Introduction – Eclipse Semantic Modeling Framework

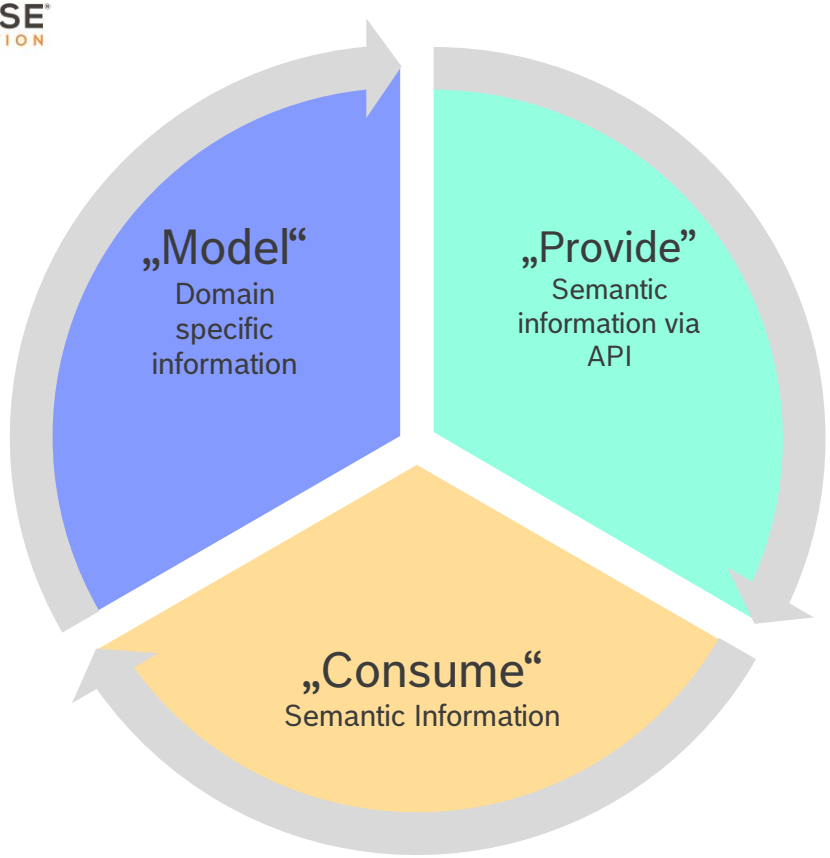
Chris Volk, Eclipse Semantic Modeling Framework

2023 Feb

Agenda

1. Eclipse Semantic Modeling Framework
   - Semantic Aspect Meta Model (SAMM)
   - Aspect Model Editor (AME)
   - Software Development Kits (SDKs)
   - Command Line Interface (CLI)
2. History and current status in Eclipse Foundation
3. ESMF in context
4. Summary and References
5. Discussion

Eclipse Semantic Modeling Framework (ESMF)



“The Eclipse Semantic Modeling Framework provides the means for defining the semantics of different aspects of information aka submodels provided via digital twins and allows to easily provide and consume them in an API.”

