



Smart Modeling

Capturing the Internet of Things via System Models

(Questions and maybe some answers from the largest project of the EU)

Géza Kulcsár (IncQuery Labs)

geza.kulcsar@incquerylabs.com

INCQUERYLABS

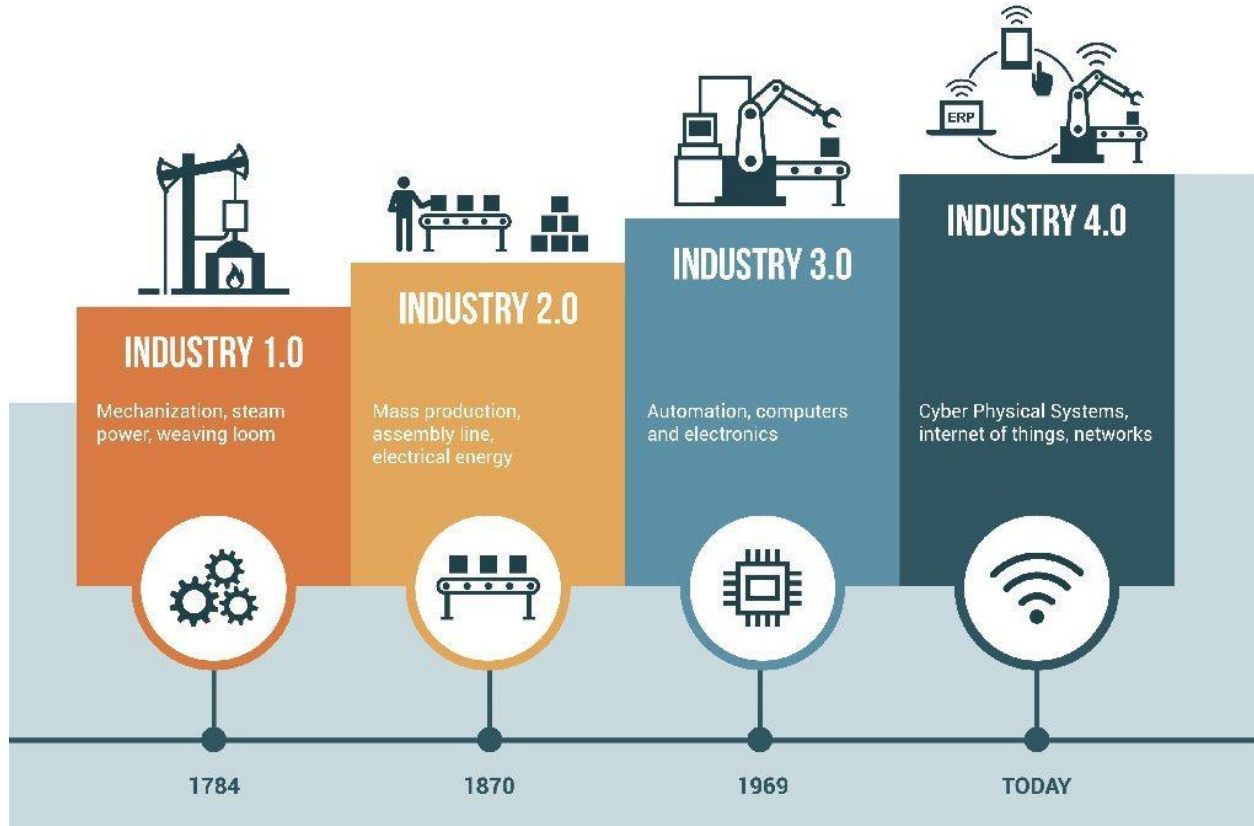


Motivation: Can we get *both* structured and dynamic in Industry 4.0?



<https://www.iotworldtoday.com/2020/02/20/bringing-the-smart-factory-vision-to-life/> (Getty Images)

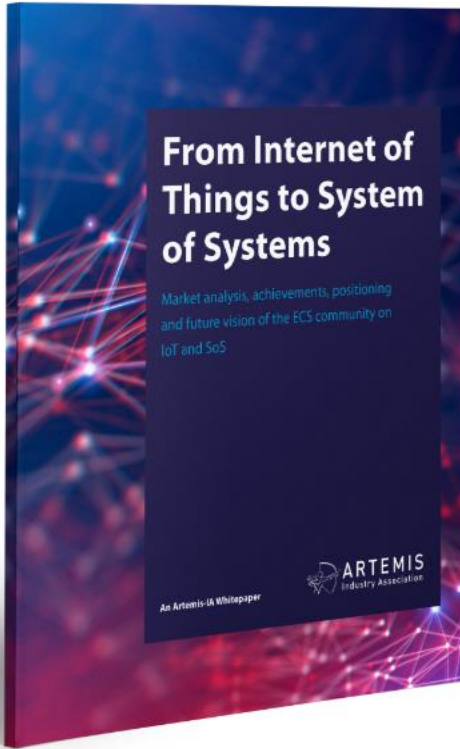
„A spectre is haunting Europe...”



Internet of Things: the perfect hype

- What is a *thing*?
- Despite some concretization experiments, primarily, still:
„[the word thing is] used to refer in an approximate way to an object or to avoid naming it” (Cambridge Dictionary)
- Also, *internetting* means any kind of interconnection since at least 1849
- Thus, the term Internet of Things creates a perfect hype: it has an intuitive meaning, but concretely, it can be anything

„Rose is a rose is a rose is a rose”



*„The Internet of Things (IoT) is the interdisciplinary solution adopted to integrate these heterogeneous and distributed objects in a single **system of systems (SoS)**, efficiently managing the collection and processing of vast amount of data, generating added-value services and applications to achieve common goals.”*

