Semantic Interoperability Reasoning for Process Design Data

Michael Wiedau (RWTH Aachen University)
Manfred Theißen (AixCAPE e.V.),

ISO TC 184 / SC 4

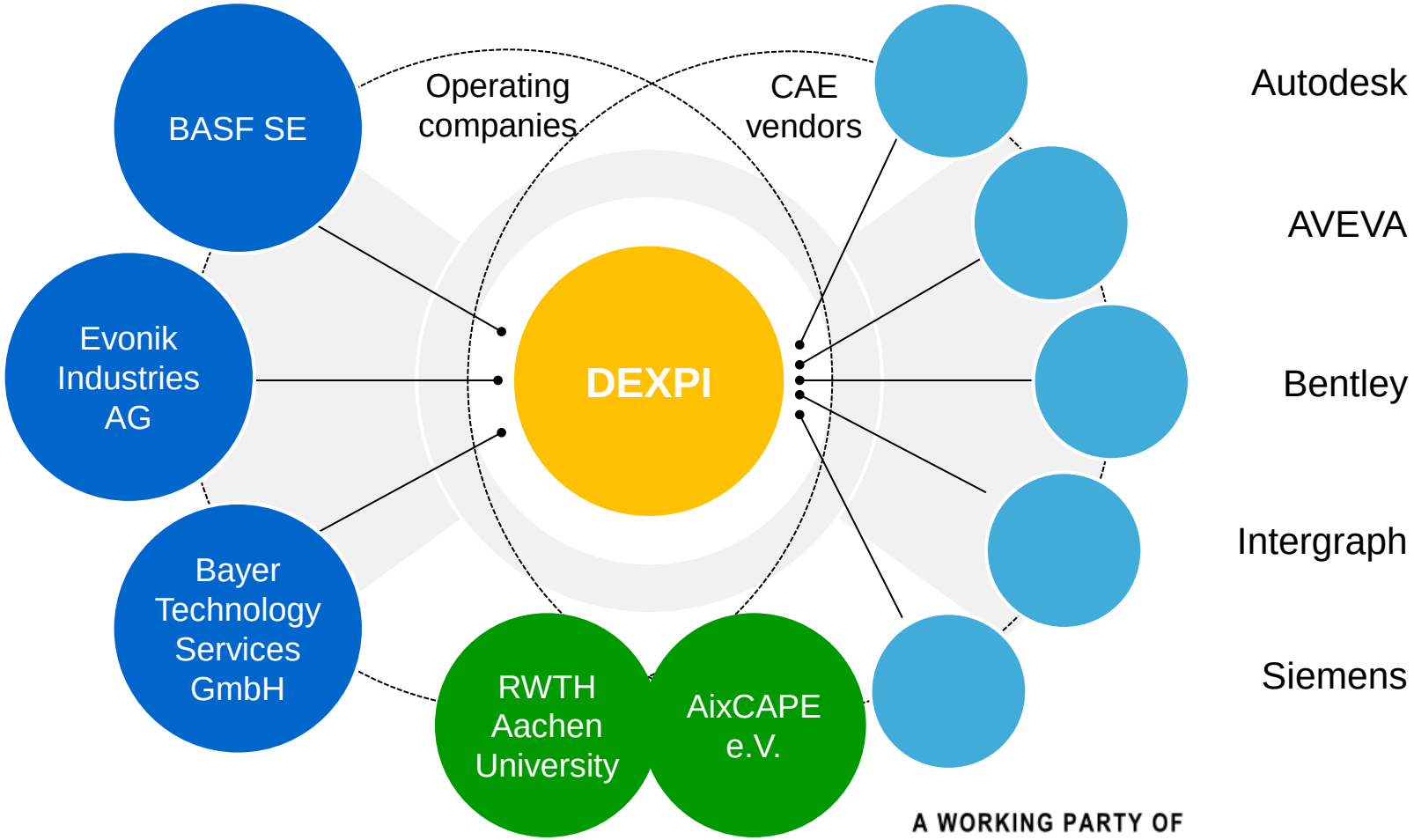
30th of July 2015

Content

- DEXPI
- Semantic interoperability – mapping problem
- 7 steps to verification
- Application of the verification method
- Conclusion & outlook

