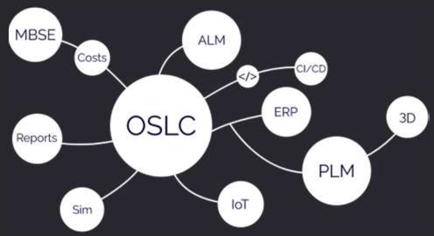


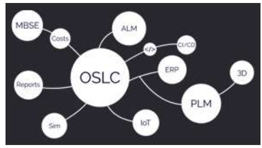
OSLC FEST 2022
Virtual Conference

17 November 2022

OSLC for Cognitive Cross-Checking of System Models

Carlos C. Insaurralde
Bristol Robotics Laboratory
University of the West of England
Bristol, United Kingdom

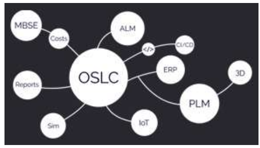
Carlos C. Insaurralde, MEng, PgCTLHE, MAst, MPhil, PhD



IEEE Senior Member, HEA Fellow

- US-funded(DoD), UK-funded(MoD) and EU-funded projects
  - US research project in collaboration with the AFRL.
  - UK research projects in collaboration with BAE Systems, Atlas Elektonik, and seabyte.
  - EU research projects in collaboration with Airbus, Eurocopter, Goodrich, Autoflug, ASG, and Secondo Mona.
- Autonomy-based projects
  - Autonomously cross-checked models from multidisciplinary design teams of high-integrity systems.
  - Autonomous decision-making support for avionics analytics.
  - Remote integration of capabilities from autonomous ground vehicles for defence.
  - Automation of distributed aircraft fuel management systems tested in lab and real-scale rigs.
  - Intelligent control architecture for autonomous maritime vehicles.
  - Autonomous reconfiguration of production lines.
- Over 100 publications, including a book and 5 book chapters.

