Semantic Technologies for Systems Engineering (ST4SE)

Update at INCOSE IW, 27 Jan 2019, Torrance, CA, USA

Hans Peter de Koning

European Space Agency, ESA/ESTEC, Noordwijk, The Netherlands

Based on earlier presentations and
contributions by other ST4SE Core Team Members

Objectives of the ST4SE Foundation

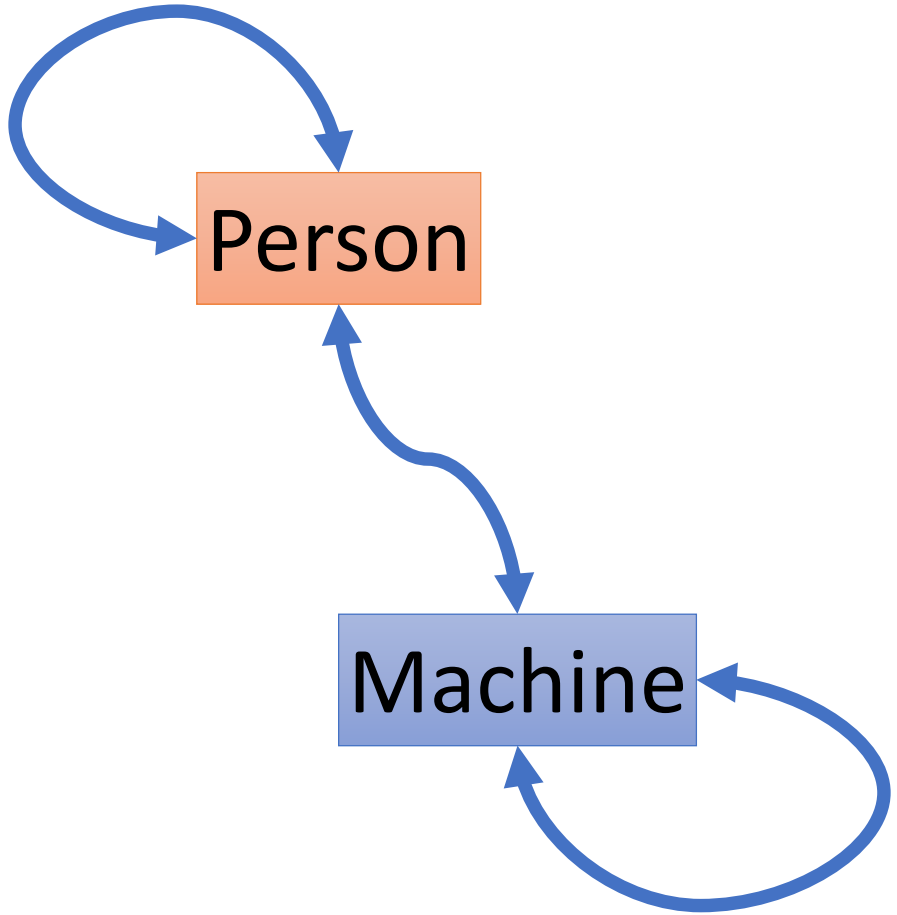
To promote and champion the open-source development and utilization of ontologies and semantic technologies to support system engineering practice, education, and research

1. Provide a semantically rich language to communicate among systems engineers and other stakeholders
2. Define patterns that can be used to check for consistency and completeness
3. Support querying of information from model
4. Focus on adding value by balancing the expected benefits from being formal and the cost of being formal

MBSE Challenge – 3 Kinds of Communication

- Person ↔ Person
- Machine ↔ Machine
- Person ↔ Machine

- All bi-directional (of course)
- All need to work flawlessly



Outline

- Background on Semantic Technologies
  - Knowledge representation, reasoning, querying
- Semantic Technologies for System Engineering
  - Motivation
  - Scope and focus
  - Relationship between ST4SE and SysML 2.0
- ST4SE Approach
  - Open-source foundation
  - Bootstrapping: (best) practices for defining, demonstrating and documenting patterns

Increasing levels of semantic precision

(and understanding by machines)

Controlled Vocabulary	Taxonomy	Ontology
• aka glossary of terms • Natural language definitions • May include synonyms • Cannot include homonyms without further qualification, since each term should be unique • May include citations to a reference source • May include some “see also” cross references • Term definitions should not be circular	• Controlled Vocabulary plus: • Hierarchical tree(s) of broader / narrower terms • Similar to mathematical sub-setting or OO generalization / specialization • Some formal structure, but still usually represented in natural language • Can range from informal to more formal taxonomy • SKOS (Simple Knowledge Organization System, W3C standard) is an example of a more formal taxonomy specified in RDF	• Taxonomy, plus: • Terms → Concepts identified by some unique identifier as well as all relationships between them • Conforming to (some) formal logic • Machine interpretable semantics • In addition label to name each concept for human understanding • Multiple labels (aliases) supported – e.g. to support different natural languages • Homonym labels allowed, but not recommended because confusing for humans

Knowledge Representation with OWL — Web Ontology Language

- *OWL is designed to represent rich and complex knowledge about things, groups of things, and relations between things* [3]

- OWL Ontologies can be easily exchanged as RDF documents — RDF (Resource Description Framework) [4] is the standard model for data interchange on the Web

- OWL is the most widely-used Knowledge Representation language in the world—by a wide margin