DEXPI Group Overview

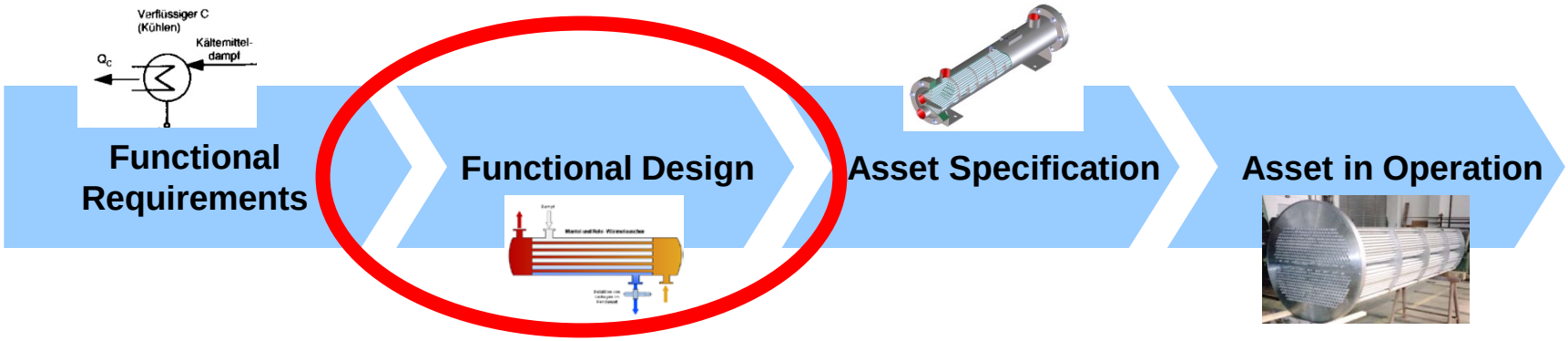
- DEXPI = Data Exchange for the Process Industry



A WORKING PARTY OF
PROCESSNET
AN INITIATIVE OF DECHEMA AND VDI-GVC

DEXPI – Common Goals

- **General standard** for the process industry based on ISO 15926, implemented in the next CAE software generation
- **Input from process industry:** Open and international information model for the entire plant lifecycle

