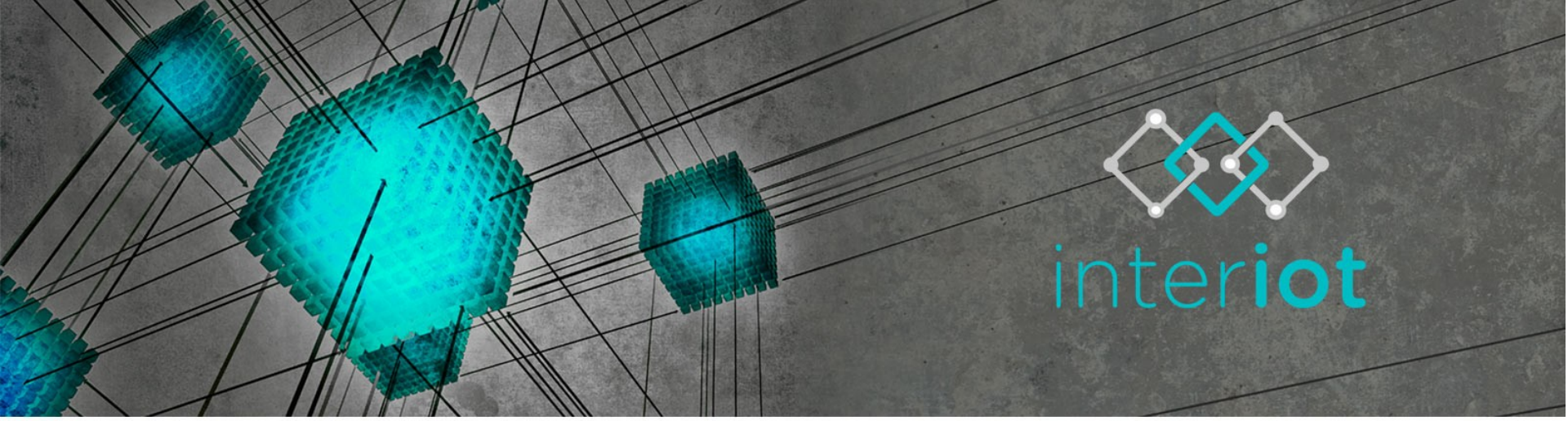


Semantic interoperability in INTER-IoT project

Katarzyna Wasielewska

Maria Ganzha, Marcin Paprzycki, Wiesław Pawłowski, Paweł Szmeja

INTER-IoT

- INTER-IoT – one of seven H2020 projects that started on January 1, 2016
- Silos problem in IoT
  - Ecosystem composed of sensors and actuators (things)
  - Things have to exchange data
  - Multiple incompatible platforms/systems/applications → IoT artifacts
- EC believes that having interoperability in IoT world will be beneficial to make EU more competitive

INTER-IoT use cases

- **Transport and logistics**
  - Port of Valencia
    - different “IoT platforms”
    - outsiders – truck companies
    - IoT-like platforms
    - need to make them talk to each other
- **(e/m)Health**
  - Body sensor network solution
  - Internet enabled devices in homes
  - doctor needs a ”unified view”

INTER-IoT approach

- Interoperabilily on all levels of software stack
- Methodology for making IoT artifacts interoperable
- “We” are interested in semantic interoperability
- Assumptions:
  - more than **two** IoT artifacts to connect/collaborate
  - these artifacts may (later) become involved in other collaborations
  - focus on information exchange (messages)

Inter Platform Semantic Mediator (IPSM)

- An independent component that helps achieving interoperability of IoT artifacts on data and semantic level
- Alignment-based semantic translation of messages in RDF format (transformation of RDF graphs)
- Architecture based on translation channels (publish – subscribe)