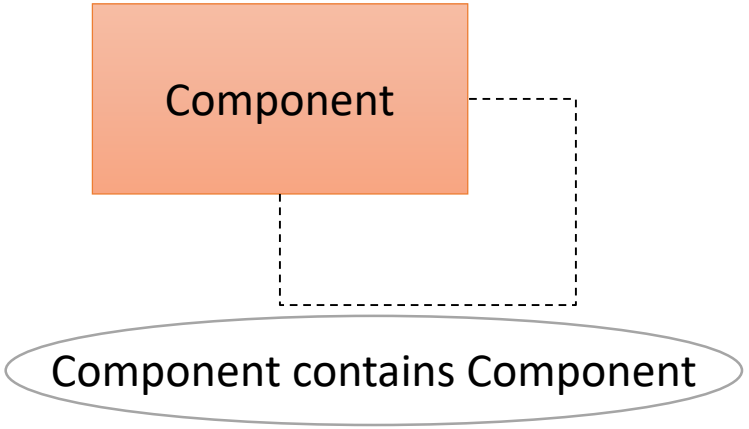
Reasoning with OWL — Selected Description Logic (DL)

- *OWL is a computational logic-based language such that knowledge expressed in OWL can be exploited by computer programs* [3]

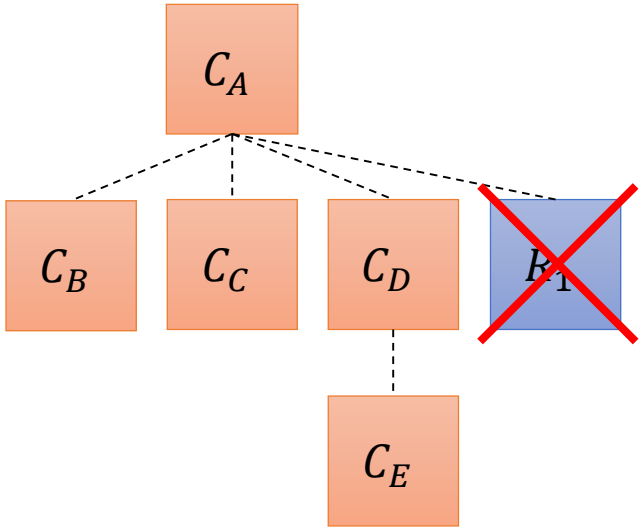
- The Description Logic subset of OWL (OWL2 DL) carefully balances expressivity with computational completeness and decidability
- Commercial and free reasoning tools are available
- Practical reasoning algorithms exist that are both:
  sound → all inferences drawn are valid
  complete → all valid inferences are drawn

Reasoning with OWL — Inference and Consistency

- Through reasoning, one can infer implicit information and make it explicit
  - Ex: "C_A *containsTransitively* C_E" can be concluded from explicit "contains" relationships



- Information expressed in OWL can be *semantically validated*
  - Unsatisfiable (i.e., overconstrained) classes → inconsistencies
  - Ex: "C_A *contains* R_1*(Requirement)*" → inconsistent (Assuming *Requirement* and *Component* are disjoint)
  - Opportunity to catch errors on *every exchange*



Querying OWL Models — SPARQL Protocol and RDF Query Language

- In addition to reasoning, we need the ability to ask questions about information
  - What is the measured mass of the flight system?
  - What are all the (recursively) contained components of the flight system?
  - What requirements refine R-12345?
  - Does every component have a supplier?
- SPARQL [5] is a language and distributed query protocol to pose such questions and get answers
- Numerous commercial and free implementations are available

Outline

- Background on Semantic Technologies
  - Knowledge representation, reasoning, querying
- Semantic Technologies for System Engineering
  - Motivation
  - Scope and focus
  - Relationship between ST4SE and SysML
- ST4SE Approach
  - Open-source foundation
  - Bootstrapping: (best) practices for defining, demonstrating and documenting patterns

Semantic Web and Systems Engineering

- SE is inherently a synthetic activity that unites information
  - across multiple disciplines,
  - across organizational boundaries (extended enterprise / supply chain),
  - across multiple product lifecycle phases
- Agreement on syntax and semantics for concepts and properties for this disparate information is essential
  - avoid unnecessary costs and delays due to work of translation
  - avoid unnecessary risks due to errors in translation
  - achieve affordable, maintainable interoperability (between different tools)
- Agreement is difficult
  - Different systems engineers use different conceptualizations of systems engineering