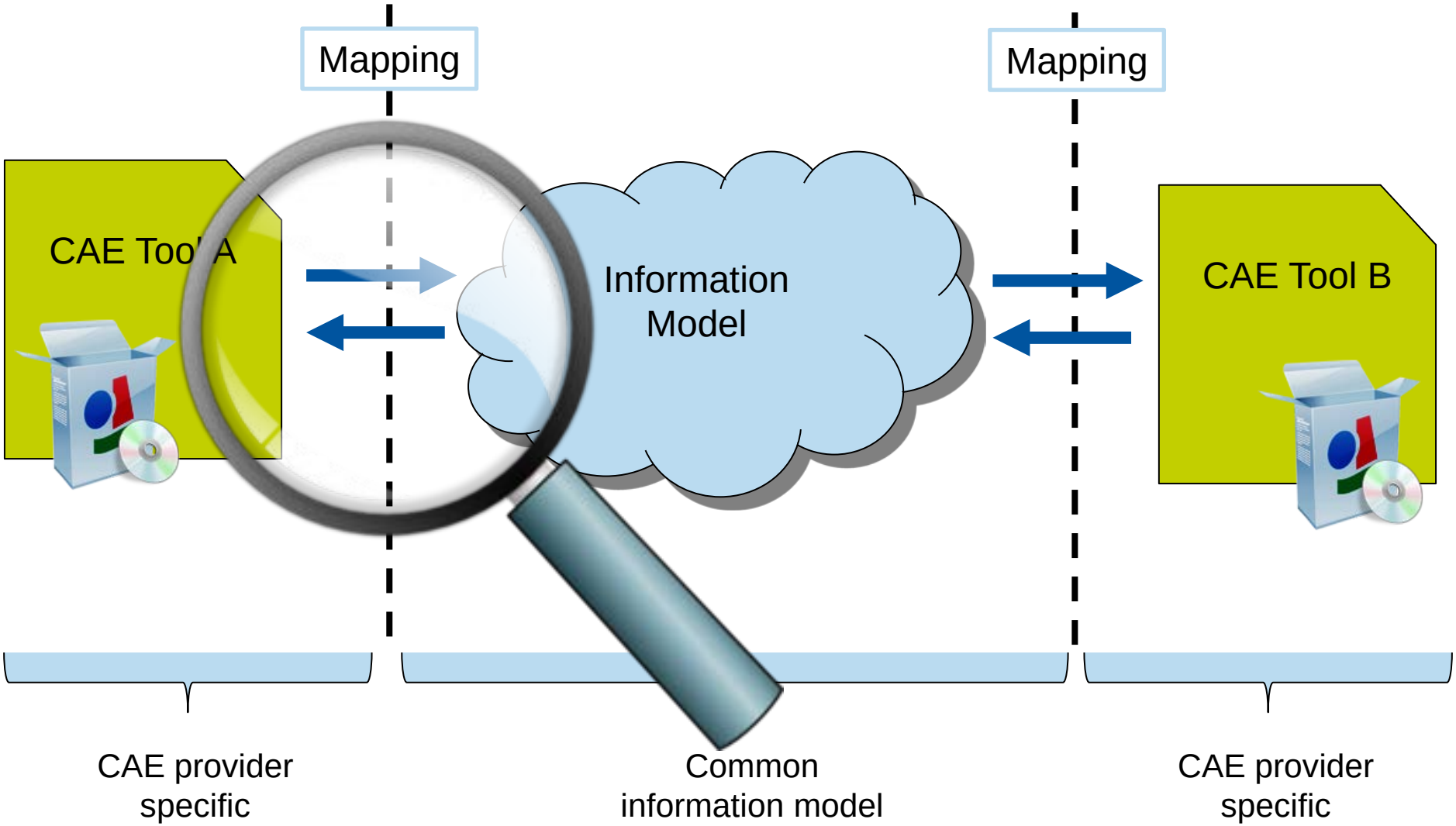


- **Input from the CAE vendors:**
  - General exchange standard for graphics
  - export and import functions based on the new information model and graphics standard

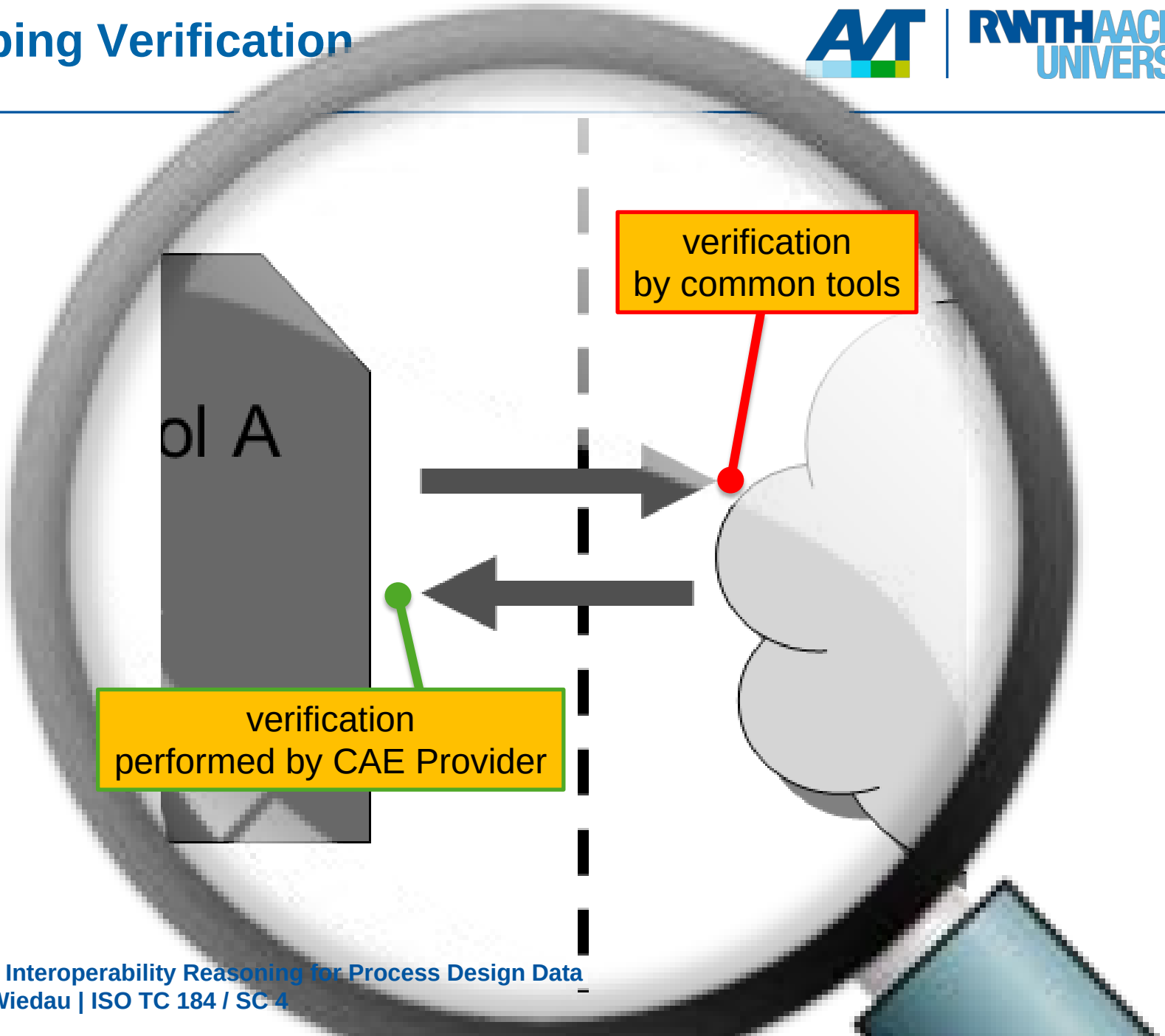
Data Exchange
→ (Semantic) Interoperability

What is the problem?