IIoTSoSaaS: a history of abbreviations

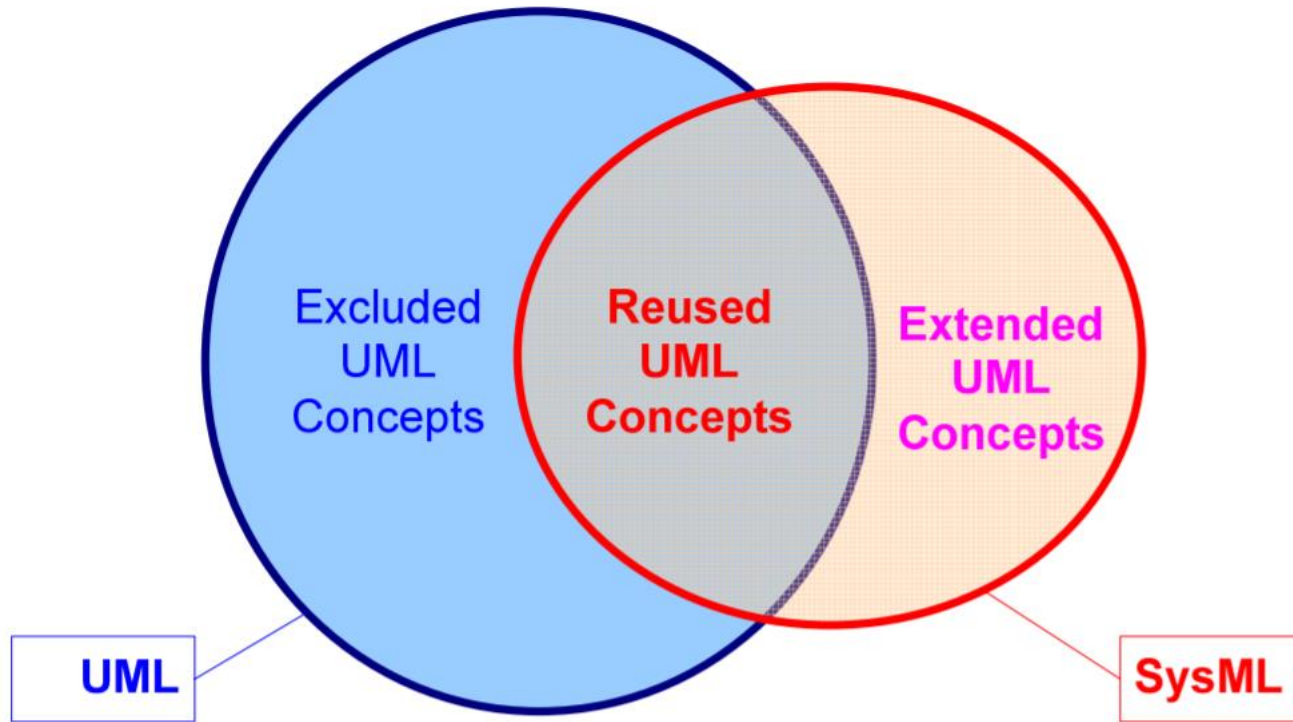
- The first IoT device: a remote-controlled Coke machine at CMU *modified by a student!*
- The term itself was coined by Kevin Ashton (MIT) in 1999
- Three stages:
 - *Monolithic (until ~2010):* interconnected devices, but a low level of automation
 - *Cloud-based (until ~2016 - AaaS – Anything as a Service):* PaaS, SaaS, IaaS, Big Data, IoT value chain, ...
 - *Embedded Intelligence (now):* the real-life sci-fi



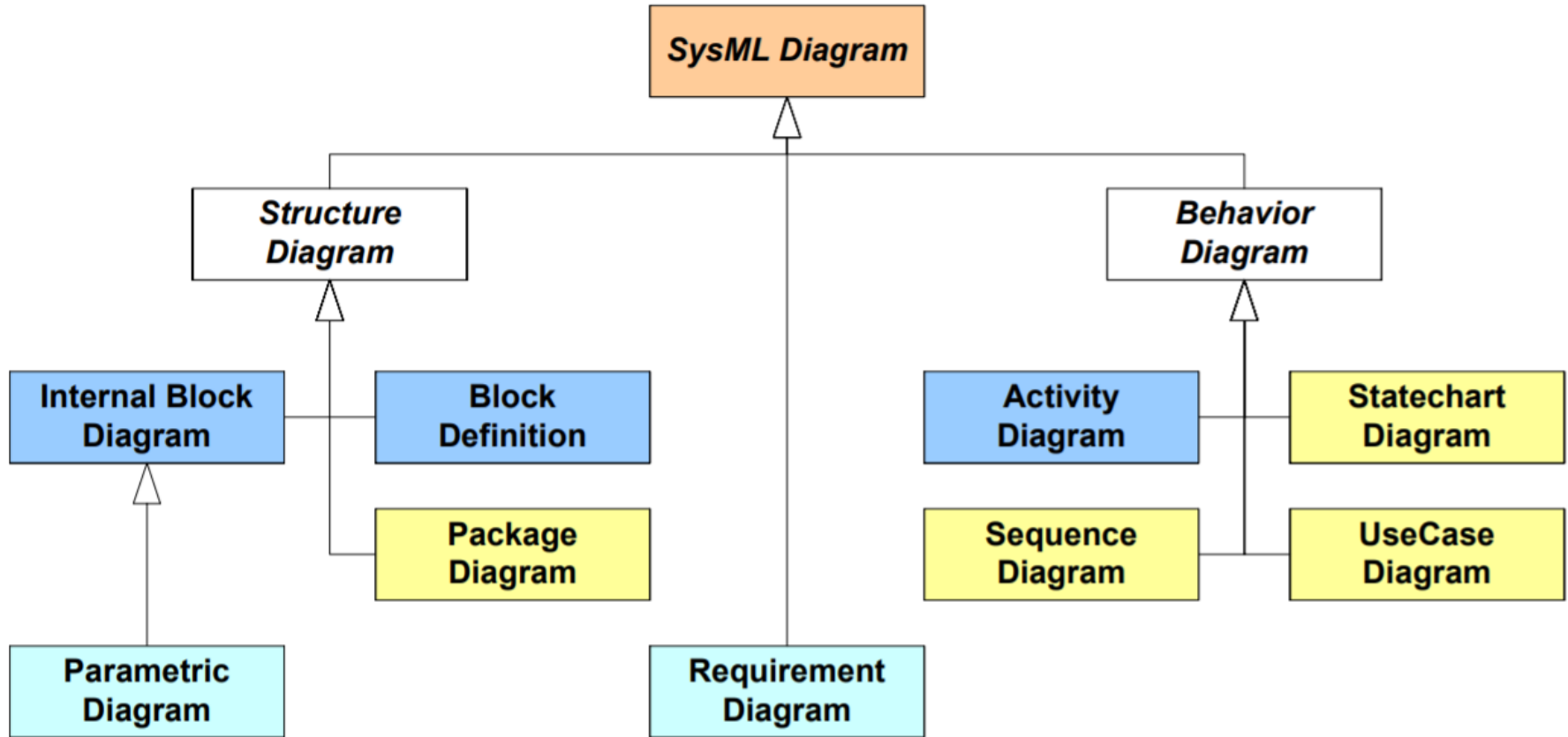
Model-Based Anything: a parallel story

- Around the same time (1980s), as software systems became more complex than a few punch cards, a new religion arose: *modeling* will help to tame complexity
- *UML*: modeling the universe using a single set of principles (an OMG standard since 1997)
- However, it has been observed early on that this single universal (meta-)language will not be expressive enough for every *domain*
- Enter *profiles!*

Systems engineering: a new domain?