Common language solution outline

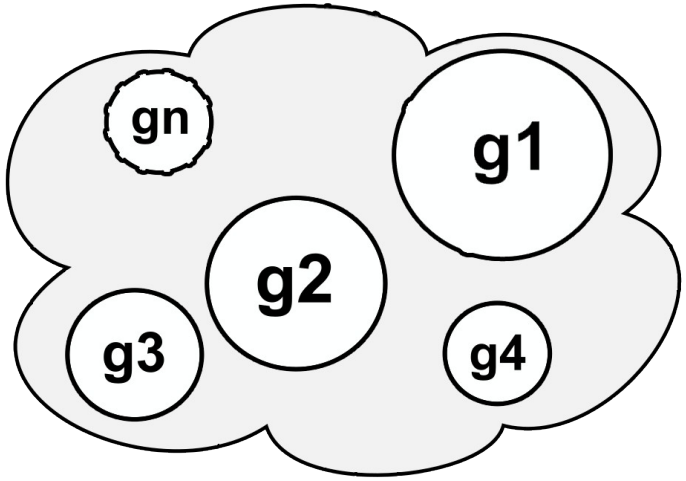
- Instantiate modular **central ontology**
- Lift **semantics** of each artifact to **OWL**
  - Create message **producers** and **consumer**
- **Align** semantics between ontology of each artifact and central ontology
- Create an *instance* of **Inter-Platform Semantic Mediator** (IPSM)

Central modular ontology (1)

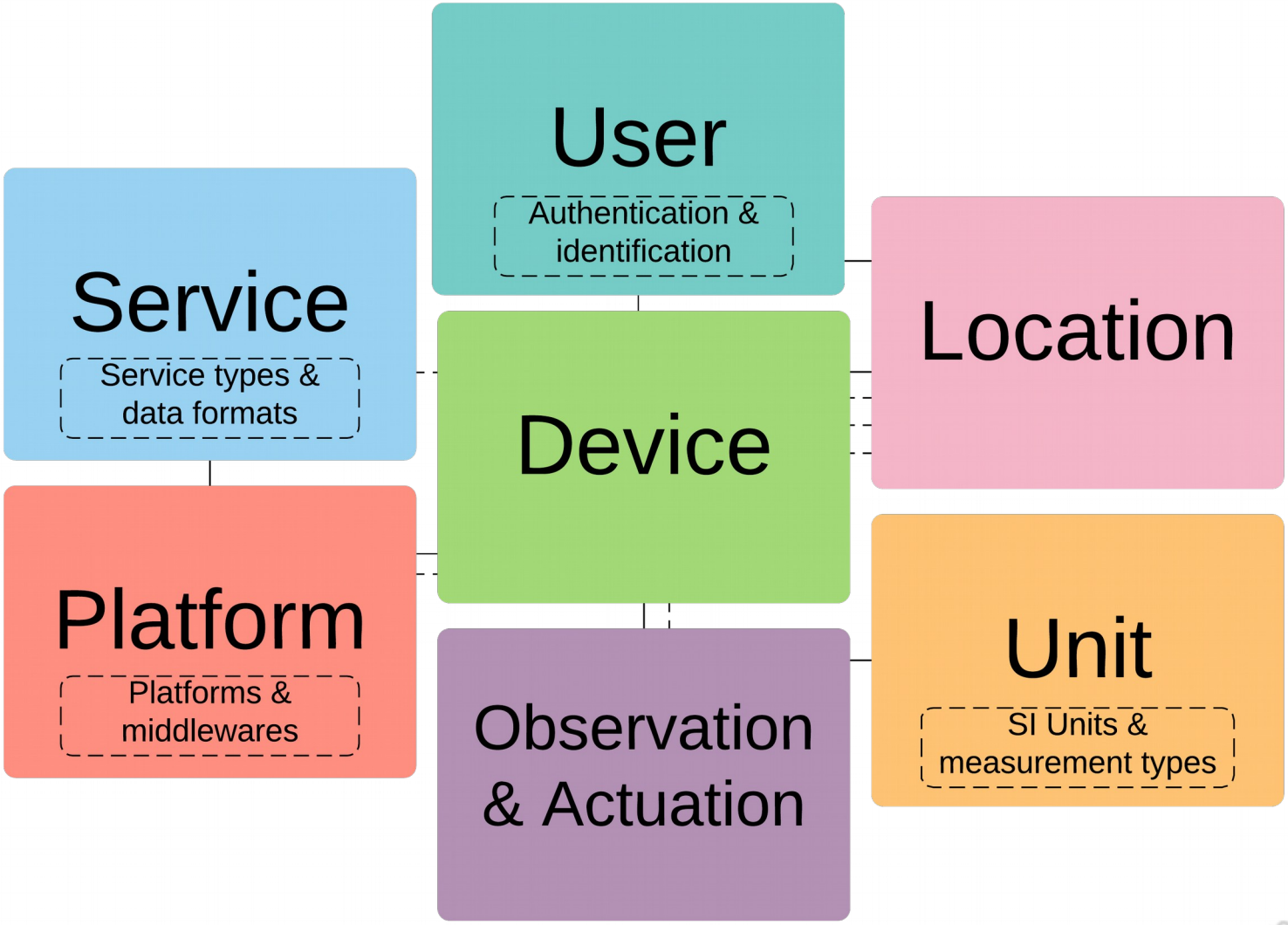
- GOIoTP – Generic Ontology of IoT Platforms
  - Core model of semantics
  - Wide, but not deep
    - Limited to general IoT concepts
    - Extendable with domain-focused modules
  - Based on a combination of existing ontologies
    - SSN / SOSA, GeoSPARQL, WKT, SWEET
  - Extension of Generic Ontology for IoT Platforms (GOIoTPex) for concrete implementation
    - Uses the generic structures of GOIoTP to model specific unit sets, domains, entities, etc.