Semantic interoperability



Mapping Verification



Problem: Common Verification needed

- Requirements for a common verification method:
  - Method must be publicly accessible → Web-Verificator
  - Method must be well known → Presentation today, etc
  - Method must rely on a common mapping model → ISO 15926
  - Method must use open transport layer → Proteus XML

- **How does the Method work?**

AACHENER VERFAHRENS-TECHNIK | RWTH AACHEN UNIVERSITY

Method description

The seven steps to verification output

Step 1 : Sample Input

- Load / Receive Input file
- File should be generally checked
  - Can it be loaded?
  - Does it have the right content-format?
  - Is it possible to load it into the parsing engine?

Step 2: Convert to triples

- Convert to RDF-triples

  → basic of semantic technologies
- Extract all necessary information for verification
- Preserve information for later inference of verification results

Step 3 : Add information model

- Information model:
  - General domain-specific and discipline-specific knowledge
  - Stored as RDF-triples
  - Extends knowledge about elements provided in the input file

Step 4 : Add semantic rules

- Rules that describe the verification
- Written in a commonly used rule language
- Rules lead to positive or negative verification results

Step 5 : Reasoning

- Usage of a **semantic rule reasoning** engine
- Several **reasoning engines** on the market:
  - Commercial (Bossam, …)
  - Free to use (OpenCyc, KAON2, …)
  - Open Source (Flora-2, Jena, …)
- Several **reasoning algorithms** developed all over the world

Step 6 : Receiving Results

- Reasoning engine adds reasoning results as triples to the triple store
- Relevant result-triples have to be selected

Step 7 : Formatted Output

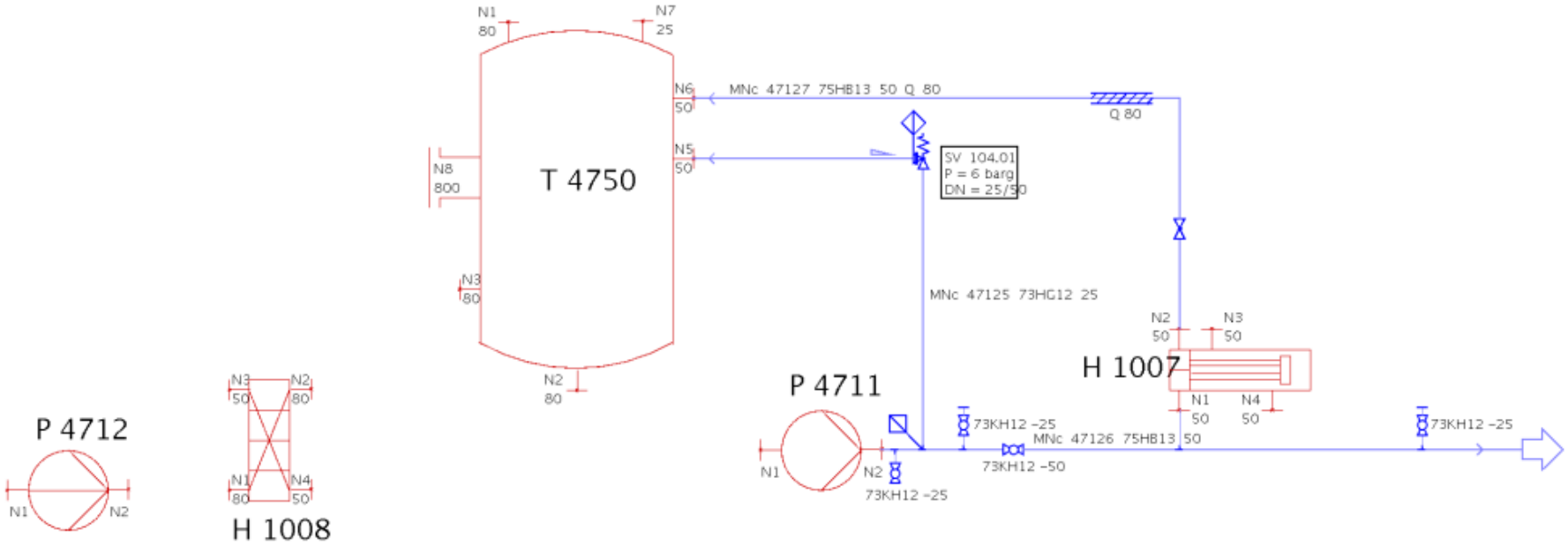
- Provide the output in
  - 1. a human readable form ← main focus!
  - 2. a computer readable form
- Provide the user with all preserved **information to be able to solve the problems** that have been derived from the verification process