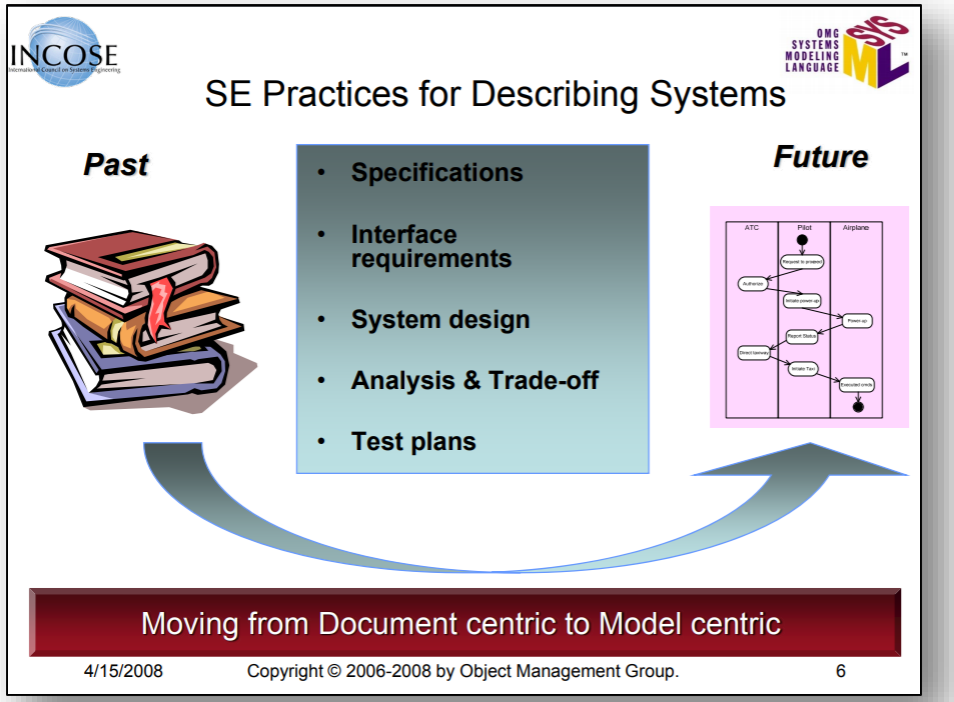
Scope and Focus of ST4SE

- Primary Focus: Patterns and Notions
  - Specific to Engineered Systems
  - Specific to Systems Engineering
  - Expressible in OWL2 DL (Description Logic)

- Out of scope (for now)
  - Relevant but not specific to SE: e.g., QUDV (Quantities, Units, Dimensions, and Values), State Machines, …
  - Application domain specific: e.g., space systems ontology
  - Not expressible in OWL2 DL: e.g., probabilistic logic, temporal logic, …

Relationship between ST4SE and SysML

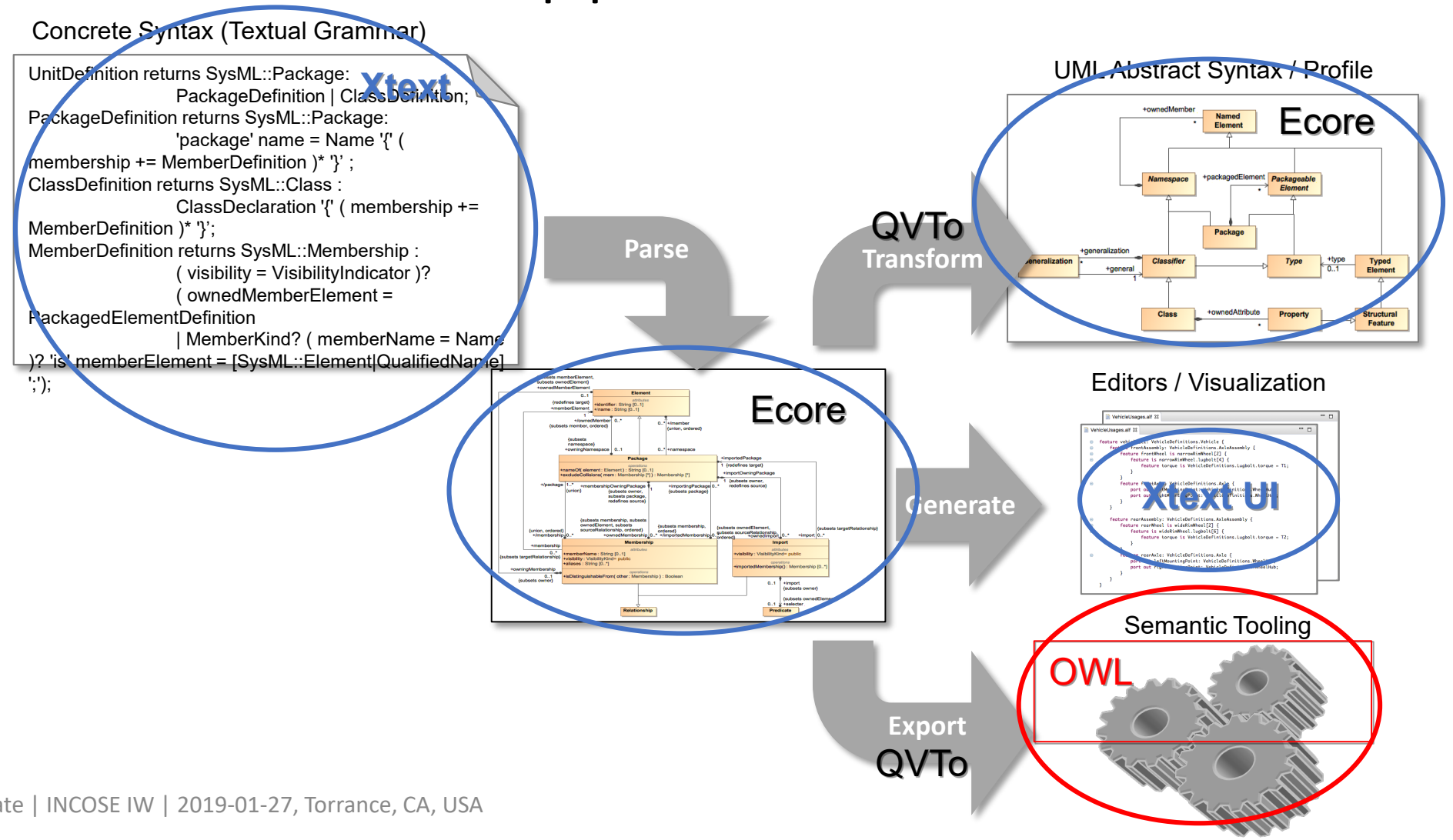
- MBSE — more formal, unambiguous, semantically rich
- SysML 1.x — important first step, but:
  - Limited taxonomy of concepts: almost everything is a «block», properties are local to «block» namespace
  - Weak semantics: lack of strong logical foundation, ill-suited for automated reasoning
- SysML 2.0 — promises to be an important next step
  - SysML v2 Submission Team proposes and prototypes full mapping to OWL
  - ST4SE coordinates with SysML v2 team to enable reasoning/querying on patterns