Main Features

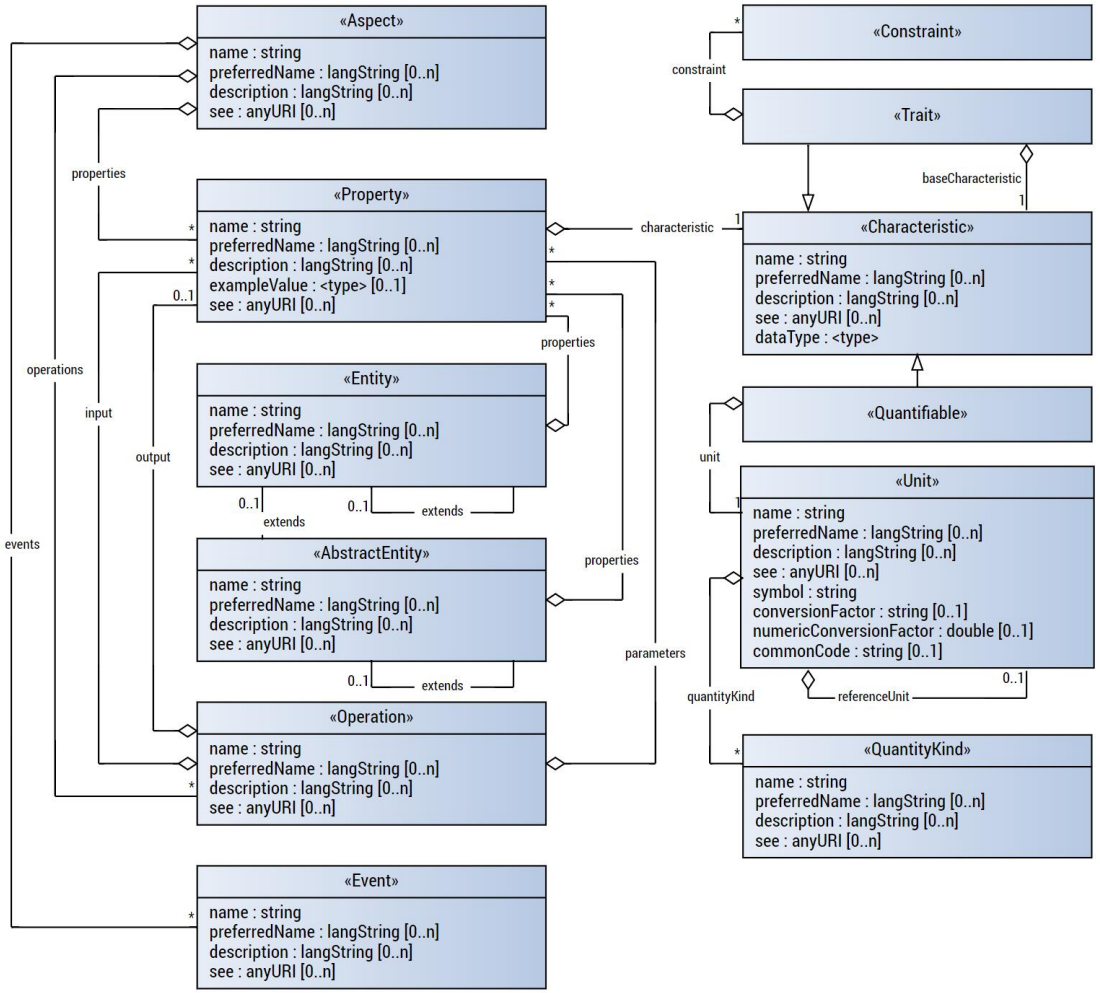
- Semantic Aspect Meta Model (SAMM)
- Visual “Aspect Model Editor” (AME)
- Software Development Kits (Java, JS, PY)
- Command Line Interface (CLI)

Benefits

- Bridge the gap between Domain & Code
- Provide a consistent iterative cycle
  Use Case → Model → API → UI
- Easier than Ontologies yet expressive

https://projects.eclipse.org/projects/dt.esmf

Semantic Aspect Meta Model (SAMM)



"The Semantic Aspect Meta Model allows easy construction of models that semantically describe domain data."