Application of the method

Verification of exchange of
Piping and Instrumentation Diagrams (P&IDs)

Example file: the DEXPI P&ID

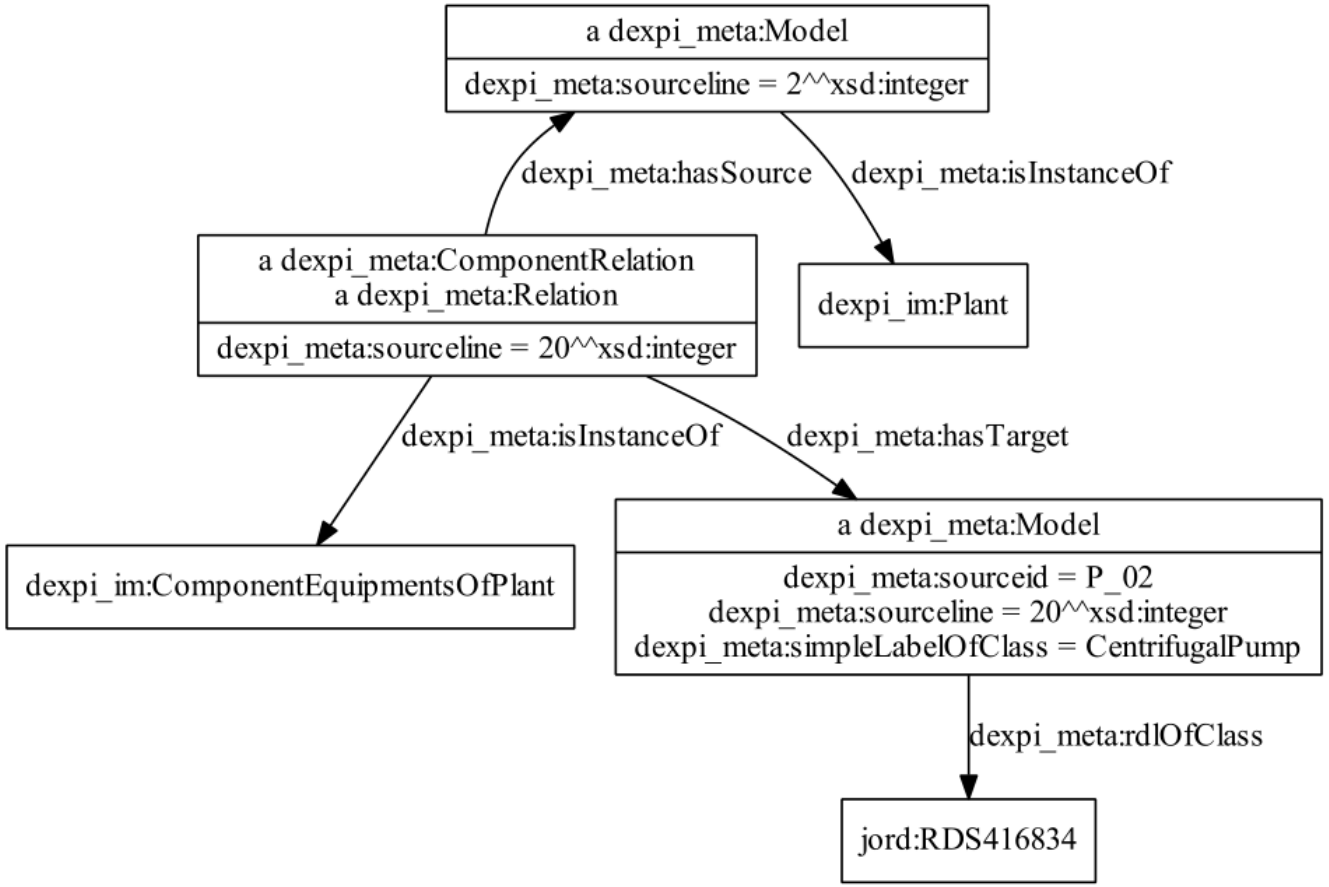
Sample Input → Convert to triples → Add information model → Add semantic rules → Reasoning → Results → Formatted Output



