
- IREB:Requirement

Statement Class Names
- IREB:satisfiedBy
- oslc_rm:refinedBy

Property Names
- dcterms:title
- SpecIF:Priority

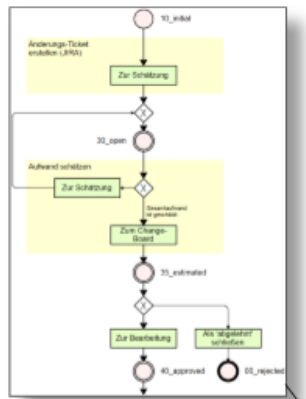
Property Values
- SpecIF:priorityHigh

Konzept schrittweise verfeinern und Inhalte vielseitig nutzen

Prozesse
Systemintegration
Fachobjekte
Annotationen

Integriertes System-Modell

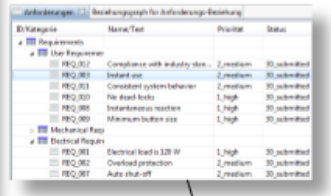
BPMN Prozesse



Integriertes System-Modell

UML/SysML Systemmodelle

Anforderungen User Stories



Integriertes System-Modell

SpecIF
Specification Integration Facility

http://specif.de

DOCX/PDF

WIKI

ALM/PLM

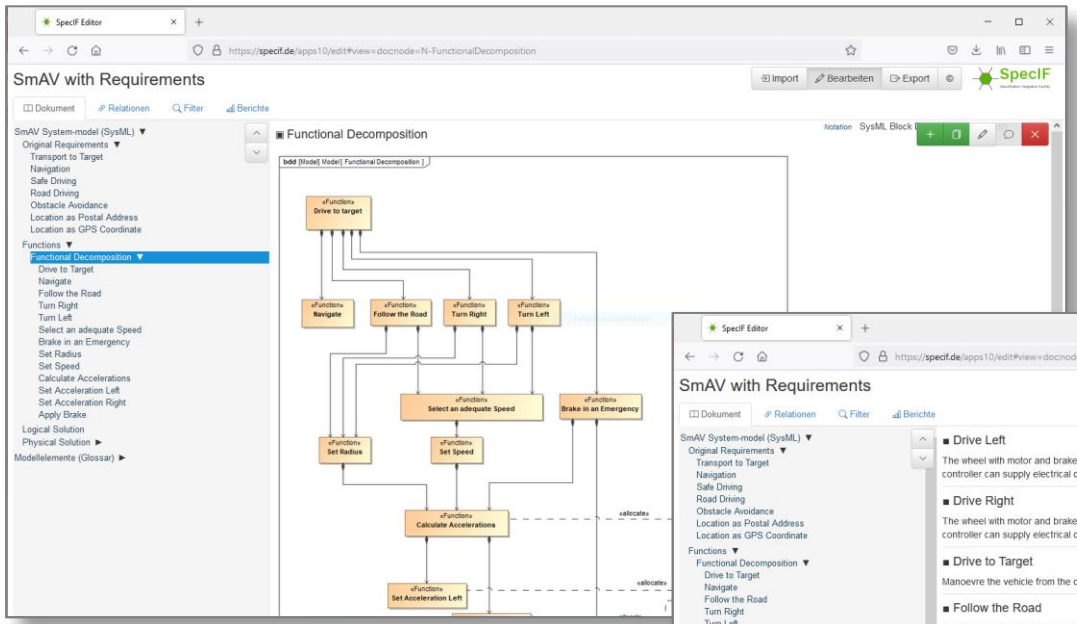
ReqIF

OSLC

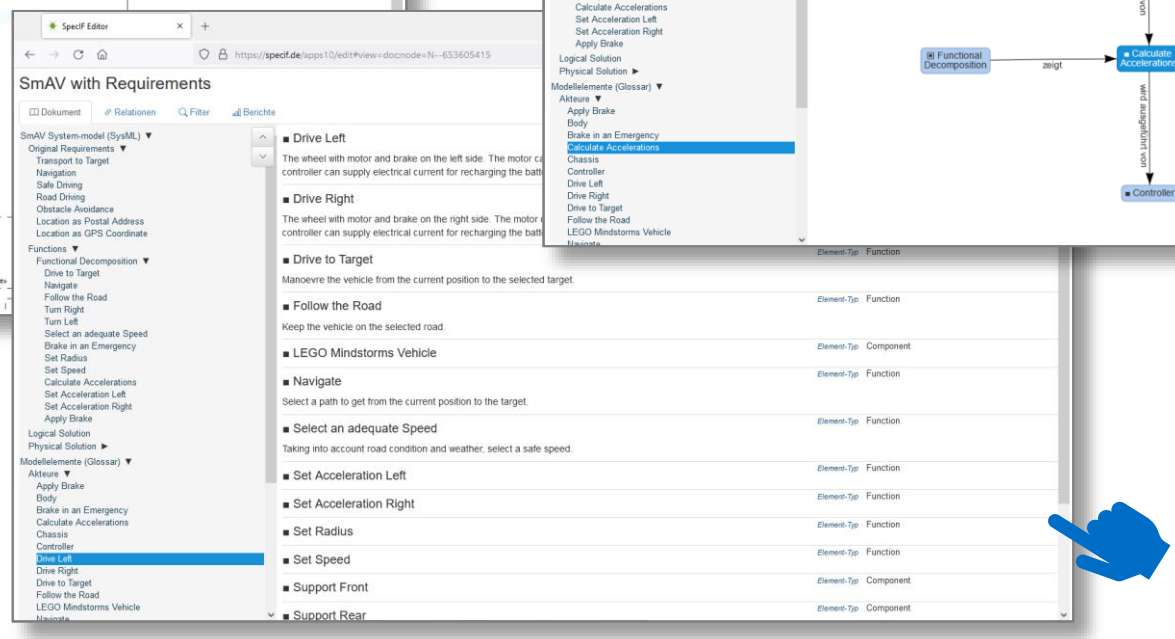
Linked Data

4 Beispiel: Anforderungen und Systemmodell zusammen führen

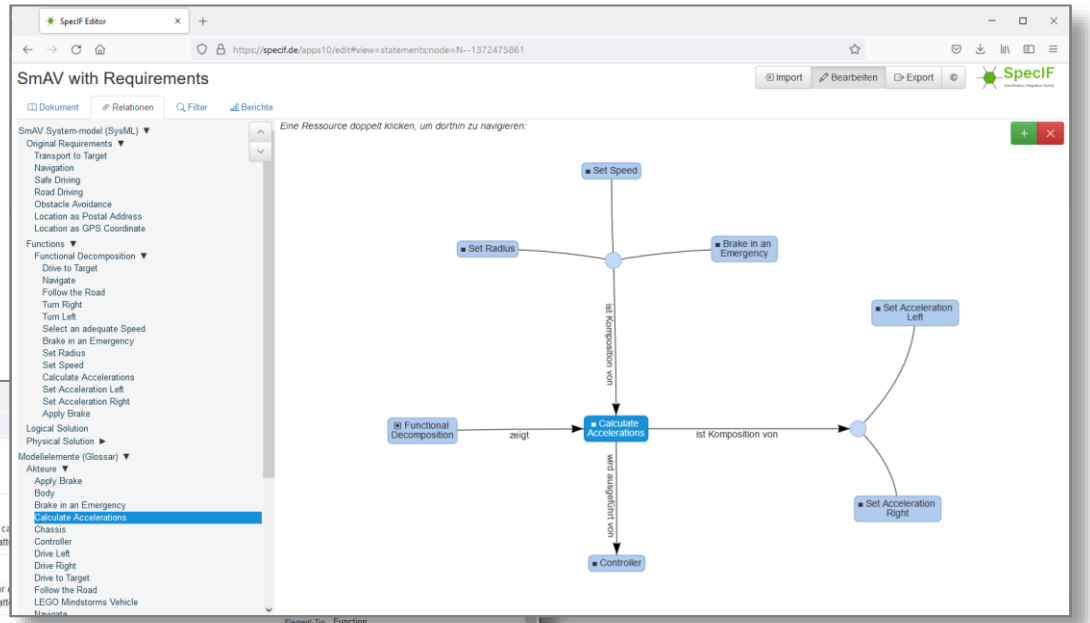
Beispiel:
Small Autonomous Vehicle



Alle Modelldiagrame



Alle Modellelemente („Resources“)