SysML: yet another UML extension?



But what is a system?

- SysML is a wide-spread language and standard for systems modeling
 - Currently established version: **1.6**, the release of version **2**, constituting a major update, is planned in 2021
- Originally conceived for single, monolithic systems (cf. *Monolithic stage*), but it started to embrace complex and dynamic scenarios
- Challenge: will SysML accomodate *systems of systems* (SoS)?

Services at your service

- At a first glance, monolithic, i.e., object-oriented modeling results in a spectacular philosophical clash with lot/SoS
 - Due to the assumption that *structure = objects*
- But there's a chance if we learn to *change perspectives* (a handy skill in the present crisis anyway)



Services at your service

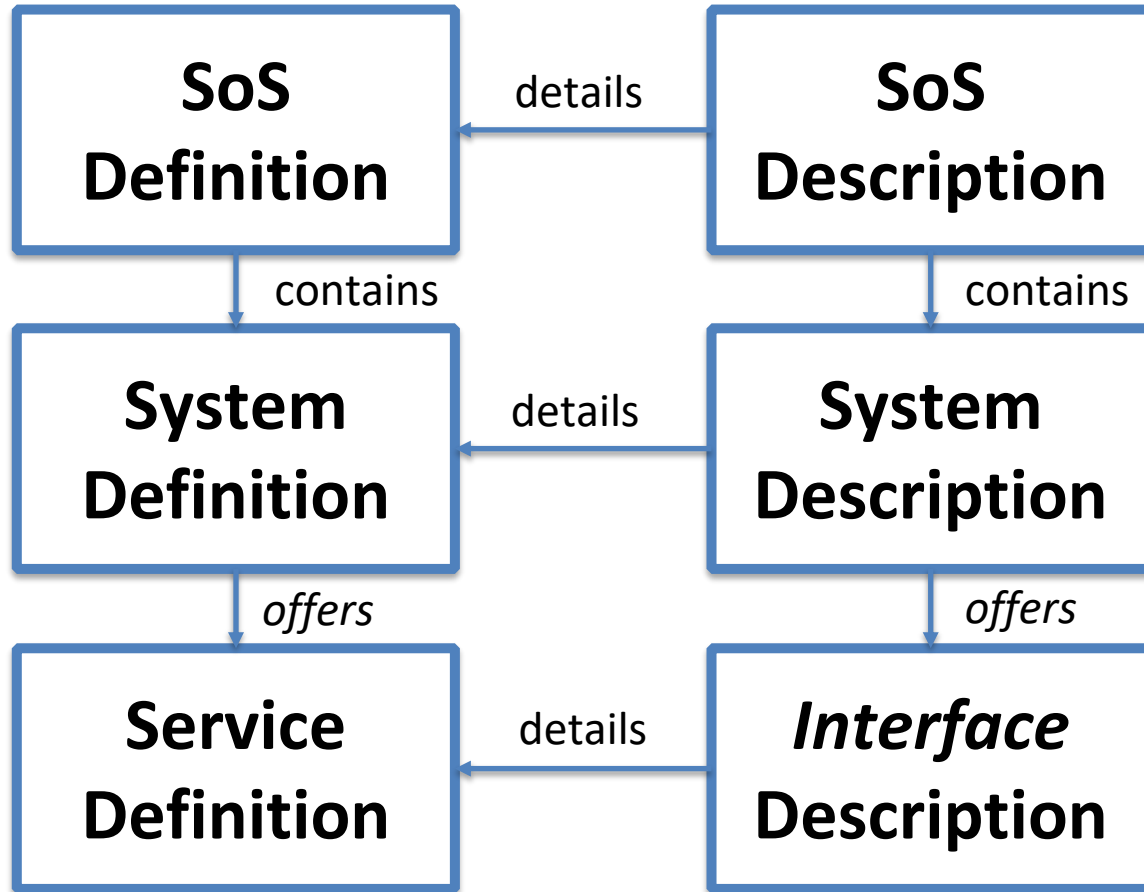
- This is the promise of the *service-oriented architecture* paradigm
 - I.e., in AaaS, we shift the focus from Anything to Service
- In order for a picture to become effective, we have to consider projection lines, but do not have to achieve a single vanishing point!