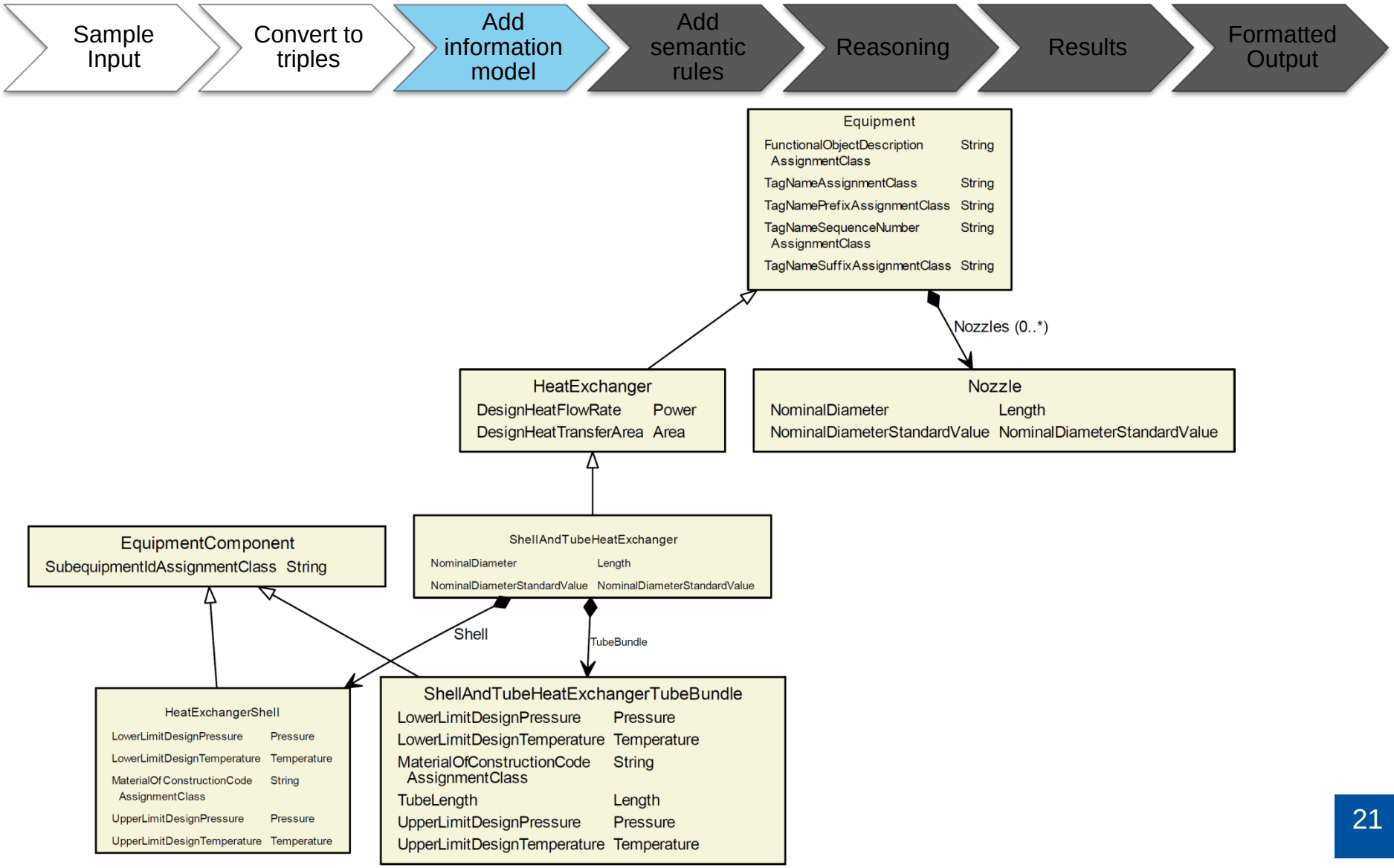
Step 1 : Sample Input as Proteus XML


```
<?xml version="1.0" encoding="utf-8"?>
<PlantModel>
  <PlantInformation
    Date = "2014-03-07"
    Discipline = "PID"
    Is3D = "no"
    OriginatingSystem = "DEXPI Test"
    SchemaVersion = "3.6.0"
    Time = "13:34:01"
    Units = "mm">
    <UnitsOfMeasure/>
  </PlantInformation>
  <Extent>
    <Min X = "0" Y = "0"/>
    <Max X = "800" Y = "600"/>
  </Extent>
  <Equipment
    ID = "P_02"
    ComponentClass = "CentrifugalPump"
    ComponentClassURI = "http://posccaesar.org/rdl/RDS416834">
  </Equipment>
</PlantModel>
```