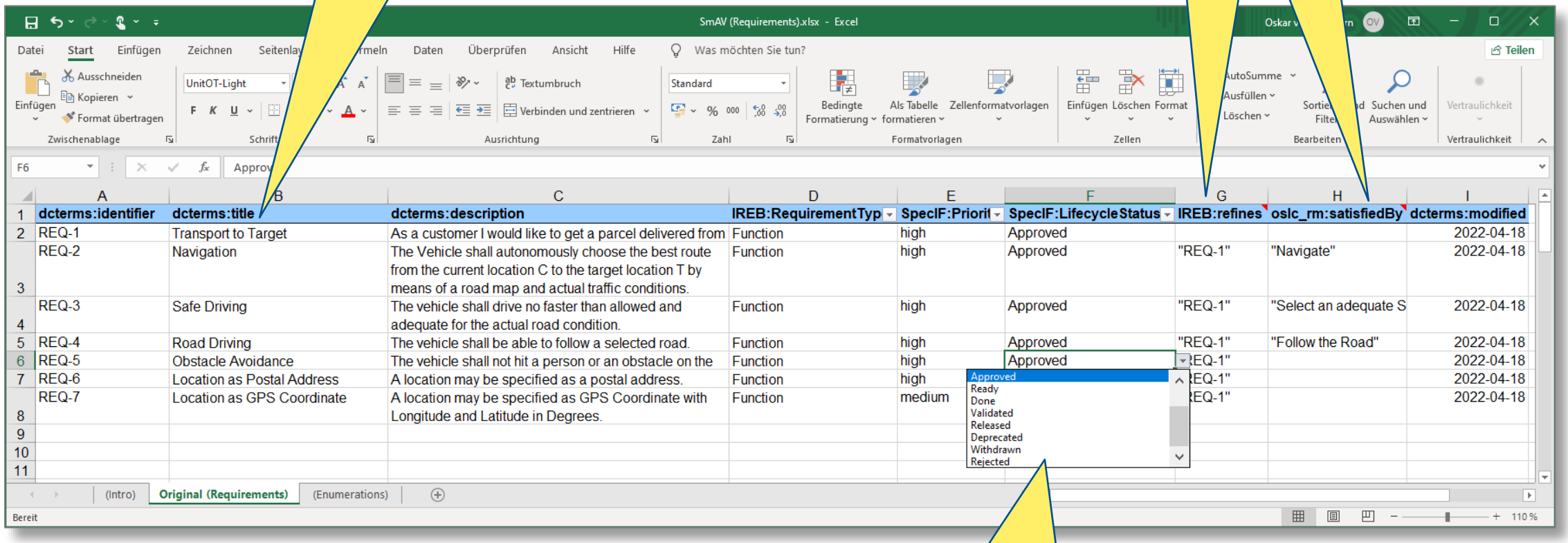


Alle Relationen („Statements“)

Anforderungen aus Excel® Tabellen hinzu fügen

Vokabular-Begriffe

Relationen



	A	B	C	D	E	F	G	H	I
1	dcterms:identifier	dcterms:title	dcterms:description	IREB:RequirementTyp	SpecIF:Priorit	SpecIF:LifecycleStatus	IREB:refines	oslc_rm:satisfiedBy	dcterms:modified
2	REQ-1	Transport to Target	As a customer I would like to get a parcel delivered from	Function	high	Approved			2022-04-18
3	REQ-2	Navigation	The Vehicle shall autonomously choose the best route from the current location C to the target location T by means of a road map and actual traffic conditions.	Function	high	Approved	"REQ-1"	"Navigate"	2022-04-18
4	REQ-3	Safe Driving	The vehicle shall drive no faster than allowed and adequate for the actual road condition.	Function	high	Approved	"REQ-1"	"Select an adequate S	2022-04-18
5	REQ-4	Road Driving	The vehicle shall be able to follow a selected road.	Function	high	Approved	"REQ-1"	"Follow the Road"	2022-04-18
6	REQ-5	Obstacle Avoidance	The vehicle shall not hit a person or an obstacle on the	Function	high	Approved	EQ-1"		2022-04-18
7	REQ-6	Location as Postal Address	A location may be specified as a postal address.	Function	high		EQ-1"		2022-04-18
8	REQ-7	Location as GPS Coordinate	A location may be specified as GPS Coordinate with Longitude and Latitude in Degrees.	Function	medium		EQ-1"		2022-04-18
9									
10									
11									

Enumerierte Werte

SpecIF: Was ist anders ?

1. SpecIF baut auf bekannte Notationen und Formate; ersetzt sie nicht.
2. SpecIF trägt Bedeutung durch Vokabular und einfache Aussagenlogik.
3. Funktioniert durch Verwendung *weniger* fundamentaler Elementtypen.
4. Graph-Daten sind hochskalierbar und erlauben ultra-schnelle Suche.
5. Dynamisches Datenmodell mit Schema und Constraint-Prüfung.
6. Letztlich ist SpecIF ein semantisches Netz mit typisierten Knoten und Kanten.

5 Vorteile einer open-source „Community Edition“

… für Kunden:

- Direkter Einfluss auf Features und Prioritäten
- Agile(s) Projekt(e), erstes "Minimal Viable Product" innerhalb eines Jahres
- Investition in ein gemeinschaftliches Open-Source-Projekt

… für Anbieter:

- Niedrige Einstiegsschwelle
- Lösung wird ohne Marketingaufwand bekannt
- Diskussion und Unterstützung durch freiwillige Mitglieder
- Markterprobte Basis für "Enterprise Editions" mit professionellen Features
- Der beliebteste Marketingansatz für innovative Softwareprodukte "Freemium".
- Es gibt viele erfolgreiche Beispiele

Beteiligen Sie sich durch Eigenleistung oder Finanzierung

Programm-Management

- Technischer Exekutivausschuss und „Product Owner“
- … bestimmt zu implementierende Funktionen und die Priorität
- Mitglieder tragen durch Eigenleistung oder Finanzierung bei

Agile Entwicklungsteams

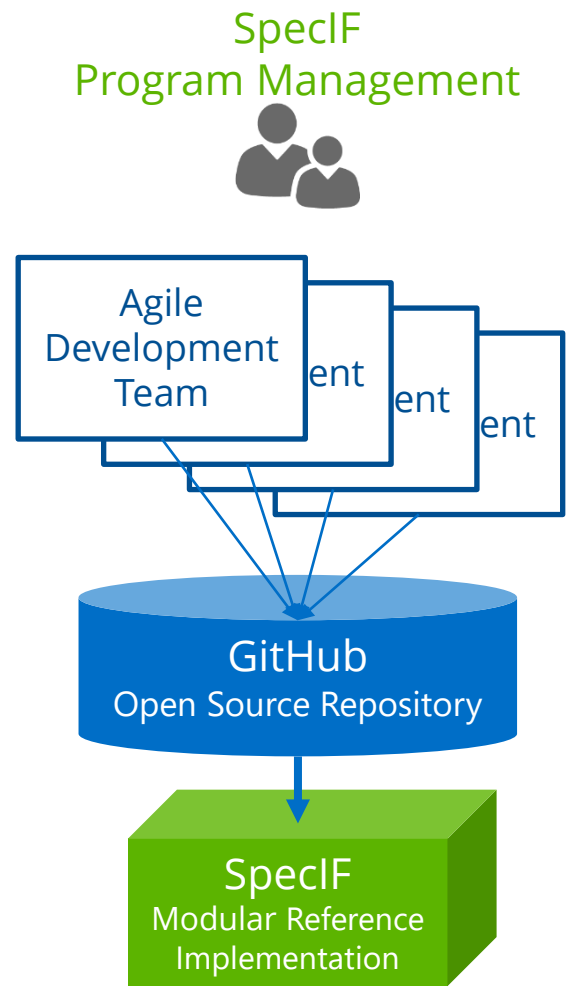
- Jeweils bis zu 7 Mitglieder für Entwicklung und Test