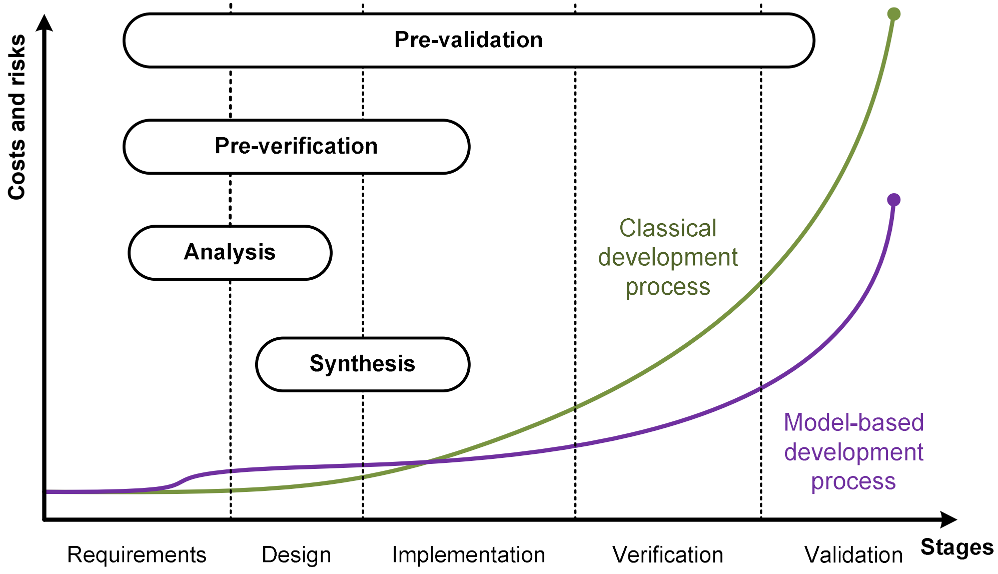
Contents

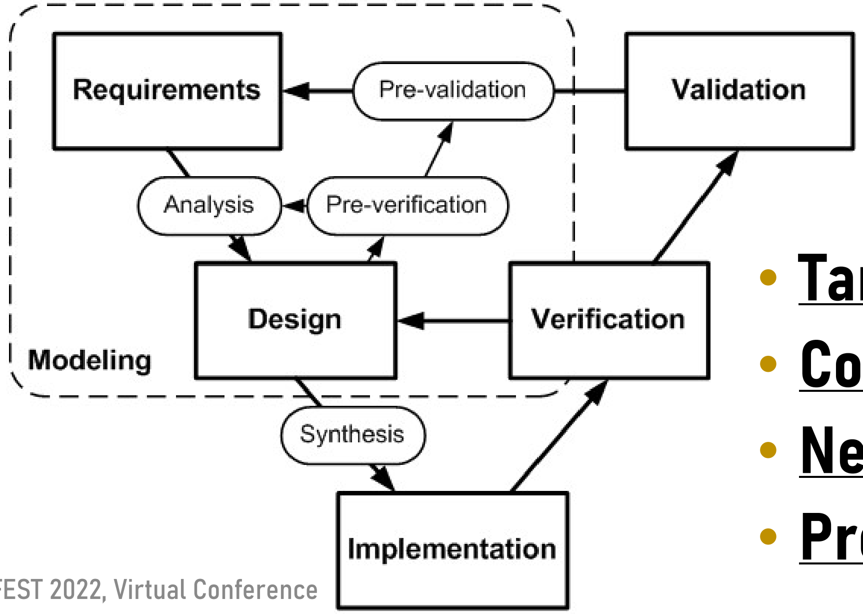


- Introduction
- Background
- Example
- Approach
- Reflection

Early Efforts to Reduce Costs and Risk

- **Focus:** Dependable Cyber-Physical Systems
- **Feature:** High-integrity; safety, security, etc.
- **Challenge:** SDLC risks & costs increasing
- **Trend:** Pre-verification & pre-validation