Step 2: Convert to triples



Step 3 : Add information model



Step 4 : Add semantic rules

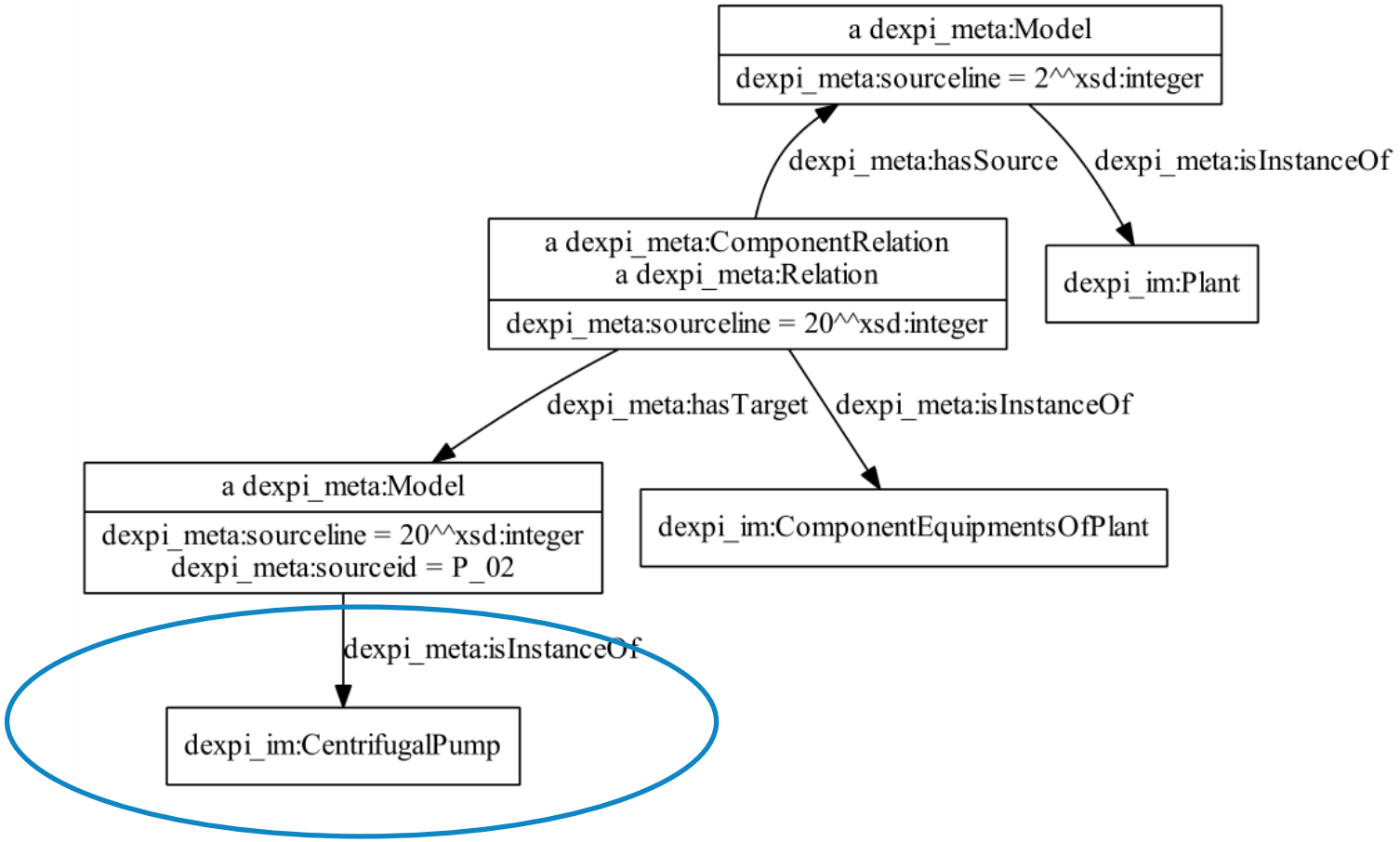

```
(?physical_quantity dexpi_meta:error ?msg)
<-
(?physical_quantity dexpi_meta:hasScale ?scale_usage)
(?scale_usage dexpi_meta:isInstanceOf ?scale)
(?physical_quantity dexpi_meta:isInstanceOf ?physical_quantity_type)
noValue(?physical_quantity_type dexpi_meta:hasScale ?scale)

(?scale dexpi_meta:simpleLabel ?scale_label)
(?physical_quantity_type dexpi_meta:simpleLabel
    ?physical_quantity_type_label)

strConcat(
  ?scale_label
  " is no valid scale for a physical quantity of type "
  ?physical_quantity_type_label
  ?msg)
```