Repository

- GitHub Open-Source-Software mit Apache- oder MIT-Lizenz

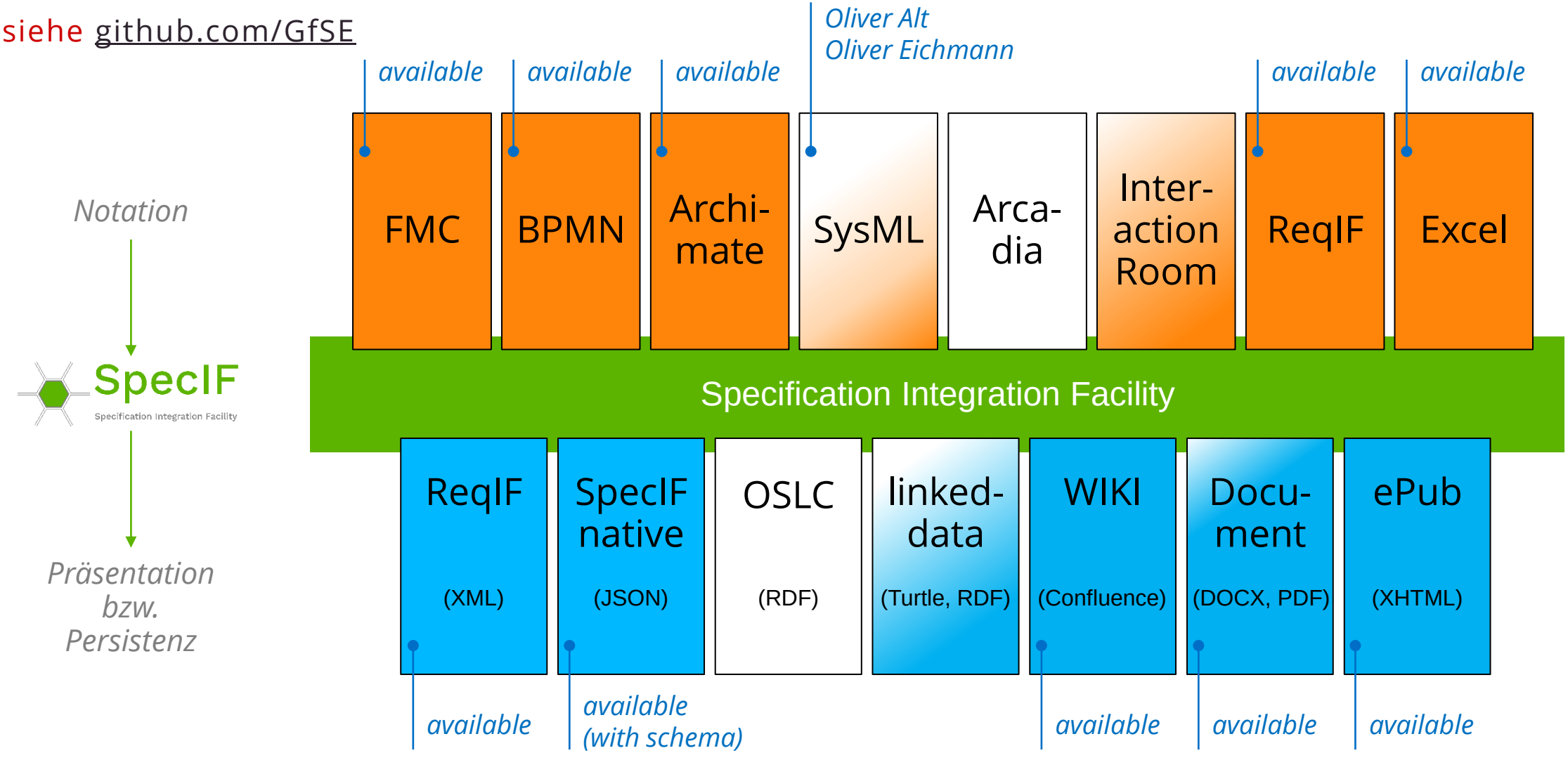
Ergebnis

- Community Edition = modulare Referenzimplementierung
- … als Basis für Produktinnovationen von Softwareanbietern



Status ...

siehe github.com/GfSE



Dokumentation und Beispiele

- SpecIF Home
- SpecIF Schema und Transformationen auf GitHub (Open Source, Apache License)
  → Das GitHub Ticket System für Fragen, Vorschläge und Anfragen nutzen
- Hosted SpecIF Schema and Consistency-check (→ neue Releases)
- SpecIF Viewer und Editor
- Beispiele:
  - Small Autonomous Vehicle (Notation: SysML)
  - System Engineering Collaboration (Notation Archimate)
  - Telephone Connection Request (Notation BPMN+Archimate+XSLX)
  - Dimmer (Notation FMC)
- SpecIF Vokabular

Literatur

[1] Wendt, S.: Ein grundlegender Begriffsrahmen für das Wissensmanagement im Software-Engineering. In Proceedings „Knowtech“ Dresden 2001.