Central modular ontology (2)

- Generic Ontology for IoT Platforms (GOIoTP)
  - **Generic** – for, broadly-understood, IoT domain (not just devices, but also platforms, services, ...)
  - **Modular** – each module satisfies specific sub-domain requirements
  - **Extendable** for concrete domains and applications



GOIoTP Outline

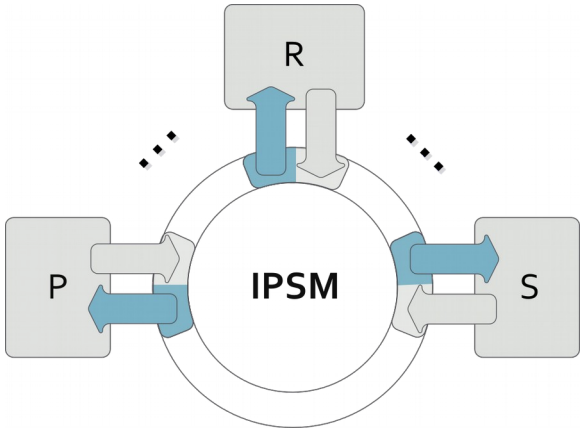


docs.inter-iot.eu/ontology/

interiot

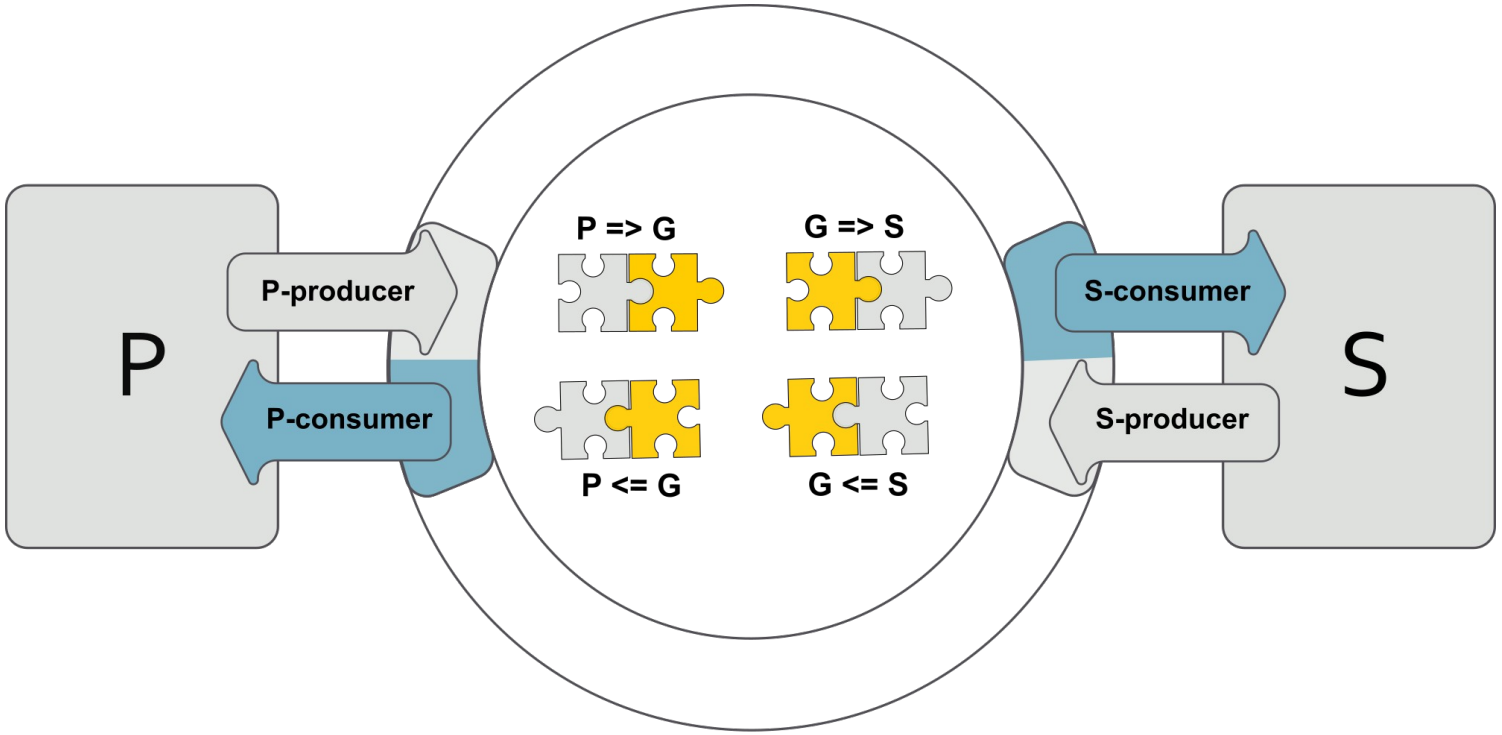
Inter-Platform Semantic Mediator

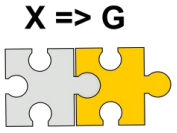
- **Universal** RDF translator / graph rewriter
- Translation medium for **Core Information Model** systems (**Core Ontology**)
- **Configurable** with alignment files
- Offers **scalable channels** (streams) based communication infrastructure



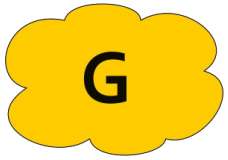