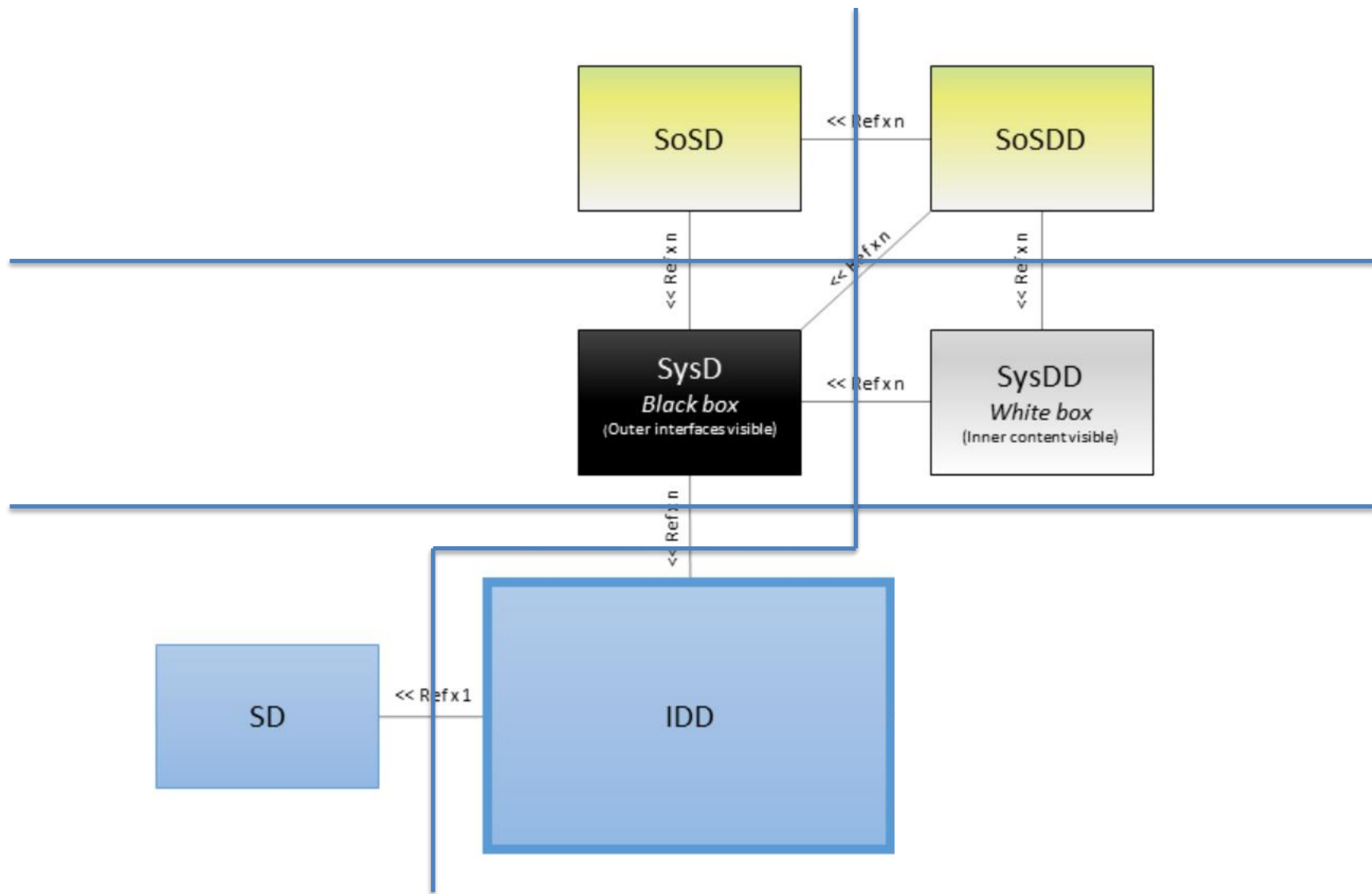


SysML concepts and challenges in Arrowhead Tools



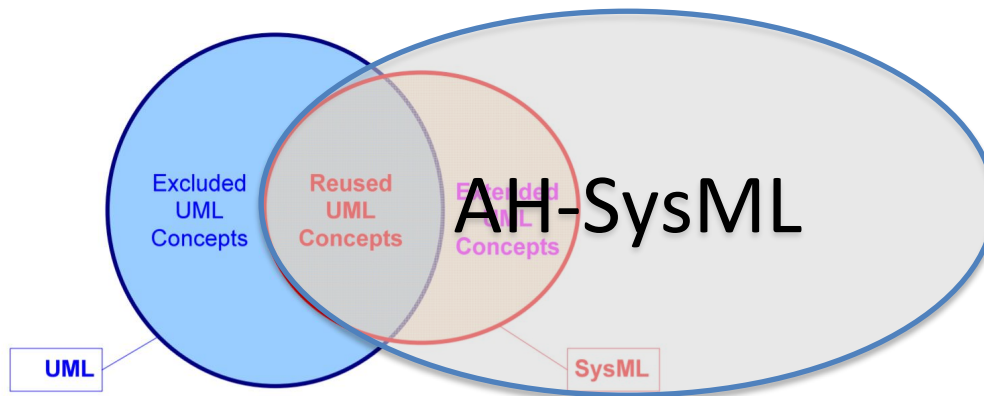
SOA à la Arrowhead





Approach: Arrowhead SysML Profile

- Providing structural models...
- ...upon well-founded concepts...
- ...but without the rigidity of monolithic systems!



Which facets of MBSE do we consider?

- Domain-specific models
 - In particular, a SysML v1.6 *profile* has been established and there are ongoing investigations for a v2 *library*
- Documentation
 - A good model is (partly) documenting itself, but we can also use it for generating textual documents
- Code generation
 - A good model can be the basis for actually deployed systems, we are investigating OpenAPI as a common ground
- Toolchain integration
 - One of the main goals of AHT, considered on two levels: *tool* and *model interoperability*