[2] Knöpfel, A.; Gröne, B.; Tabeling, P.: Fundamental Modelling Concepts – Effective Communication of IT Systems. ISBN-13: 978-0-470-02710-3. John Wiley&Sons, Chichester, 2005.

[3] Kaufmann, U., Pfenning, M.: 10 Theses about MBSE and PLM.

[4] Object Management Group: Systems Modeling Language (OMG SysML™), Version 1.3, June 2012.

[5] Object Management Group: Requirements Interchange Format (ReqIF).

[6] Open Services for Lifecycle Collaboration (OSLC).

[7] Specification Integration Facility (SpecIF).

[8] Dungern, O.v.: Semantic Model-Integration for System Specification – Meaningful, Consistent and Viable, 7.Grazer Symposium Virtuelles Fahrzeug, Graz, Mai 2014.

[9] Dungern, O.v.: Integration von Systemmodellen mit fünf fundamentalen Elementtypen. TdSE Tag des Systems Engineering der GfSE, Ulm, November 2015.

[10] Dungern, O.v.: Von Anforderungslisten zu vernetzten Produktmodellen – am Beispiel der Gebäudeautomation. REConf, Unterschleißheim, März 2016.

[11] Dungern, O.v.: Semantic Model Integration for System Specification. TdSE Tag des Systems Engineering der GfSE, Herzogenaurach, October 2016.

[12] Uphoff, F.: Konzept und prototypische Implementierung der Modellintegration der Interaction-Room-Methode in die Specification Integration Facility, Kamp-Lintfort, März 2017.

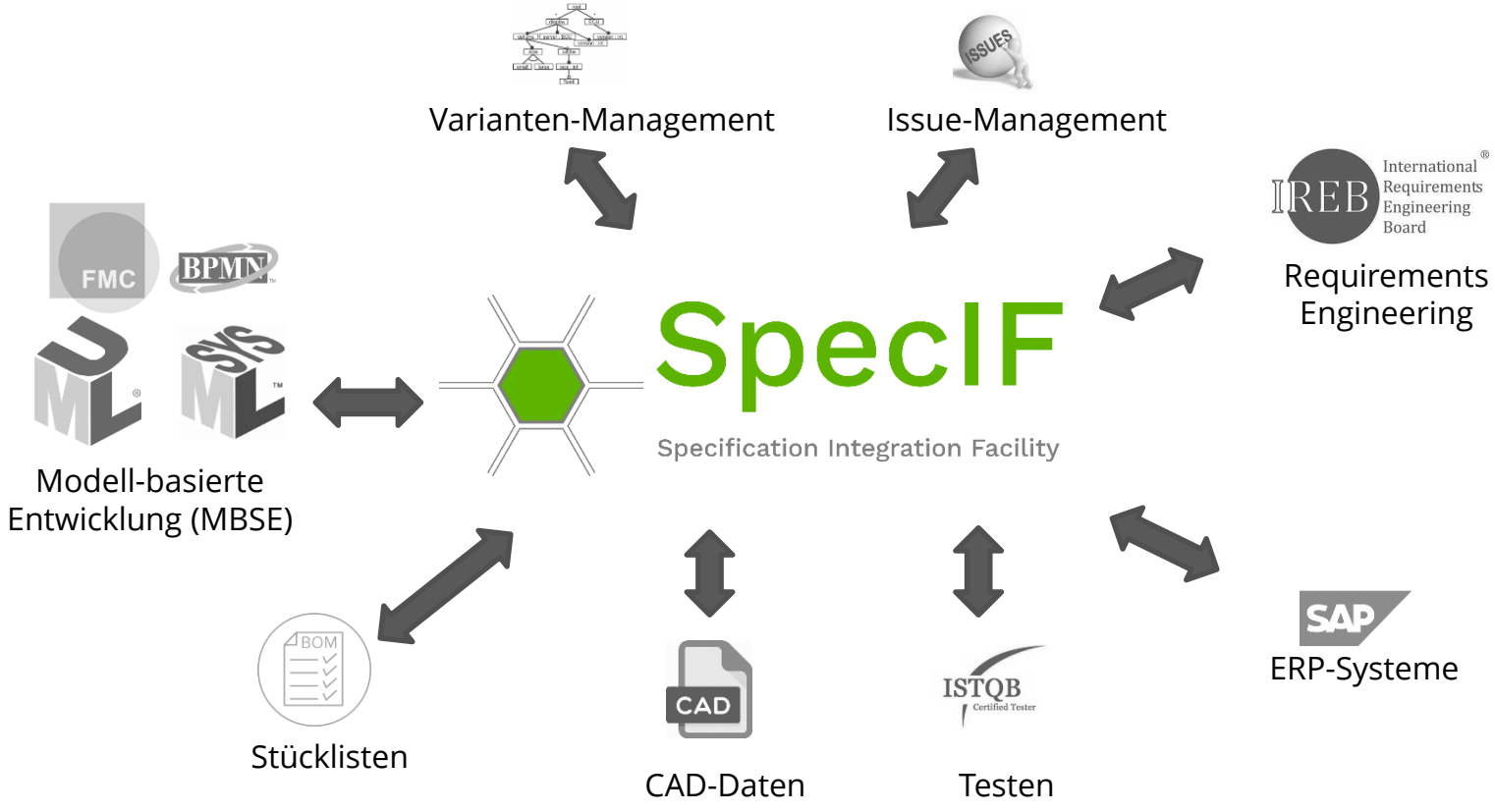
[13] Mochine, P.; Sünnetcioglu, A.; Dungern, O.v.; Stark, R.: SysML-Modelle maschinell verstehen und verknüpfen. TdSE Tag des Systems Engineering der GfSE, Paderborn, Oktober 2017.