SysML v2 mapping to OWL

- The SST is developing a bi-directional mapping from the SysML v2 meta-model to OWL
- This will ensure in the future that any SysML v2 model can be transformed into an OWL equivalent, on which established automated reasoning can be applied
  - To support rule checking – e.g. to establish model quality
  - To produce all sorts of entailments and perform other graph computations

SysML v2 Submission Implementation Approach



Outline

- Background on Semantic Technologies
  - Knowledge representation, reasoning, querying
- Semantic Technologies for System Engineering
  - Motivation
  - Scope and focus
  - Relationship between ST4SE and SysML
- ST4SE Approach
  - Structure: Open-source foundation
  - Bootstrapping: (best) practices for defining, documenting and demonstrating patterns

ST4SE Foundation

- The ST4SE Foundation is still in development
  - Does not yet exist as a legal entity – work in progress
- It is modeled after successful open-source software development efforts such as Apache and Eclipse
  - Will be officially hosted on GitHub
  - Core team provides technical guidance in both SE and Semantic technology
  - Contributions can be made by any/all volunteers
  - Core team will moderate to ensure architectural coherence
- Steering group leadership
  - Chi Lin, Integrated Model-Centric Engineering Program Manager, Jet Propulsion Laboratory
  - Dinesh Verma, Executive Director, U.S. Systems Engineering Research Center

Process for Capturing Patterns & Notions — Primary Focus of Core Team So Far

1) Define the scope
   - Delineate a perspective and corresponding candidate patterns
   - Scoping a small set of patterns at a time allows the discussion to remain focused
2) Brainstorm potentially relevant patterns and notions
   - Will likely result in different overlapping and possibly conflicting terms
3) Reconcile and converge
   - Discuss what the terms mean, aiming to move towards a common understanding
   - Agree on the terminology for patterns and notions
4) Formalize in OWL
   - Capture notions and patterns in OWL (TBox)
   - Create usage examples in OWL (ABox)
5) Demonstrate the value of the patterns
   - Create example query patterns (SPARQL) and/or reasoning patterns (DL solver)
6) Document on ST4SE Wiki
   - Could/should be automatically generated from OWL in the future

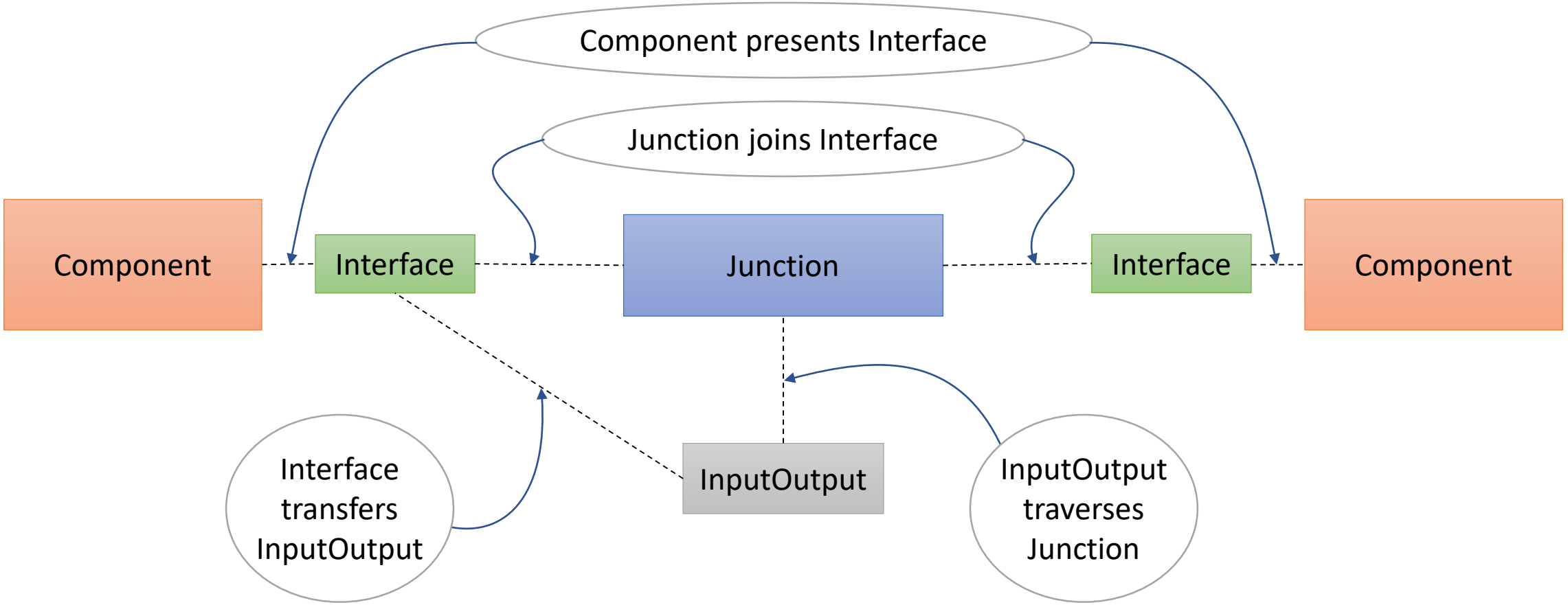
Example: Patterns Related to Interfaces

1) Define the Scope and 2) Brainstorm Patterns & Notions

- Potential Notions
  - Interface, Interaction , Junction, Item, Flow, SystemOfAccess, InterfaceEnd, Connector, Binding, Direction, Input/Output…
- Potential Patterns
  - Component presents Interface
  - Inputs/Outputs flow through Interface
  - Interaction describes the behavior of Interface
  - SystemOfAccess provides the transport medium of Interface
  - Junction joins Interface pair
  - Item or Flow traverses Junction
  - Interface transfers in/out Flow
  - Interface has a Function
  - Interface consists of two InterfaceEnds and a Connection
  - Component realizes Interface