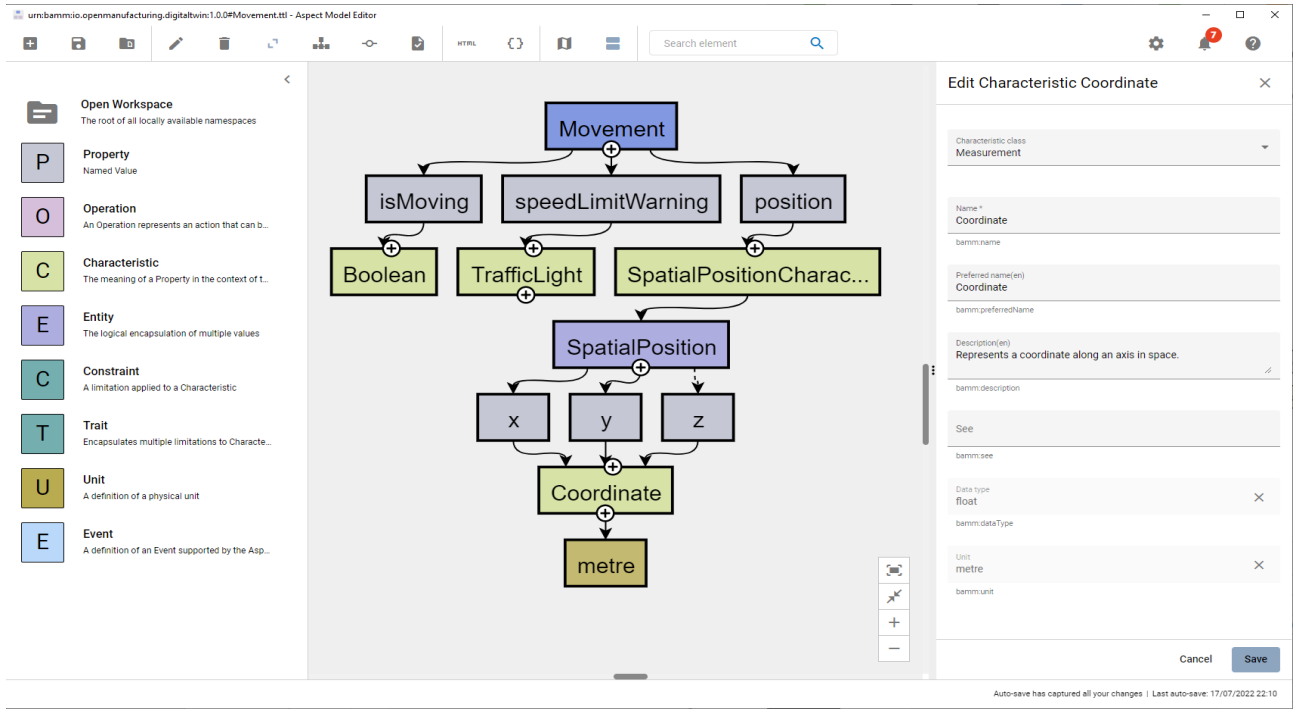
Main Features

- Formalizes semantic information (datatypes, physical units, Constraints, Characteristics, example values, descriptions, references)
- Built upon well established standards such as RDF, XSD (datatypes) and UNECE Unit catalog (physical units)
- Validation via SHACL

Benefits

- Empower Domain Experts to formalize domain knowledge
- Reuse of semantic information

Aspect Model Editor (AME)



"AME makes managing Aspect Models easy."

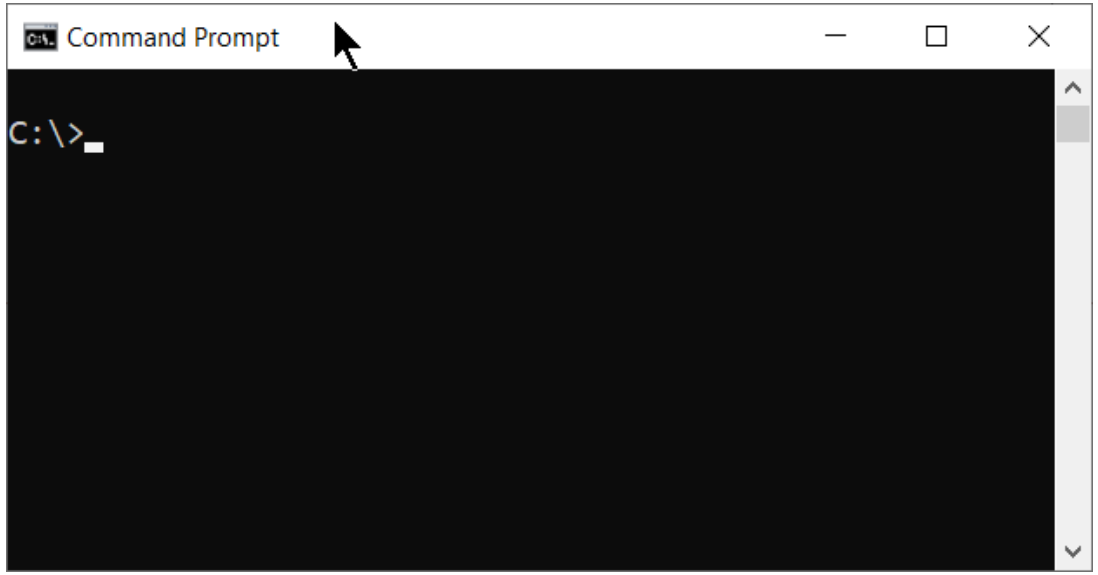
Main Features

- Create, edit & store Aspect Models
- Validate Aspect Models against the Semantic Aspect Meta Model
- Manage Aspect Models: file-based, locally, version-controlled

Benefits

- Empower Domain Experts to formalize domain knowledge
- Learn SAMM intuitively
- Power of visualization allows better comprehension of large models

Command Line Interface (CLI)



“The CLI validates models and transforms them into artifacts to create Aspect APIs.”