The Shape of Things to Come: SysML 2

```
/** Mandatory Services and Systems */
package AHFCoreLib {
  import AHFProfileLib::*;
  import ScalarValues::*;

  port def ServiceDiscovery :> SD{
    // The functionalities as Requests (Operations) cannot be defined yet
    // We could consider using flows to designate the functionalities
  }

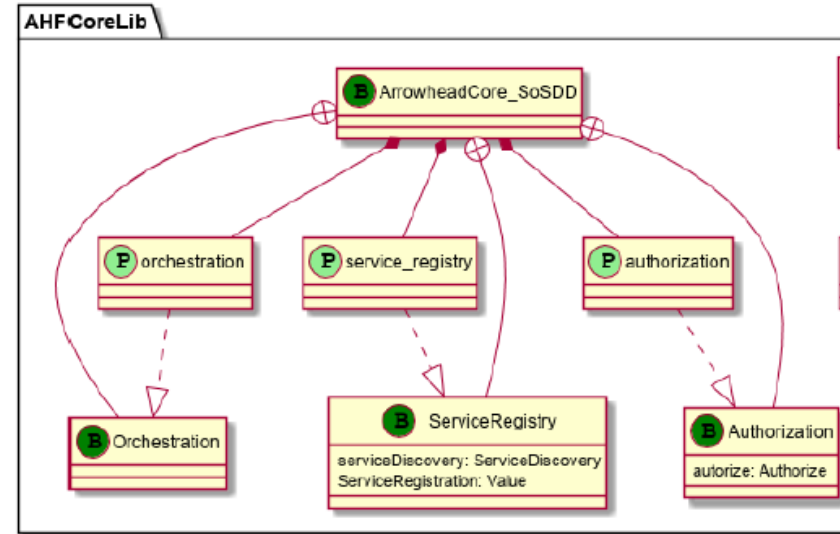
  port def Authorize :> SD{
    publickey:String; // just as examples
  }

  block ArrowheadCore_SoSDD :> LocalCloud_SoSDD{
    block ServiceRegistry :> SysDD {
      port serviceDiscovery:ServiceDiscovery subsets services;
      ServiceRegistration subsets ServiceMethod;
    }

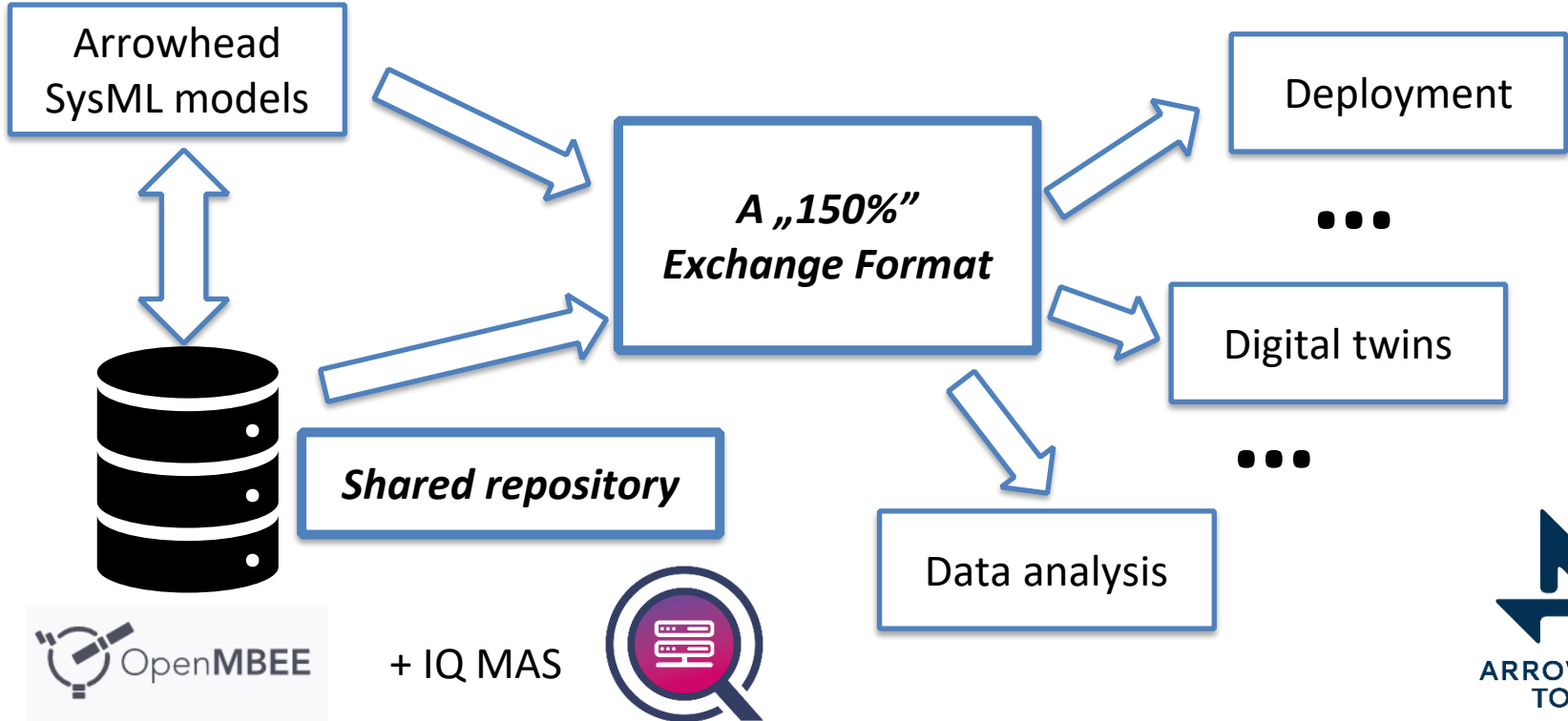
    block Authorization :> SysDD {
      port authorize:Authorize subsets services;
    }

    block Orchestration :> SysDD; // short for now

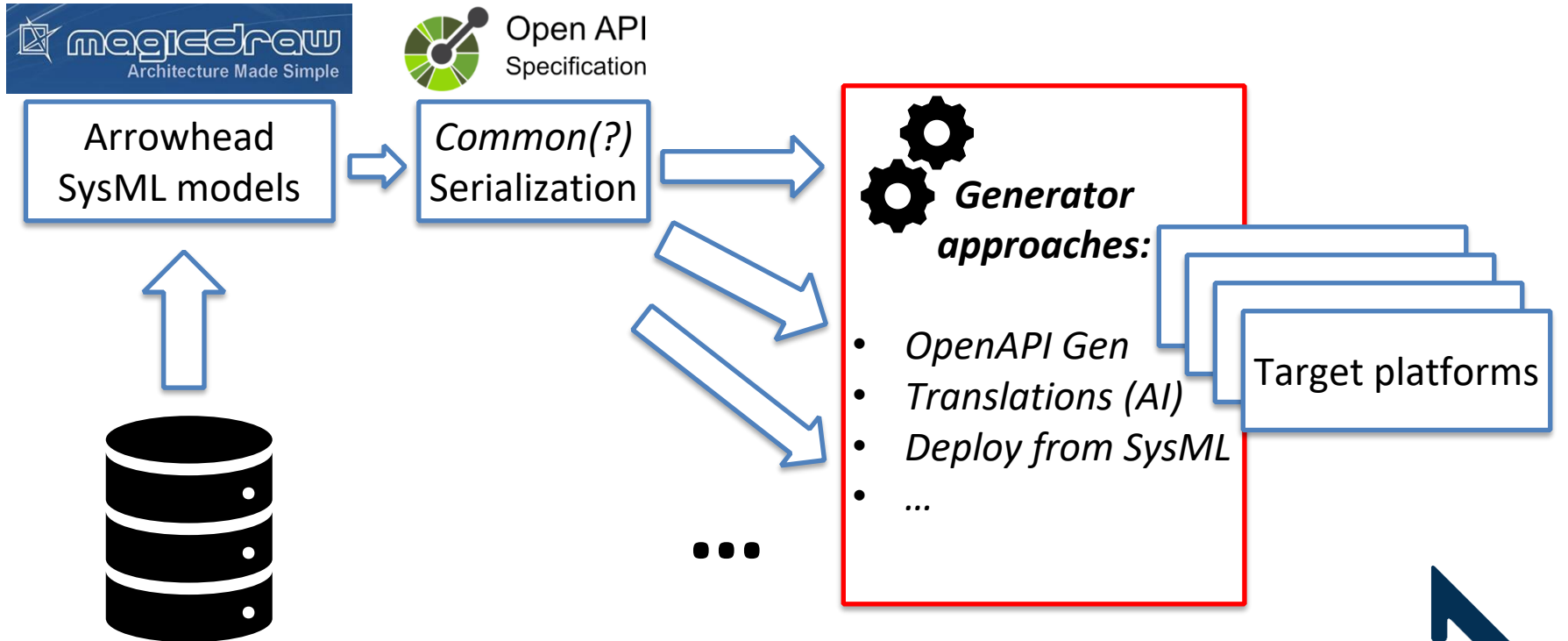
    part service_registry:ServiceRegistry subsets systems;
    part authorization:Authorization subsets systems;
    part orchestration:Orchestration subsets systems;
  }
}
```