Example: Patterns Related to Interfaces

3) Reconcile and Converge

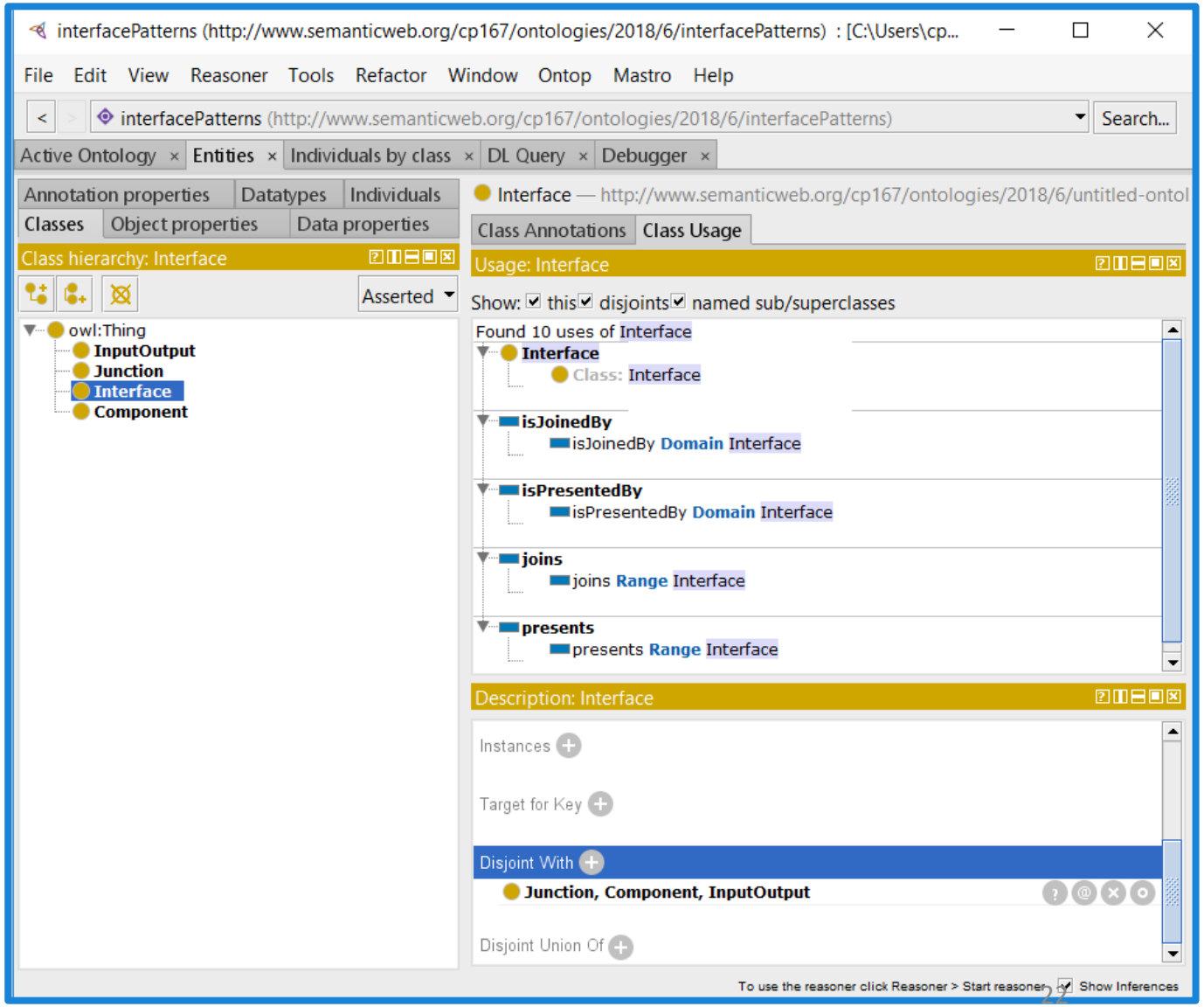


Example: Patterns Related to Interfaces

4) Formalize in OWL

- Formalize the reconciled patterns and notions in OWL (TBox)

- Create usage examples (OWL ABox)
  - To illustrate the patterns and notions
  - To demonstrate reasoning and querying



Example: Patterns Related to Interfaces

5) Demonstrate the value of the pattern

- Demonstrate types reasoning/querying based on the patterns and notions that are useful from a systems engineering perspective

- Query examples:
  - List all interfaces associated with a particular component
  - List all the interfaces, I_A, that are "compatible" with a interface, I_B (i.e., for which there exists a Junction that joins to both I_A and I_B)