Main Features

- Validation
- OpenAPI Spec
- JSON Schema & Sample Payload
- Documentation
- Java Code

Benefits

- Programming language independent set of commands that support the provisioning and consumption of APIs

Software Development Kit Java

```
@GetMapping( "/movement" )
public ResponseEntity<Movement> getMovement() {
   return findAspect( aspectName: "Movement" )
         .map( movementAspectClient::getAspectData )
         .orElseGet( emptyResponse() );
}
```

https://github.com/eclipse-esmf/esmf-sdk

“The SDK Java turns Aspect Models into Java Code – easy to provide and consume“

Main Features

- Load domain specific semantic information from Aspect into code
- SAMM “logic” itself represented in code

Benefits

- Semantic knowledge at the developer’s fingertips
- Reduced effort for providing APIs in java
- Reduced effort for consuming APIs in java

Software Development Kit JS

```
Local
  _aspect = DefaultAspect {_preferredNames: Map(0), _descriptions
    _preferredNames = Map(0) {size: 0}
    _descriptions = Map(0) {size: 0}
    _see = Array(0) []
    _parents = Array(0) []
    _metaModelVersion = "1.0.0"
    _aspectModelUrn = "urn:bamm:io.openmanufacturing:1.0.0#Asp
    _name = "AspectDefault"
    _isAnonymousNode = false
    _properties = Array(2) [DefaultProperty, DefaultProperty]
```

https://github.com/eclipse-esmf/esmf-sdk-js-aspect-model-loader

"Tools for developers to leverage information from an Aspect Model to realize modern frontends in JavaScript"

Main Features

- Smart fully functional Angular components
- Aspect Models at runtime for dynamic UIs

Benefits

- Semantic knowledge accessible to UI developer
- Quickly visualize data from Aspect APIs
- Developer experience: easy, quick, robust

Software Development Kit PY

```
aspect = {DefaultAspect} <sds_aspect_meta_model_python.impl.default_aspect.DefaultAspect object at 0x000002303848C190>
  descriptions = {dict: 1} {'en': 'Aspect for movement information'}
  events = {list: 0} []
  is_collection_aspect = {bool} False
  meta_model_version = {str} '1.0.0'
  name = {str} 'Movement'
  operations = {list: 0} []
  parent_element = {NoneType} None
  preferred_names = {dict: 1} {'en': 'Movement'}
  properties = {list: 3} [<sds_aspect_meta_model_python.impl.default_property.DefaultProperty object at 0x00000230384A... Vie
    0 = {DefaultProperty} <sds_aspect_meta_model_python.impl.default_property.DefaultProperty object at 0x00000230384A0
    1 = {DefaultProperty} <sds_aspect_meta_model_python.impl.default_property.DefaultProperty object at 0x00000230384A0
      characteristic = {DefaultEnumeration} <sds_aspect_meta_model_python.impl.characteristics.default_enumeration.Defau
        data_type = {DefaultScalar} <sds_aspect_meta_model_python.impl.data_types.default_scalar.DefaultScalar object at
        descriptions = {dict: 1} {'en': 'Represents if speed of position change is within specification (green), within tolerance
        meta_model_version = {str} '1.0.0'
        name = {str} 'TrafficLight'
        parent_element = {DefaultProperty} <sds_aspect_meta_model_python.impl.default_property.DefaultProperty object
        preferred_names = {dict: 1} {'en': 'Warning Level'}
        see = {list: 0} []
        urn = {str} 'urn:bamm:io.openmanufacturing.digitaltwin:1.0.0#TrafficLight'
        values = {list: 3} ['green', 'yellow', 'red']
          0 = {str} 'green'
          1 = {str} 'yellow'
          2 = {str} 'red'
          __len__ = {int} 3
        Protected Attributes
      data_type = {DefaultScalar} <sds_aspect_meta_model_python.impl.data_types.default_scalar.DefaultScalar object at 0x0
      descriptions = {dict: 1} {'en': 'Indicates if the speed limit is adhered to.'}
      effective_characteristic = {DefaultEnumeration} <sds_aspect_meta_model_python.impl.characteristics.default_enumerat
      example_value = {NoneType} None
      is_not_in_payload = {bool} False
```