[14] Alt, O.: SpecIF - Die kommende vielschichtige Datenquelle für Spezifikationsdaten. Fachgruppentreffen GI-RE, Nürnberg, November 2018.

[15] Dungern, O.v.: Model-Integration with SpecIF. ProSTEP ivip e.V. SysML-Workflow-Forum November 2019.

Interessant ?

od@enso-managers.de
oliver.alt@mdd4all.de



Details zur weiteren Diskussion

SpecIF Ziele und Nutzen

- Modellbasierte Systemspezifikationen zwischen Organisationen und Werkzeugen austauschen.
- Texte und Modelle aus verschiedenen Werkzeugen zusammen führen.
- In einem gemeinsamen Kontext navigieren, suchen und prüfen.
- Den Produktlebenszyklus von Anfang bis Ende verwalten („end-to-end“):
  - Referenz für alle Ingenieurdisziplinen
  - Methoden-übergreifend
  - Technologie-neutral
  - Hersteller-unabhängig
  - Schema-treu
  - Standard-einhaltend
  - Offen und kollaborativ

CC BY SA

What is SpecIF ?

- „**Spec**ification **I**ntegration **F**acility“, a GfSE initiative

SpecIF
Open SE Models

- SpecIF adds **conventions to convey meaning**
  using known technical formats such as ReqIF or OSLC.

1. Vocabulary for Objects,
   Relations and Attributes

   „Requirement“, „Actor“, „State“, „Event“ …
   „satisfies“, *„reads“*, *„contains“*, „triggers“, …

2. Logic Assertions
   („First-order predicate logic“)

   „A Component *satisfies* a Requirement“
   „An Event *triggers* an Activity“

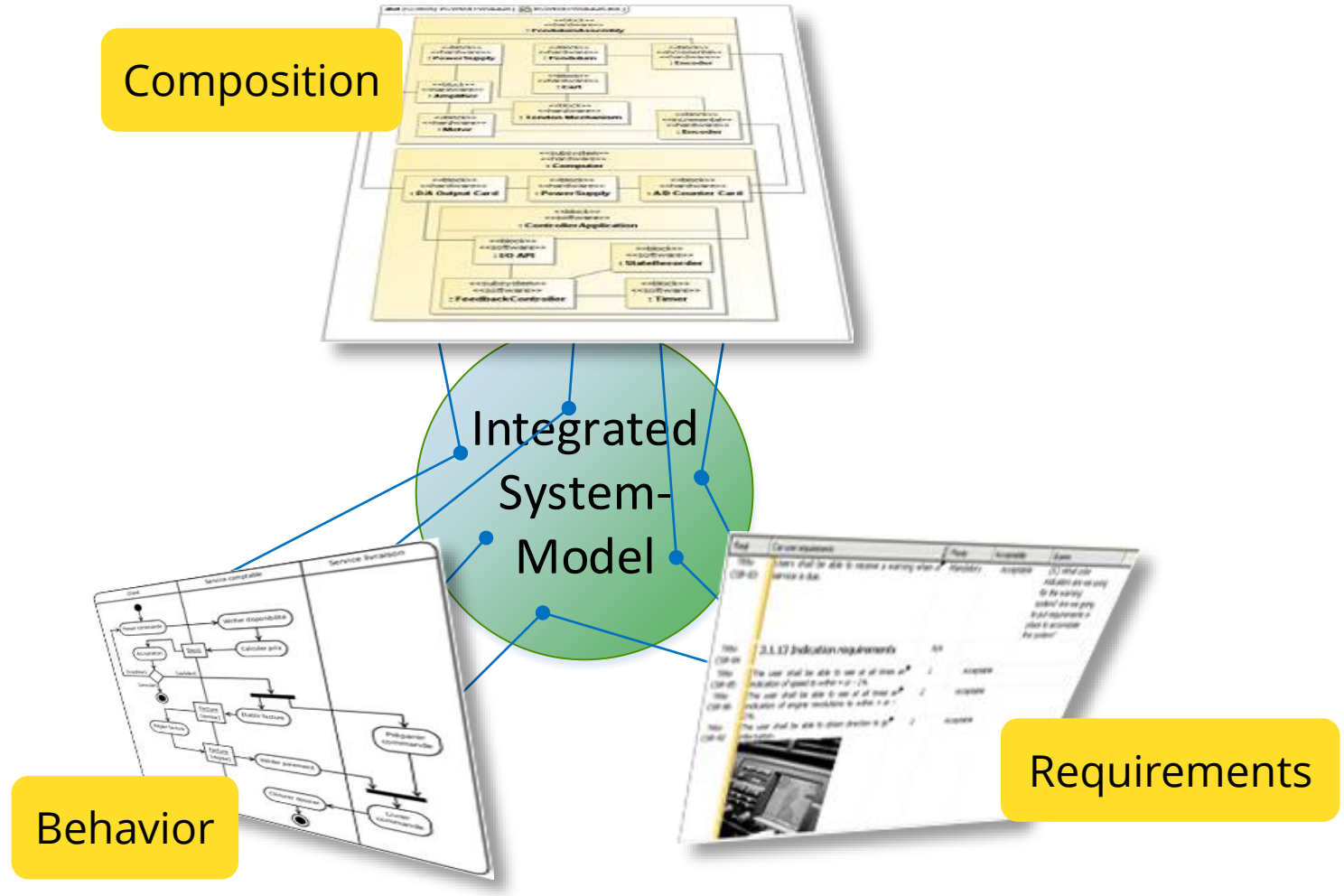
→ SpecIF carries both the „Visible“ and the „Meaning“