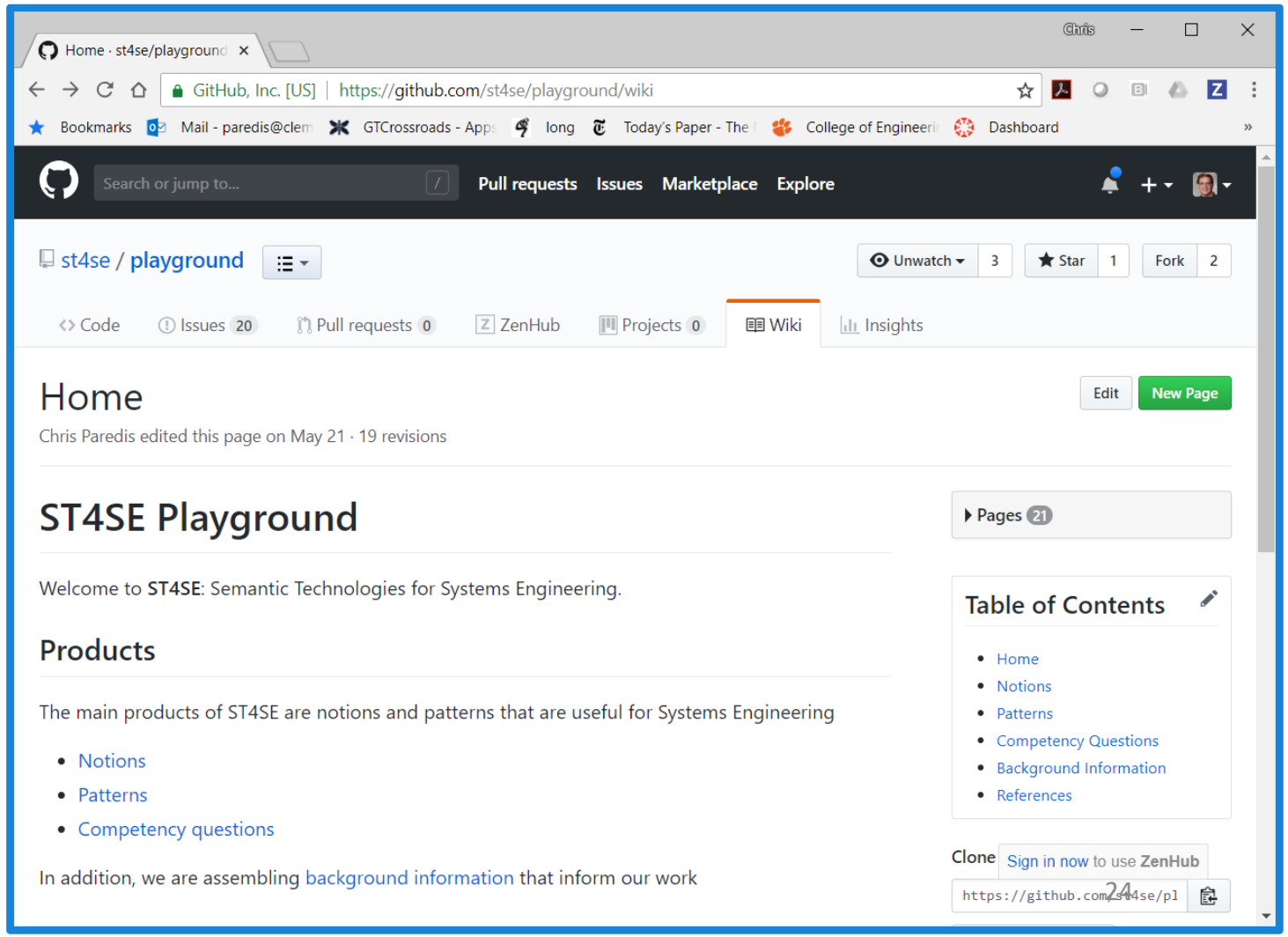
- Inference examples:
  - Determine whether two components are joined through a (particular) interface
  - Determine whether a particular InputOuput could potentially flow from Component A to Component B