https://github.com/eclipse-esmf/esmf-sdk-py-aspect-model-loader

"Tools for developers to leverage information from an Aspect Model to realize python applications"

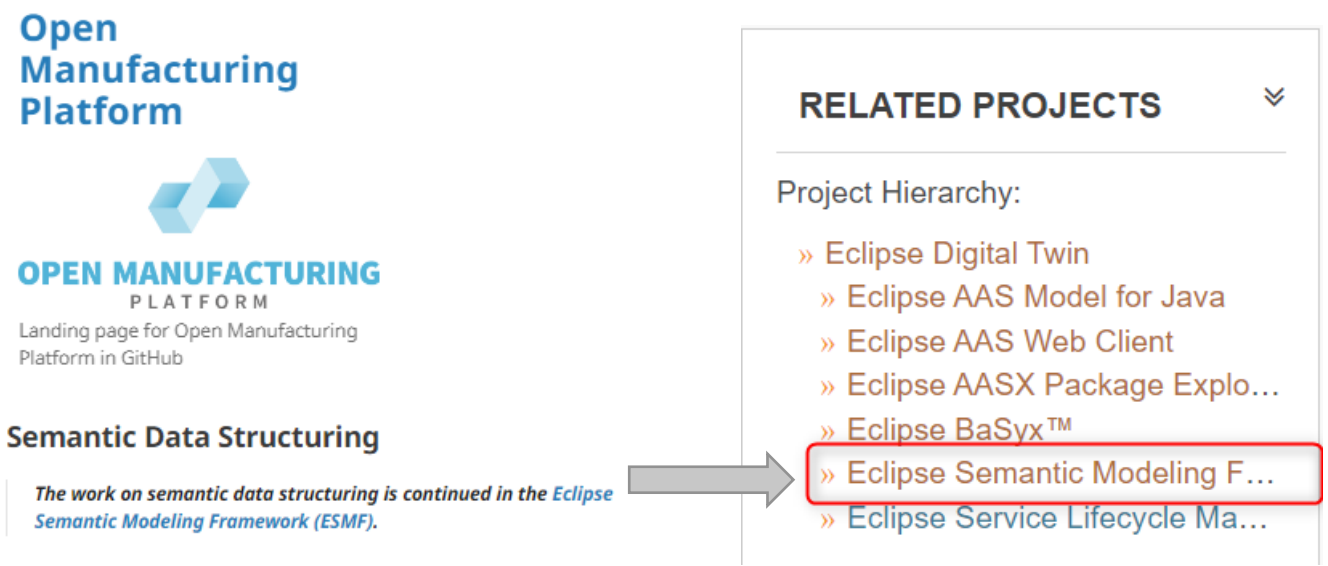
Main Features

- Load and traverse Aspect Models within your python code
- Pandas DataFrame support (planned)

Benefits

- Semantic knowledge natively available in python
- Developer experience: easy, quick, robust

ESMF – History and current status



- Initially developed and released within the OMP semantic data structuring working group (2021-2022), see https://openmanufacturingplatform.github.io/
- December 2022: Transition to Eclipse Foundation into new ESMF with **parent project Eclipse Digital Twin by IDTA**
- 2023 Q1: Renaming & Regular Setup for for further releases

AAS Submodel Templates and SAMM Aspect Models

AAS Submodel Template

SubmodelReference

submodelRef:
(Submodel) (local) [IRI] http://bosch-connected-industry.com/demo/sm/instance/1/0 59D

Submodel

Referable:
idShort: TechnicalData

Identifiable:
idType: IRI
id: http://bosch-connected-industry.com/demo/sm/instan 9908D59DDLOPYR