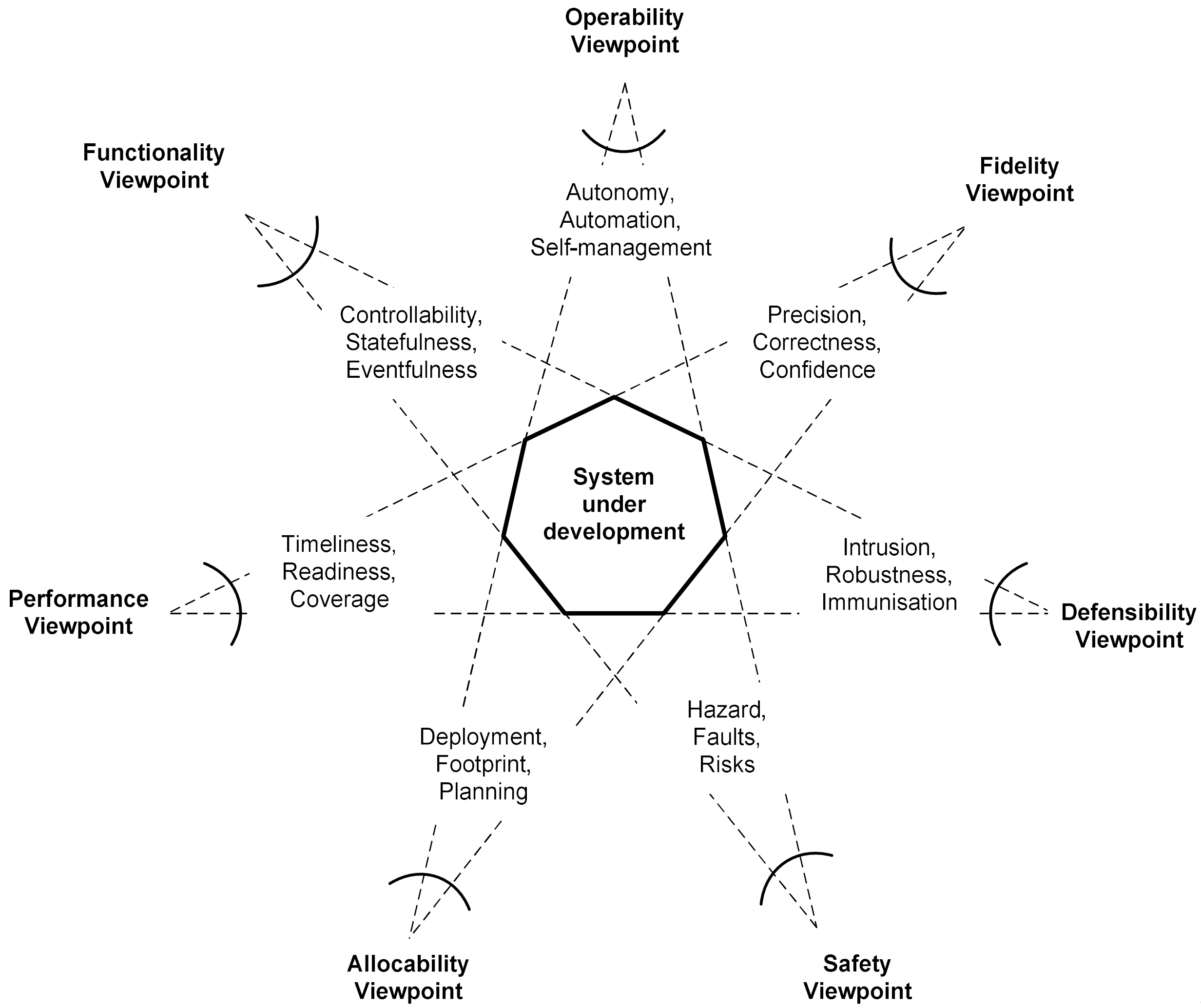




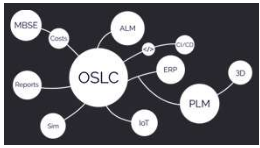
- **Target:** Functional & non-functional requirements
- **Constrain:** Multiple models and teams
- **Need:** Agile model cross-checking
- **Problem:** Deadlocks between development Teams

Different Views/Models of the Same System

- One single (unified) system model is impossible
- But it could be a notation-unified system model
- The approach is to merge notation from different domain-specific representations
- To check the impact of each representation on others can quickly be reflected.

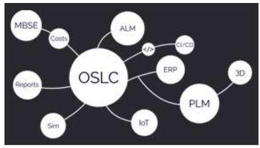


Related Existing Technologies



- Integration of Models
  - Cyber-Physical Modelling [1]
  - OSLC
  - MIC (Model-Integrated Computing) [2]
  - CIF (Compositional Interchange Format) [3]
- Multi-View Tools
  - Modelica (control) [4], 20sim (mechatronics) [5]
  - MIC, MVM [6]
- Single-View Tools
  - AADL [7], MARTE [8]
  - SysML, UML
  - COMPASS [9], CRYSTAL [10]
  - ISO/IEC/IEEE 42010 standard [11]