Challenge: Tool integration

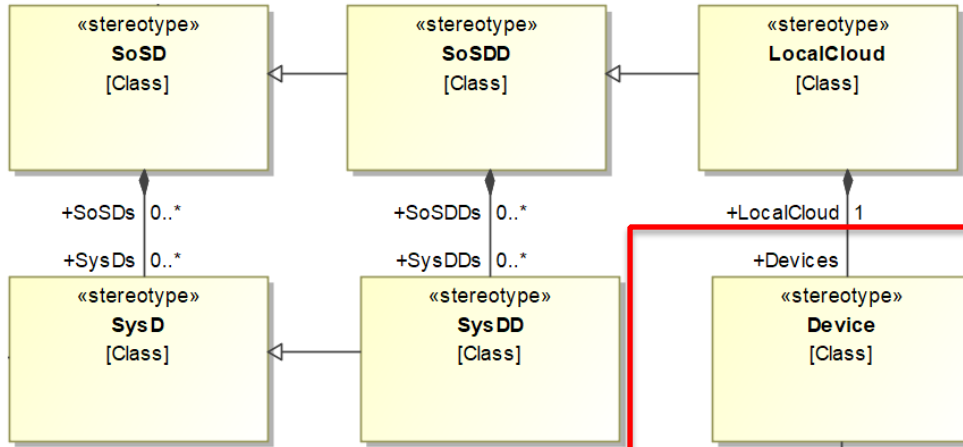


Challenge: Code generation



Challenge: Model integration

Systems modeling realm



```
using org.eclipse.vorto.Location; 1.0.0
using org.eclipse.vorto.Temperature; 1.0.0
```

```
infomodel RaspberryPi{
functionblocks {
```

```
temperature as Temperature
location as Location|
```

```
}
```

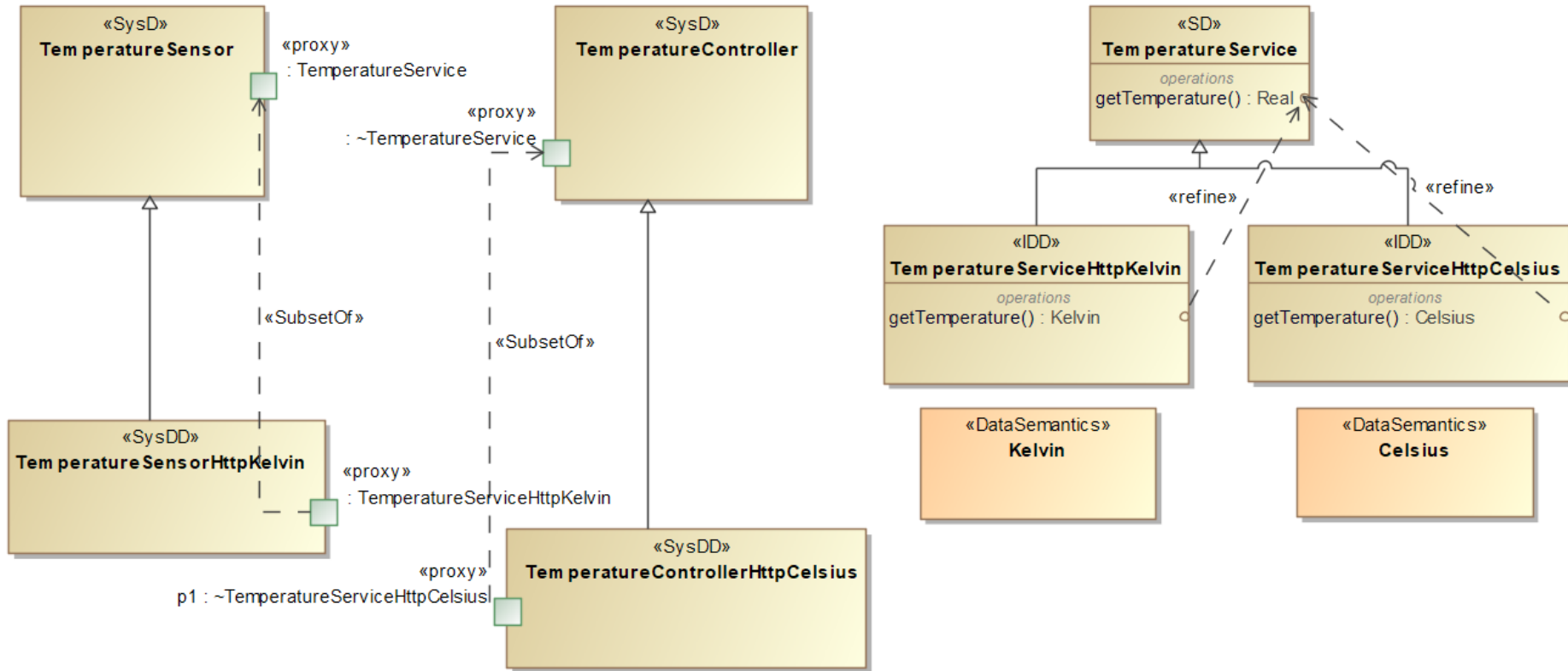
Eclipse Vorto

Many standards:

- STEP (ISO 10303)
-

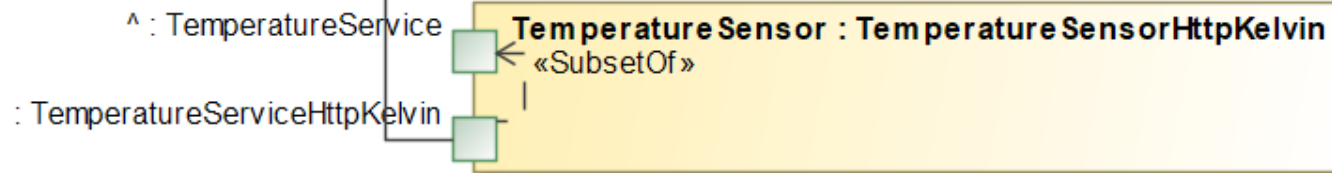
Deployment modeling realm

Challenge: Data semantics



Challenge: Data semantics

- Such a connector represents a *desired* connection, which is here prevented by data mismatch
- Models allow for an *early detection* of such situations and even for inducing an *advanced orchestration* involving conversion



The Smart Modeling Experience

A Model-Based Arrowhead Toolchain Prototype



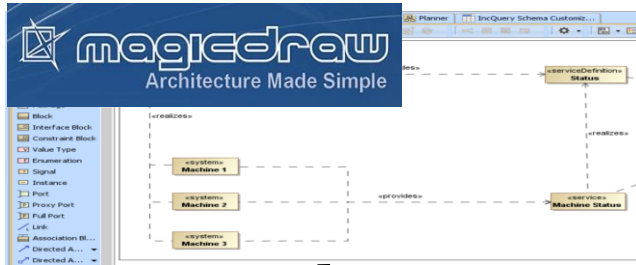
The Arrowhead toolchain vision

- The *Tools* in Arrowhead Tools
- „A tool supports SoS engineering activities; it can be a *design-time* or a *run-time* tool, depending on its place in the engineering process.”
- „A *toolchain* is a collection of tools and their interfaces; it can be design-time, run-time or *both*.”