Purpose: Model Integration

- Search, navigate and audit partial engineering results in a common context
- Interrelate information elements of disparate sources to create a common view
- Find inconsistencies and gaps between different views
- Use-Cases:
  - Localize requirements on a BoM: Which components are affected when changing a requirement (or vice versa)?
  - When a use-case mentions a data-object, which system components and interfaces are involved?
  - Interrelate system structures with process models: Which activities are affected when modifying a system component (or vice versa)?
  - Associate a FMI simulation-routine with a SysML system component (block)
  - Collect and compare information about an element from different sources.

Consolidate model elements from different diagrams

- Key to success is the abstraction using 5 fundamental model element types
- Impossible with 162 model element types in SysML and almost as many in BPMN
- Even within SysML the current tools fail to properly consolidate model elements from different model views



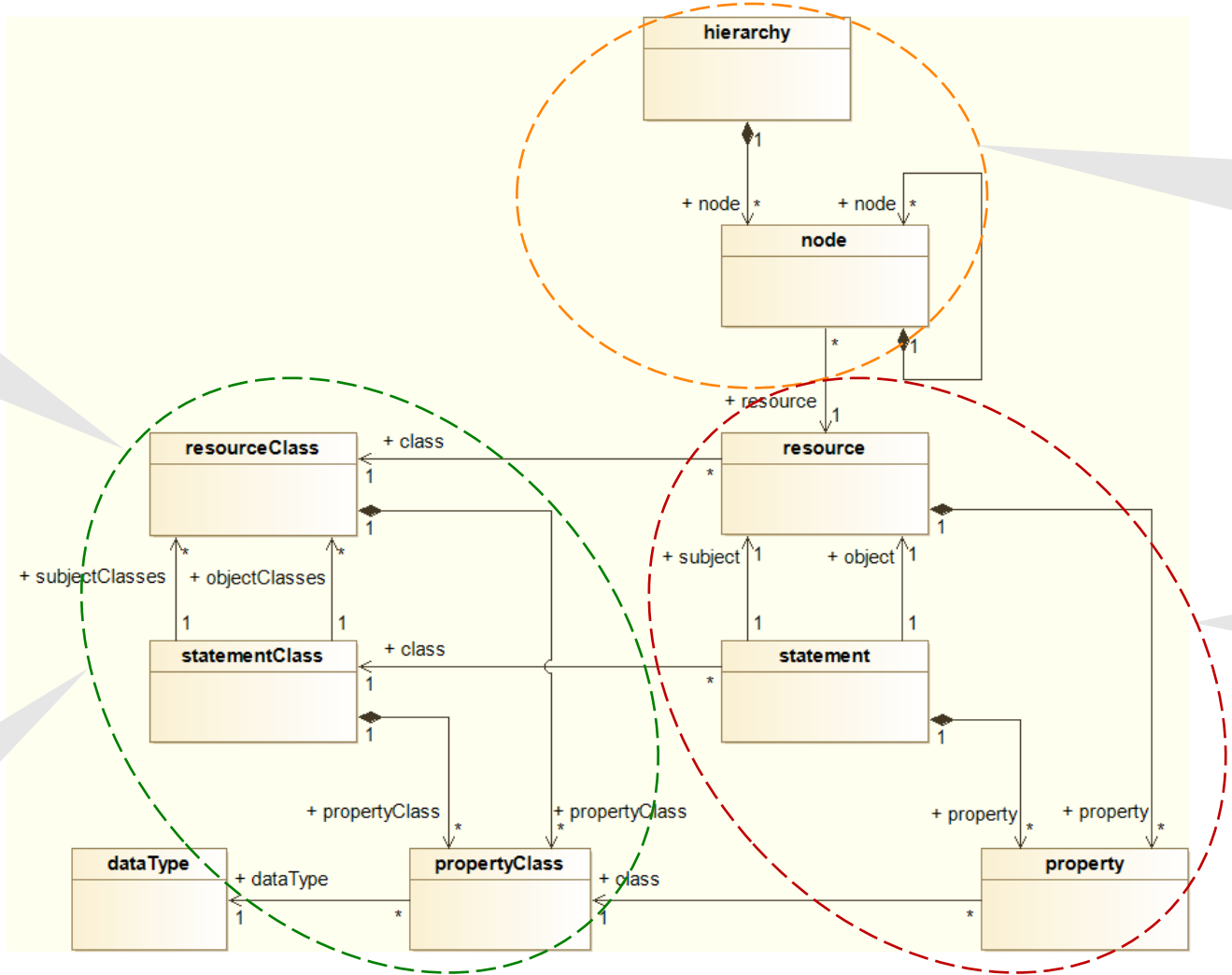
A SpecIF data set contains both the Set types („model“) and the instances („data“ = „payload“)