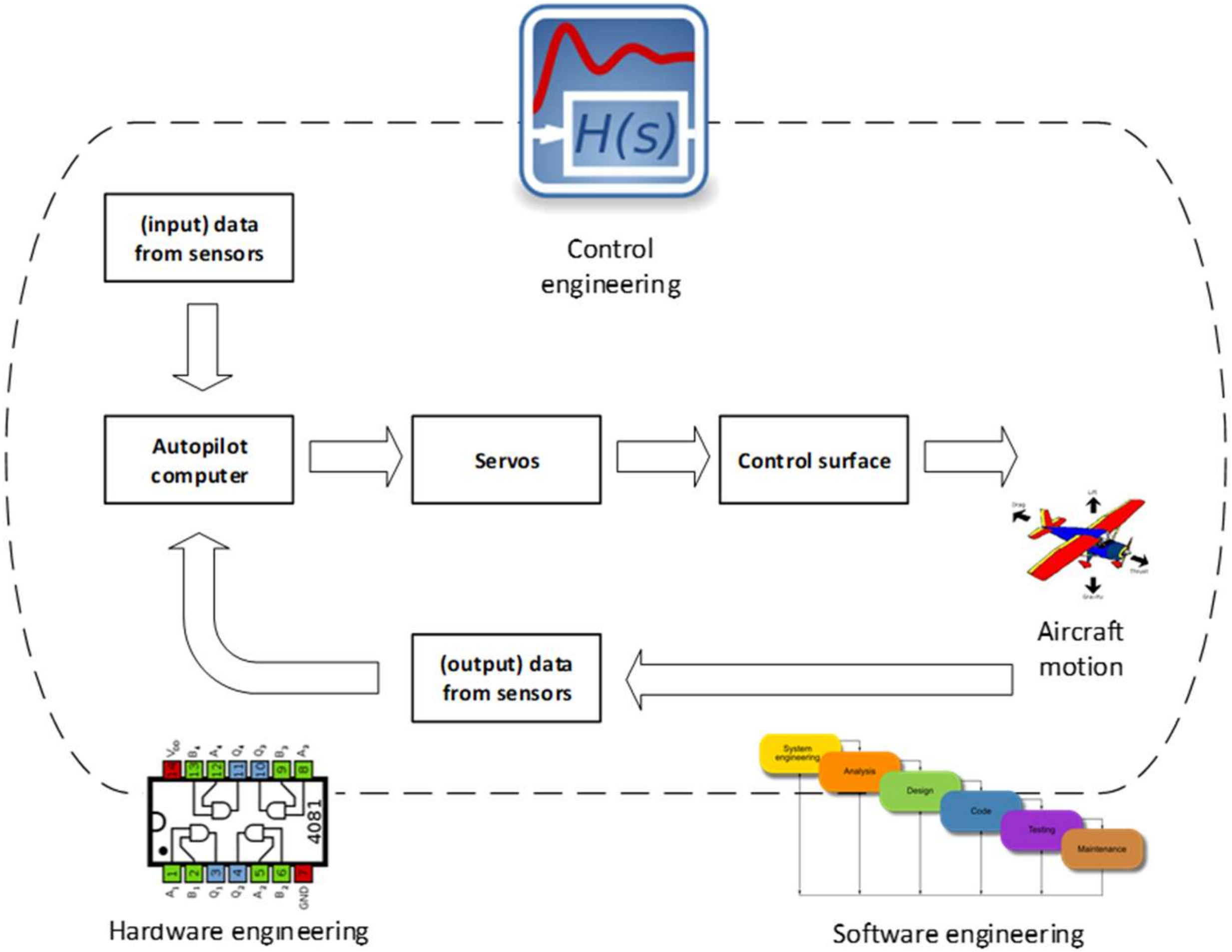
Application Example



- An automatic Flight Control System (AFCS).
- Three distinct models (views) from different engineering disciplines are considered to design the above AFCS:
  - a control engineering model,
  - a software application model
  - a hardware platform model.
- They have different description languages to model the AFCS, e.g. block diagrams, UML diagrams, and AADL diagrams.
- Three application scenarios:
  - a software application model connected to a hardware platform model and a control engineering model.
  - a hardware platform model connected to a software application model and a control engineering model.
  - A control engineering model connected to a hardware platform model and a software application model.