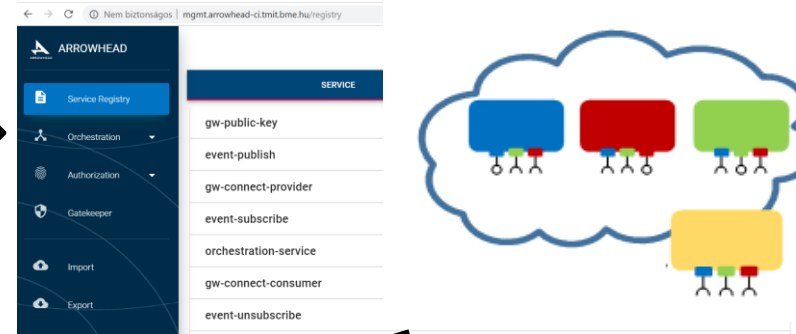


A Model-Driven SoS Toolchain

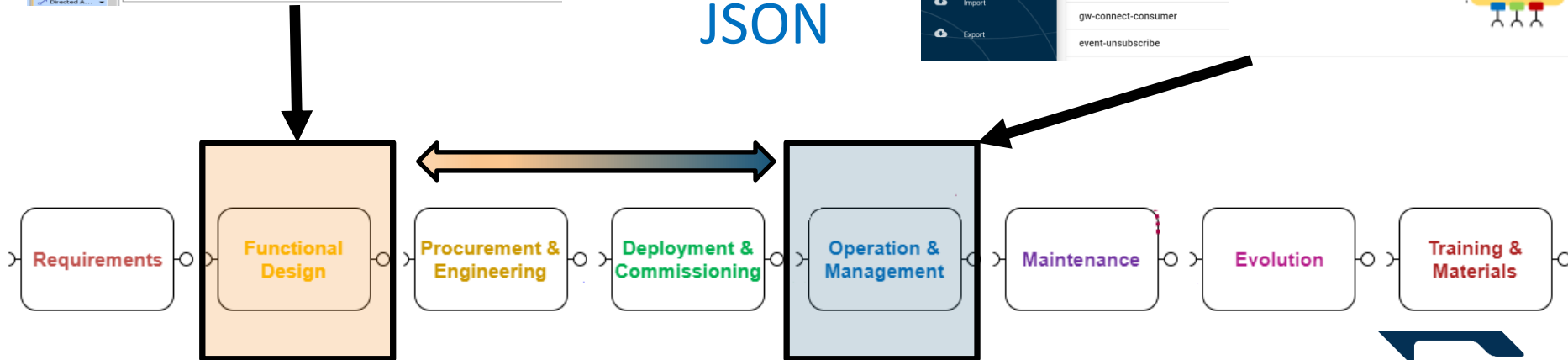
SysML models



AH Management Tool



AHX
JSON



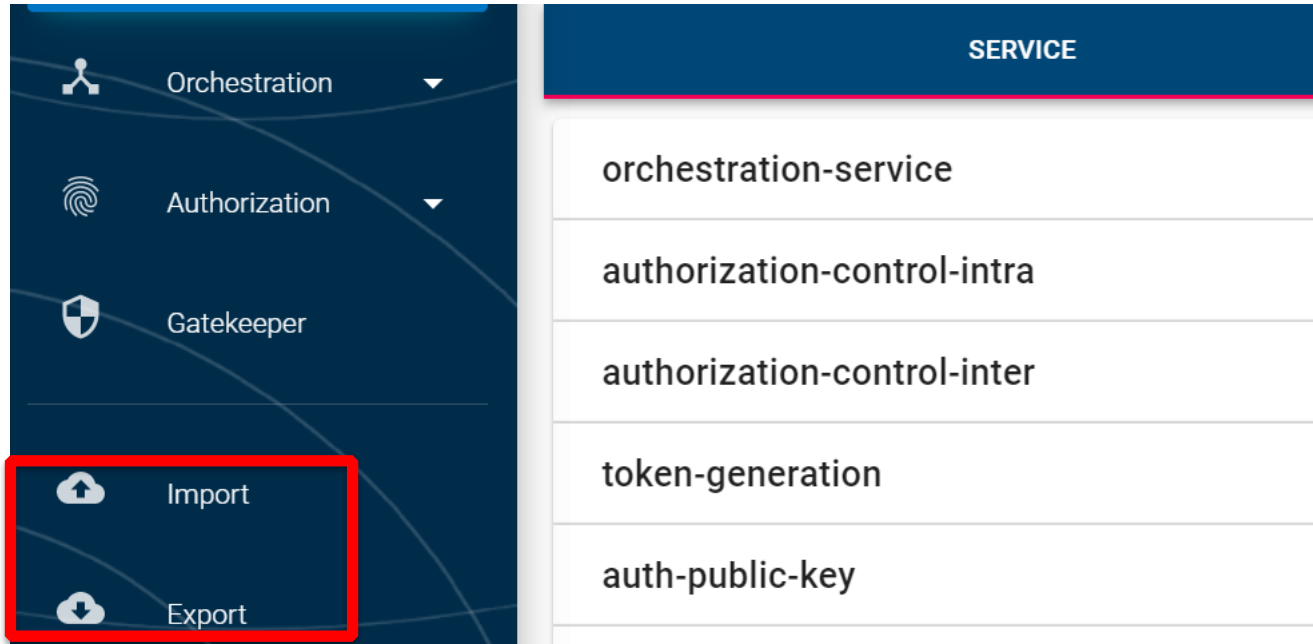
Interoperability via Serialization: AHX

- ArrowHead eXchange is a simple textual descriptor of an Arrowhead *local cloud* (i.e, SoS) in the simple and popular JSON format
- AHX is tailored to fit both the system model structure and the information needed by the management tool in running time

```
"serviceDefinition": "SenML IndoorData",
"provider": {
  "systemName": "IndoorSensor1",
  "address": "127.8.1.2",
  "port": 333
},
"secure": "NOT_SECURE",
"interfaces": [
  "HTTP-INSECURE-JSON"
],
"metadata": {}
"consumer": "Forecaster1",
"serviceDefinition": "SenML IndoorData",
"providers": [
  "IndoorSensor1"
],
"interfaces": [
  "HTTP-INSECURE-JSON"
]
```

Run-Time Management in Arrowhead

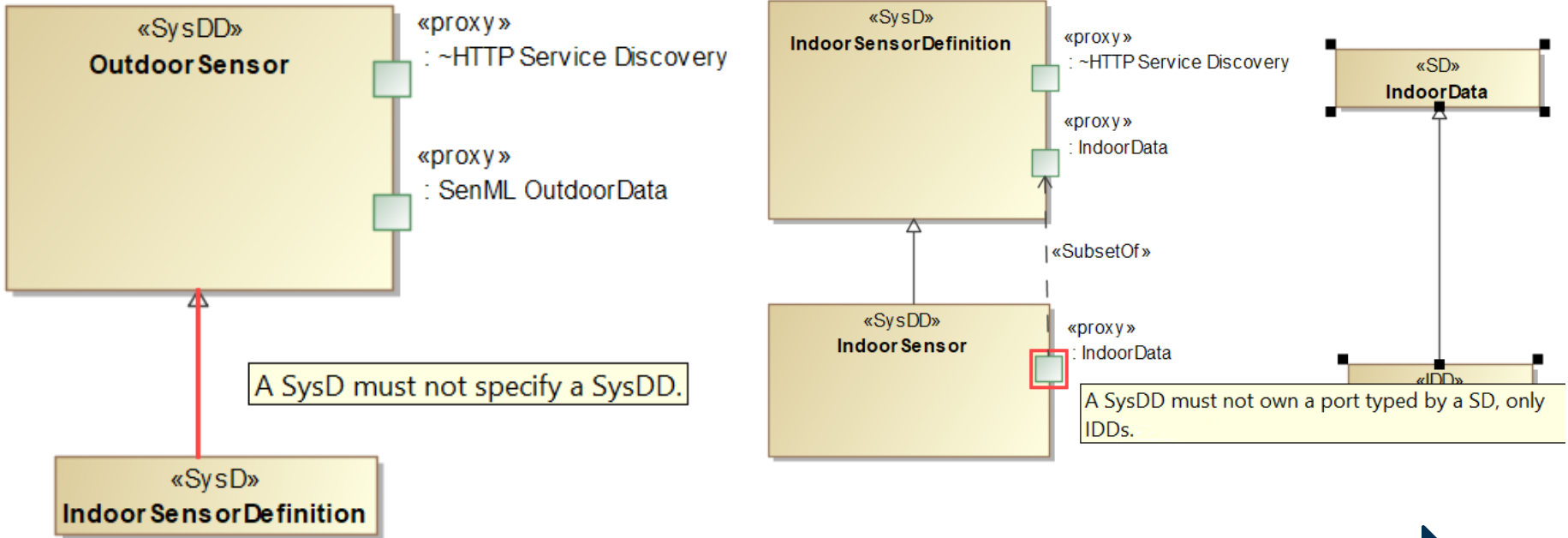
- The Arrowhead Management Tool is an easy-to-use browser tool for interacting with a running Arrowhead local cloud
- It is also an ideal place to import AHX JSON files for populating the system and service list based on the model



The screenshot displays the Arrowhead Management Tool interface. On the left is a dark blue sidebar menu with icons and labels for 'Orchestration', 'Authorization', 'Gatekeeper', 'Import', and 'Export'. The 'Import' and 'Export' options are highlighted with a red rectangular box. On the right is a table with a dark blue header labeled 'SERVICE'. The table contains a list of service names: 'orchestration-service', 'authorization-control-intra', 'authorization-control-inter', 'token-generation', and 'auth-public-key'.

SERVICE
orchestration-service
authorization-control-intra
authorization-control-inter
token-generation
auth-public-key

Validation: Models are Alive



Automated Documentation

- The following table is generated from MagicDraw as well!

Types

Name	Documentation
SysD	The System Description (SysD) describes the outer interfaces of a system kind by defining its services it provides. In this profile, SysDs are blocks having (proxy) ports on them, representing Service Descriptions (SD) typing the ports (cf. also the description of the SD stereotype).
SysDD	The System Design Description (SysDD) provides a white-box description, describing which SysDs) and which components are used to implement the system. In this profile, a SysDD might be provided, those should reside inside a corresponding SysDD. The ports on a SysDD typically can be expressed by the SubsetOf dependency.
IDD	The Interface Design Description (IDD) provides implementational and technical details on how SysDD ports should be typed by IDs (conforming to the corresponding SysD-SysDD relation). Parameters can be modeled as operations, whose behavioral specifications then can be placed in

Thank you for your attention!

The research has received funding from the EU ECSEL JU under the H2020 Framework Programme, JU grant nr. 826452 (Arrowhead Tools project, <https://www.arrowhead.eu>) and from the partners' national funding authorities.

Project no. 2019-2.1.3-NEMZ_ECSEL-2019-00003 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the 2019-2.1.3-NEMZ_ECSEL funding scheme.