- Main use-case is translation of „small” **messages** on-the-fly (**publish-subscribe streaming**)
- Does **not** offer data-store schema transformation for a whole data set (like e.g. Virtuoso), translates *only when needed* (not-preemptively)
- Enables IoT artifacts to **use their own semantics** and be *globally* understood

Inter-Platform Semantic Mediator





X *output alignment*



common ontology

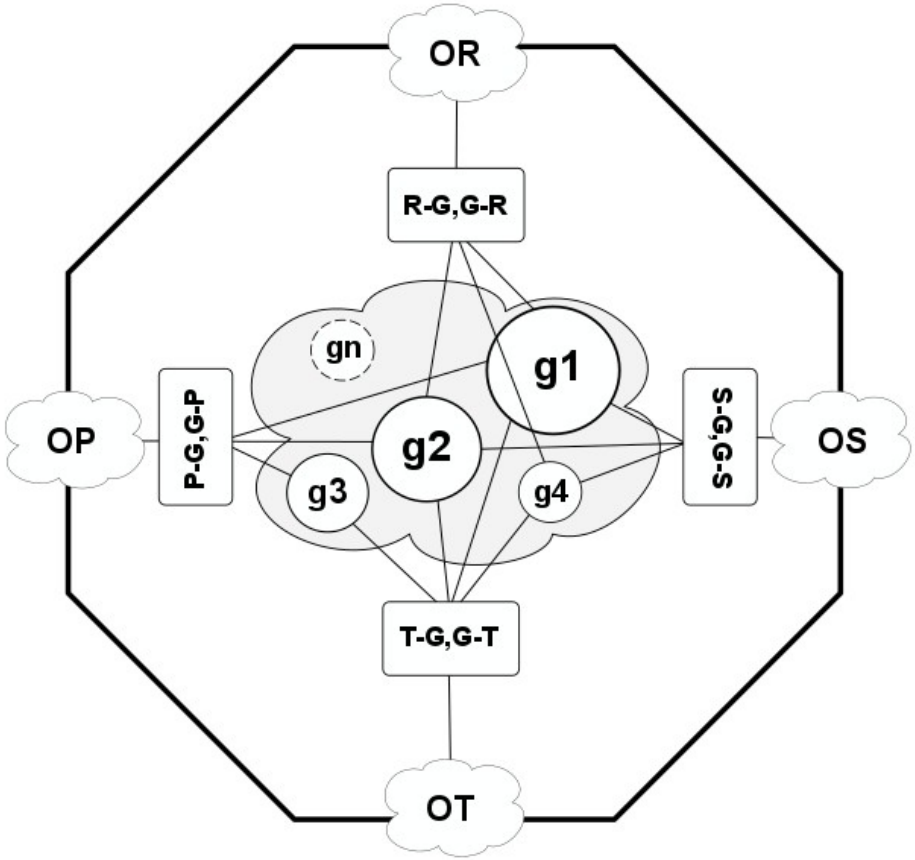
Y *input alignment*

Each translation channel has a pair of alignments (may be „empty”)

interiot

Alignment-based Translation

- Central Ontology (CO) used in IPSM deployment is **modular**
- Alignments address only modules about which an artifact needs to communicate
- Any new joining artifact needs to define <u>two one-way</u> alignments between selected modules of **CO** and its own semantics