Hierarchical ordering of Resources (e.g. for convenient reading)

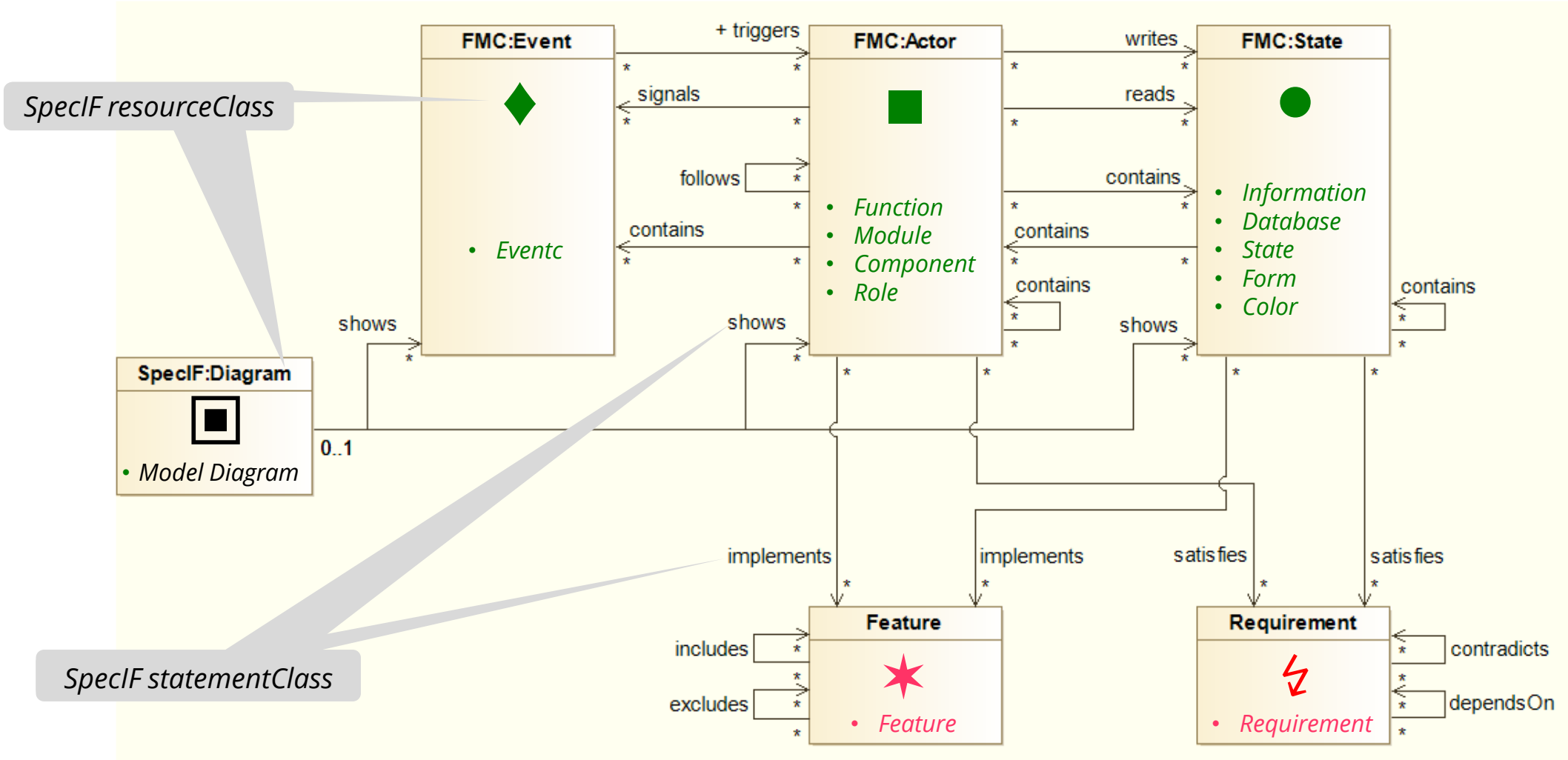
SpecIF model with Resource- and Statement-Classes

SpecIF data (payload) with Resources and Statements

The classes can be defined at runtime („dynamic model“)



The SpecIF Integration Model with 5 Fundamental Model-element Types



Simple Model-Integration „Adopt“

- Adopt existing model-element types, if they are equivalent
- Add new diagrams (views)
- Adopt existing model-elements having the same title
- Add new statements
- Build new glossary of model-elements, sorted by fundamental type