Example: Patterns Related to Interfaces

6) Document on ST4SE Wiki

- Document the notions and patterns in human-readable form on the ST4SE Wiki

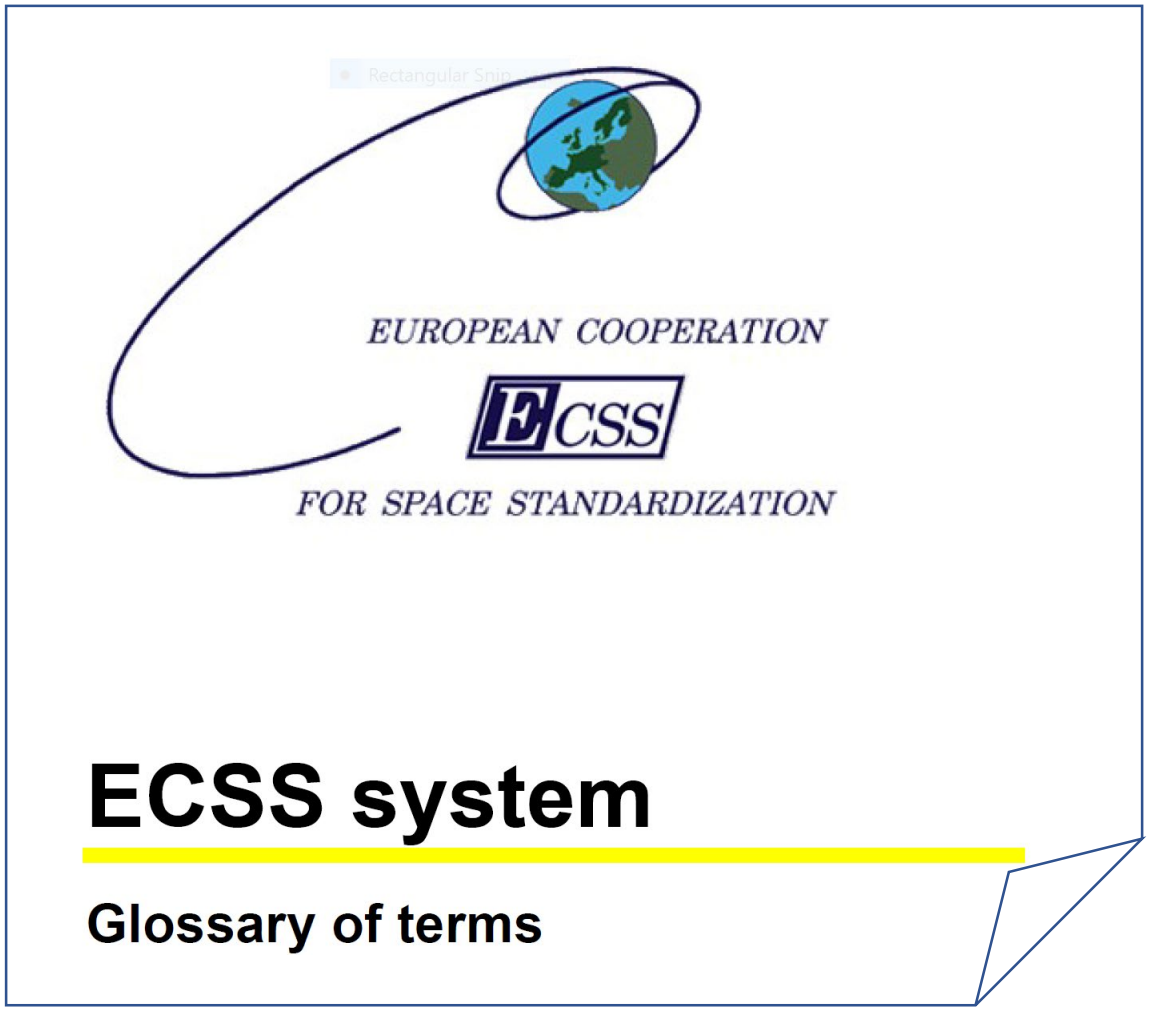
- Note: currently still in “playground” status, until we finalize (“best”) practices



Example: Contribute and Map Experiences from ECSS

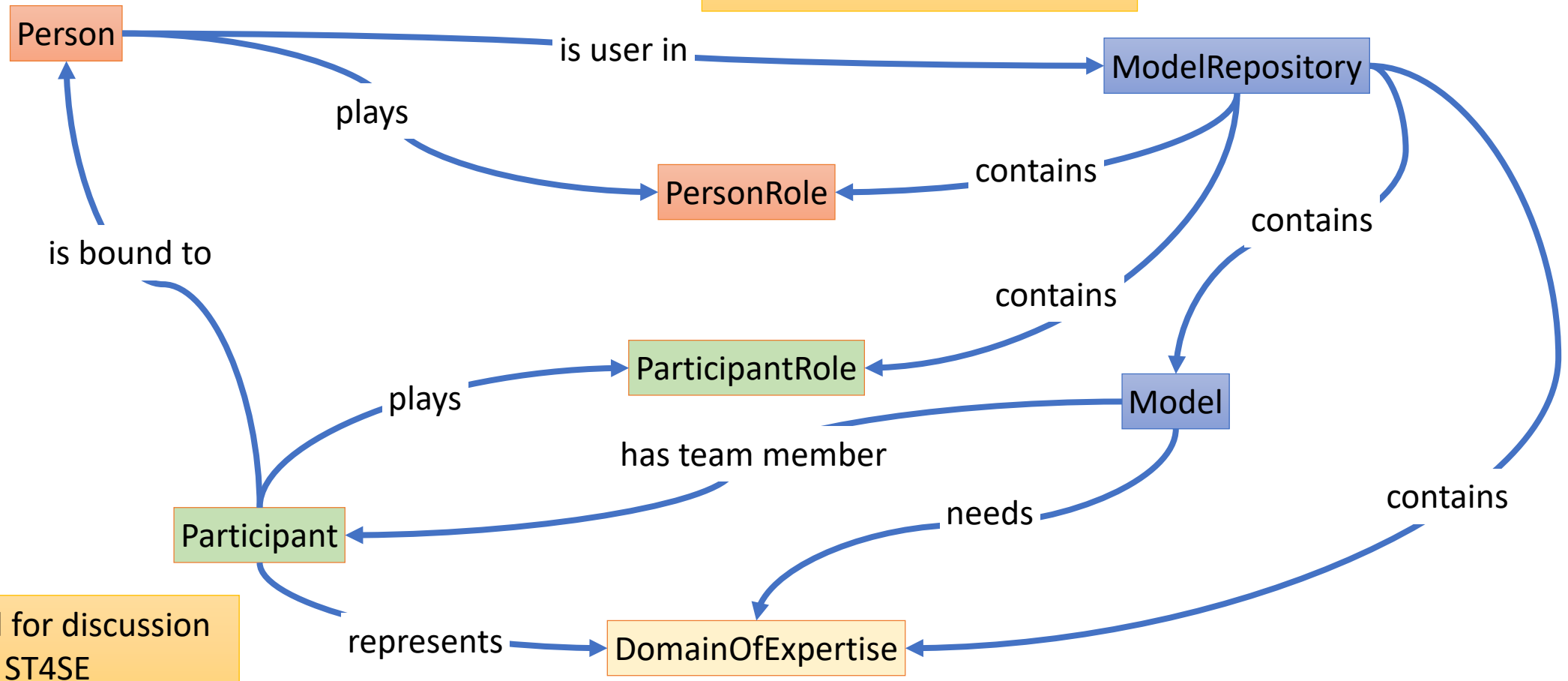
- My own contributions are grounded in ECSS [6] and SysML
- ECSS comprises ~200 standards with one global glossary of terms
  - Including Systems Engineering branch
  - Looking for generalization of embedded patterns and making them explicit