Step 5 : Reasoning

- Implemented in Jena Semantic Web Framework
- General form:
  (set of RDF triples) -> (set of RDF triples)
- Rules are used to
  - enrich the model (cf. previous example)
  - detect errors, e.g., detect invalid units

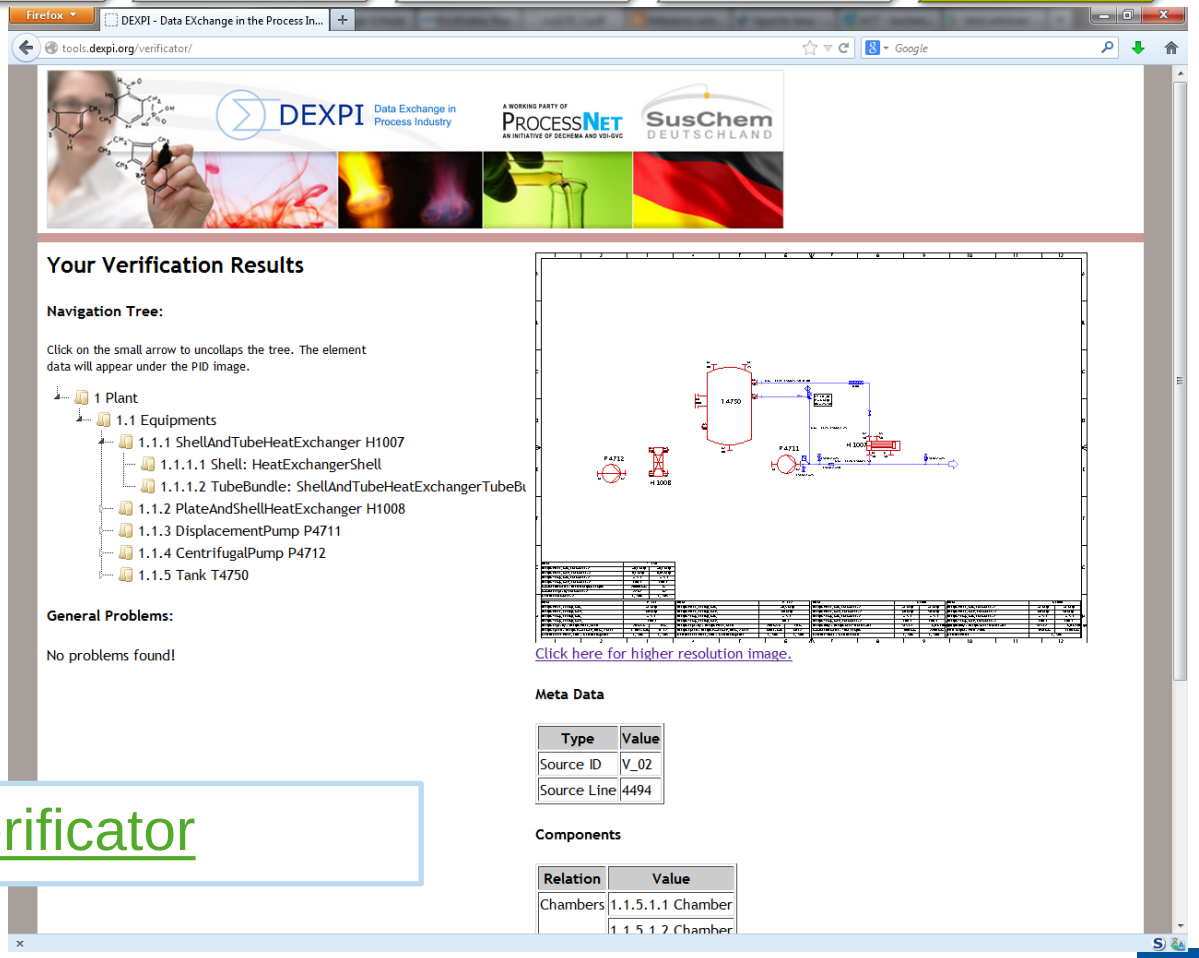
Step 6 : Receiving Results

Sample Input → Convert to triples → Add information model → Add semantic rules → Reasoning → **Results** → Formatted Output



Step 7 : Formatted Output

Sample Input → Convert to triples → Add information model → Add semantic rules → Reasoning → Results → **Formatted Output**



http://tools.dexpi.org/verificator

Summary and outlook

- File verification method based on semantic reasoning
- Applied in the DEXPI project
  - Uses Proteus XML files
  - Can be easily adapted to ISO-15926:8 RDF-OWL files
- Information model will be extended in the near and far future
  - Current focus is on modelling functional Instrumentation of P&IDs
- Future goal is to make use of this method also in other lifecycle steps

Thank you!

Thank you for your attention!