IPSM Alignment Format

- IPSM **applies IPSM-AF** files to translation
- Each translation channel requires **two IPSM-AF files** (alignments)
- Each alignment is treated as a **uni-directional** mapping to and from **CO**
- Alignments may be „*desctructive*” (may replace RDF triples), or *preserving* (only adding information)
- Unless explicitly specified in an alignment, no RDF triple is removed from messages – this includes those not translated – **IPSM only translates what it was told to**

IPSM-AF – header

```
{ alignment title, version, creator etc. }
<align:xml>yes</align:xml>
<align:level>2IPSM</align:level>
<align:type>**</align:type>
{ auxiliary Alignment API properties }
<align:onto1>
   { source ontology information }
</align:onto1>
<align:onto2>
   { target ontology information }
</align:onto2>
```

IPSM-AF – steps & mappings

```
<sripas:steps rdf:parseType="Literal">
   <sripas:step sripas:order="[cell order]"
      sripas:cell="[cell id]"/>
   { more steps }
</sripas:steps>

<align:map>
   { alignment cell }
</align:map>
{ more alignment mappings }
```

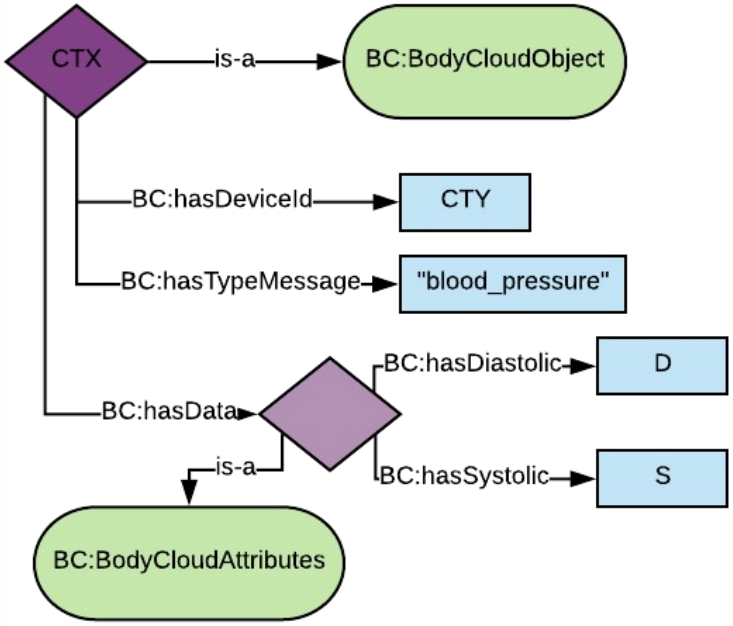
Alignment Cell

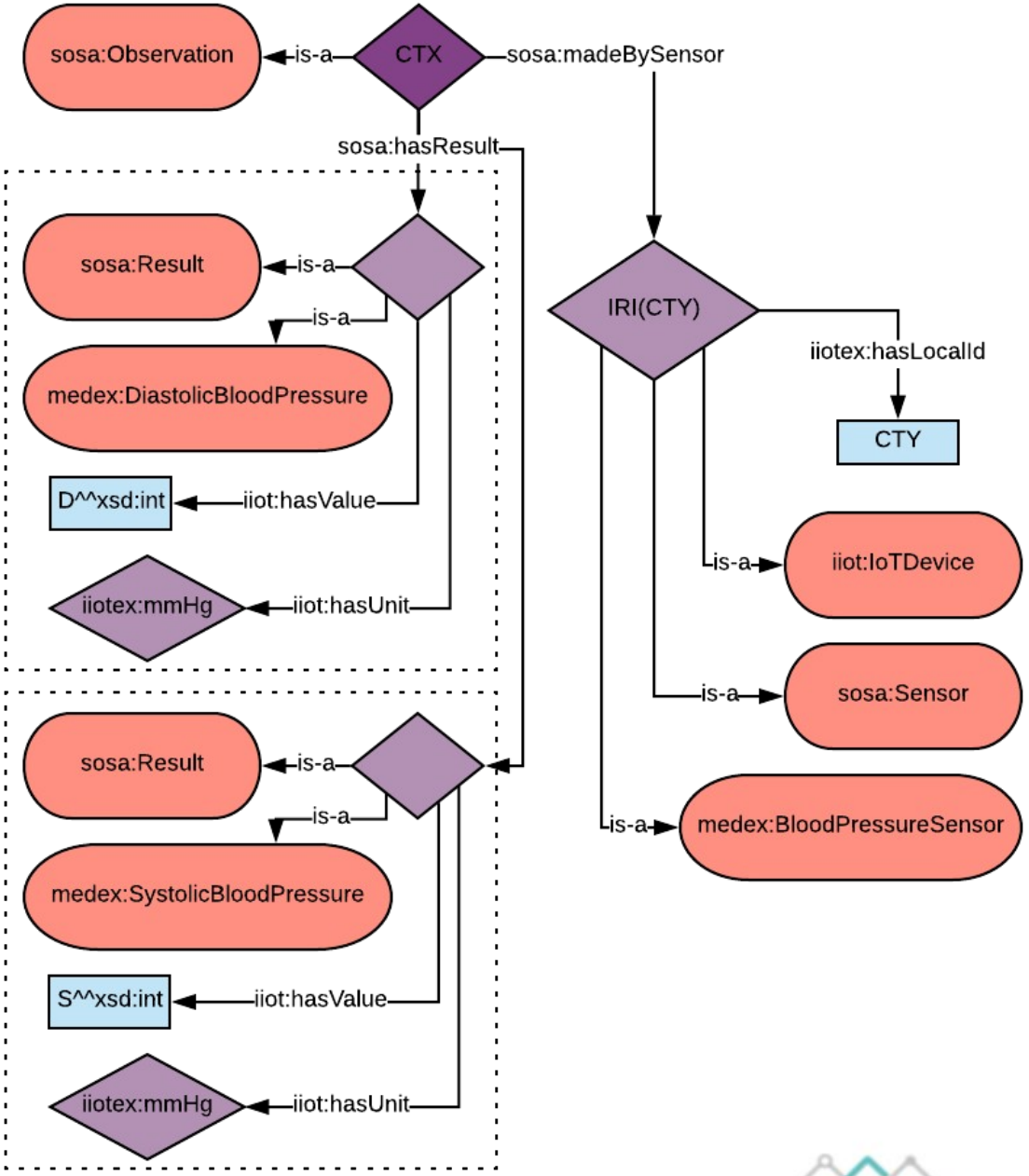
```
<align:Cell rdf:about ="[cell id]">
    <align:entity1 rdf:parseType ="Literal">