[ECSS Top Concepts Map]



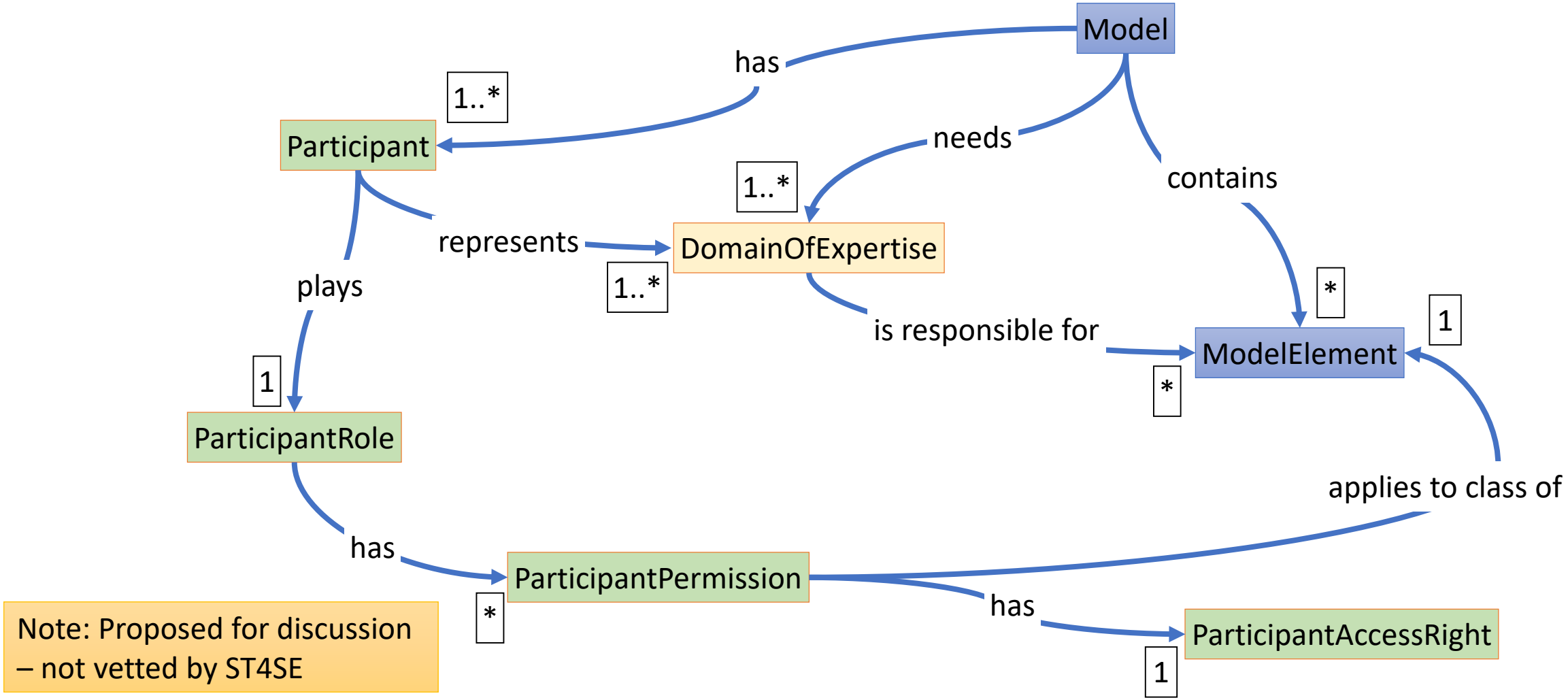
Example: Multi-Domain Collaboration (1/2)

Source: ECSS-E-TM-10-25



Note: Proposed for discussion – not vetted by ST4SE

Example: Multi-Domain Collaboration (2/2)



Summary — ST4SE Foundation

To promote and champion the open-source development and utilization of ontologies and semantic technologies to support system engineering practice, education, and research

1. Provide a semantically rich language to communicate among systems engineers and other stakeholders
2. Define patterns that can be used to check for consistency and completeness
3. Support querying of information from model
4. Focus on adding value by balancing the expected benefits from being formal and the cost of being formal

Acknowledgments

- ST4SE Core Team:
  - Mark Blackburn (Stevens Inst)
  - Hans Peter de Koning (ESA/ESTEC)
  - Henson Graves (Algos Associates)
  - Steven Jenkins (NASA/JPL)
  - David Long (Vitech Corp)
  - Chris Paredis (Clemson University)
  - Bill Schindel (ICTT System Sciences)
  - Todd Schneider (Engineering Semantics)
- Additional Support
  - Barry Smith (U Buffalo)
  - Chi Lin (NASA/JPL)
  - Dinesh Verma (SERC / Stevens Institute)
  - Troy Peterson (INCOSE)

References

1. Shadbolt, N., Hall, W., Berners-Lee, T., The Semantic Web Revisited, IEEE Intelligent Systems, 2006
2. W3C (World Wide Web Consortium), Semantic Web
3. W3C, Web Ontology Language (OWL)
4. W3C, Resource Description Framework (RDF)
5. W3C, SPARQL Query Language for RDF
6. European Cooperation for Space Standardization (ECSS)