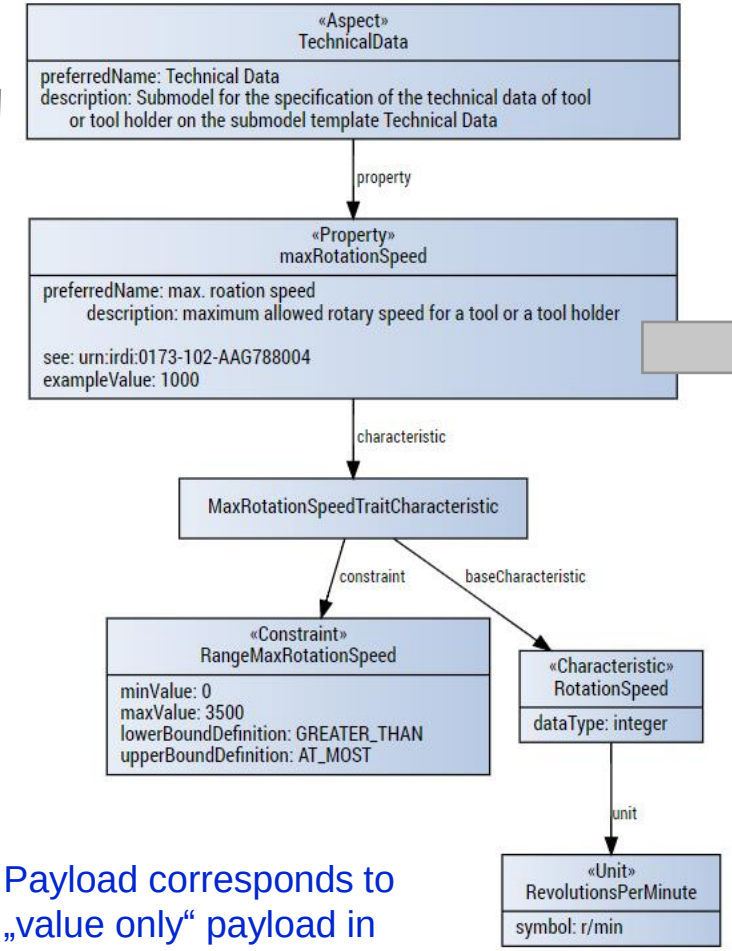
Kind (of model):
kind: Instance

Semantic ID:
semanticId:
(GlobalReference) (no-local) [IRI] urn:bamm:admin-shell-io:tool:TechnicalData:1.0.0#TechnicalData
() (no-local) []

Qualifiable:

HasDataSpecification (Reference):

SAMM Aspect Model



ECLASS

Preferred name	max. permissible speed of rotation
Definition	maximum allowed rotary speed for a tool or a tool holder
IRDI	0173-1#02-AAG788#004
Data type	INTEGER_MEASURE

Payload corresponds to „value only“ payload in AAS Submodel

ESMF in the context of Catena-X

Catena-X
Automotive Network

Vision & Goals

Benefits

Semantic Hub

Semantic models are used to structure data. As a concept, they make it possible to give fundamental meaning to data and their relationships. In the context of data modeling and data organization they realize the development of applications as well as the maintenance of data consistency.

In Catena-X, semantic data models are provided in a suitable publication system, the so-called **Semantic Hub**. For data or service providers, the Semantic Hub presents a foundation for the use and reuse of semantic models. For this reason, the publication of a semantic data model takes place under the specification of a version. This supports transparency as well as control over all semantic models and their release status.

Last revision: 24-05-22

- Catena-X (Eclipse Tractus X) includes SAMM in the "Catena-X Standard Library"
- used for semantic models within the Semantic Hub

Summary & References

"The Eclipse Semantic Modeling Framework provides the means for defining the semantics of different aspects of information aka submodels provided via digital twins and allows to easily provide and consume them in an API."

- Parent project: https://projects.eclipse.org/projects/dt
- Project Page: https://projects.eclipse.org/projects/dt.esmf
- Github home: https://github.com/eclipse-esmf/
- Forum: https://www.eclipse.org/forums/index.php/f/617/
- Current landing page (archived): https://openmanufacturingplatform.github.io/
- New landing page: TBD (designated: https://eclipse-esmf.github.io/ or similar)
- Mailing list: https://accounts.eclipse.org/mailing-list/esmf-dev