        {source RDF pattern}
    </align:entity1>
    <align:entity2 rdf:parseType ="Literal">
        {target RDF pattern}
    </align:entity2>

    {relation (e.g. equivalence) and confidence – Alignment API „legacy”}
    {datatype filters and typings}

    <sripas:transformation rdf:parseType ="Literal">
        {functional constraints}
    </sripas:transformation>
</align:Cell>
```

Alignment Excerpt





interiot

```
<align:entity1 rdf:parseType ="Literal">
    <sripas:node_CTX>
        <bc:hasTypeMessage>"blood_pressure"</bc:hasTypeMessage>
        <bc:hasDeviceId>  <sripas:node_CTY/>  </bc:hasDeviceId>
    </sripas:node_CTX>
</align:entity1>
<align:entity2 rdf:parseType ="Literal">
    <sripas:node_CTX>
        <sosa:madeBySensor>
            <rdf:Description rdf:about="&sripas;node_uri">
               <rdf:type rdf:resource="&iiot;IoTDevice" />
               <rdf:type rdf:resource="&personalHealthDevice;BloodPressureSensor" />
            </rdf:Description>  </sosa:madeBySensor>
    </sripas:node_CTX>
</align:entity2>
<sripas:transformation>
    <sripas:function about="IRI">
        <param order="1" about="&sripas;node_CTY"/>
        <return about="&sripas;node_uri"/>
    </sripas:function>
</sripas:transformation>
```