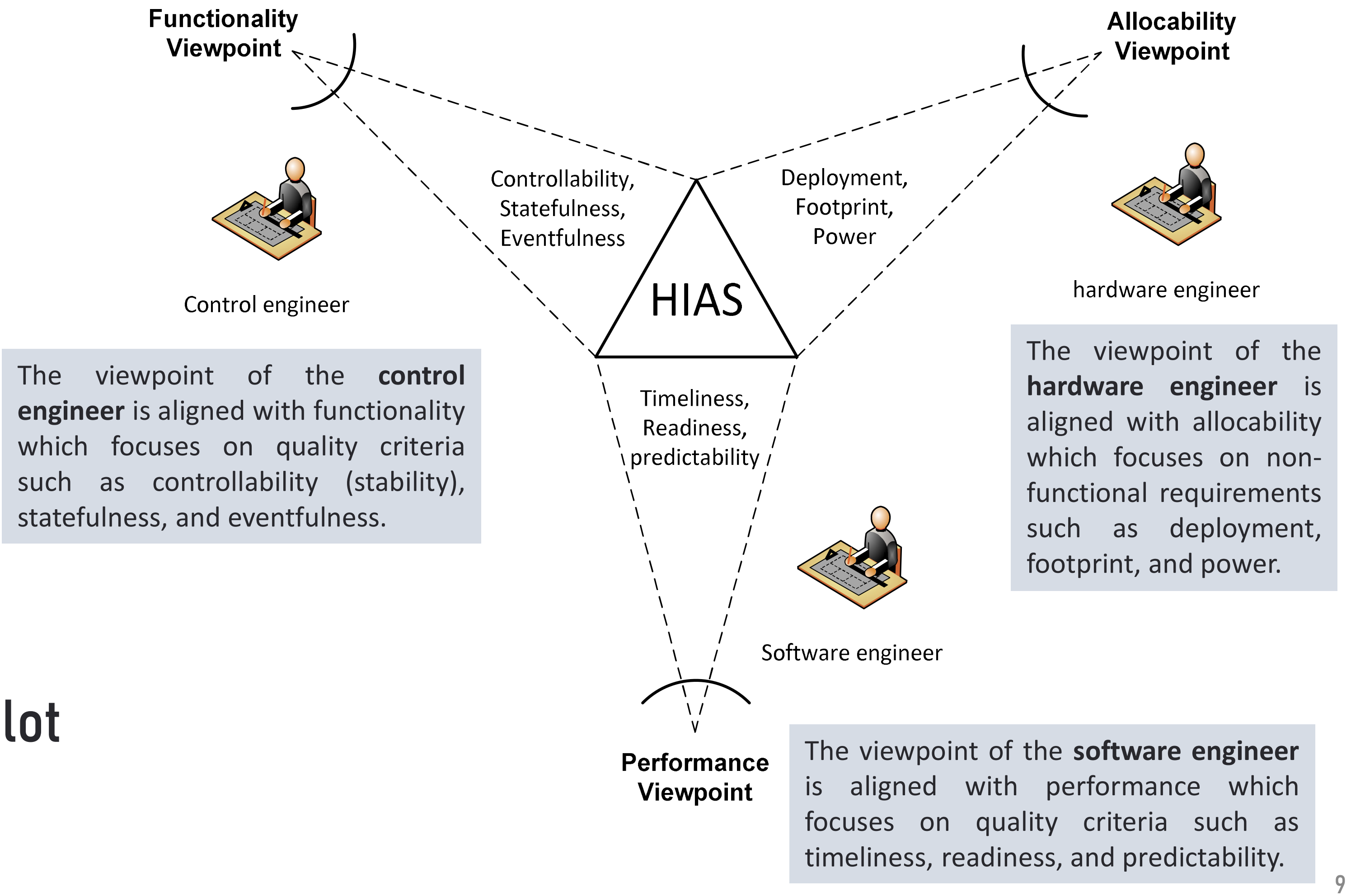
Simplified Example of Aircraft Autopilot

- Simple example but enough to show the idea
- Three disciplines
  - Control engineering
  - Hardware engineering
  - Software engineering
- Stakeholders from each discipline



Multiple Stakeholder Viewpoints

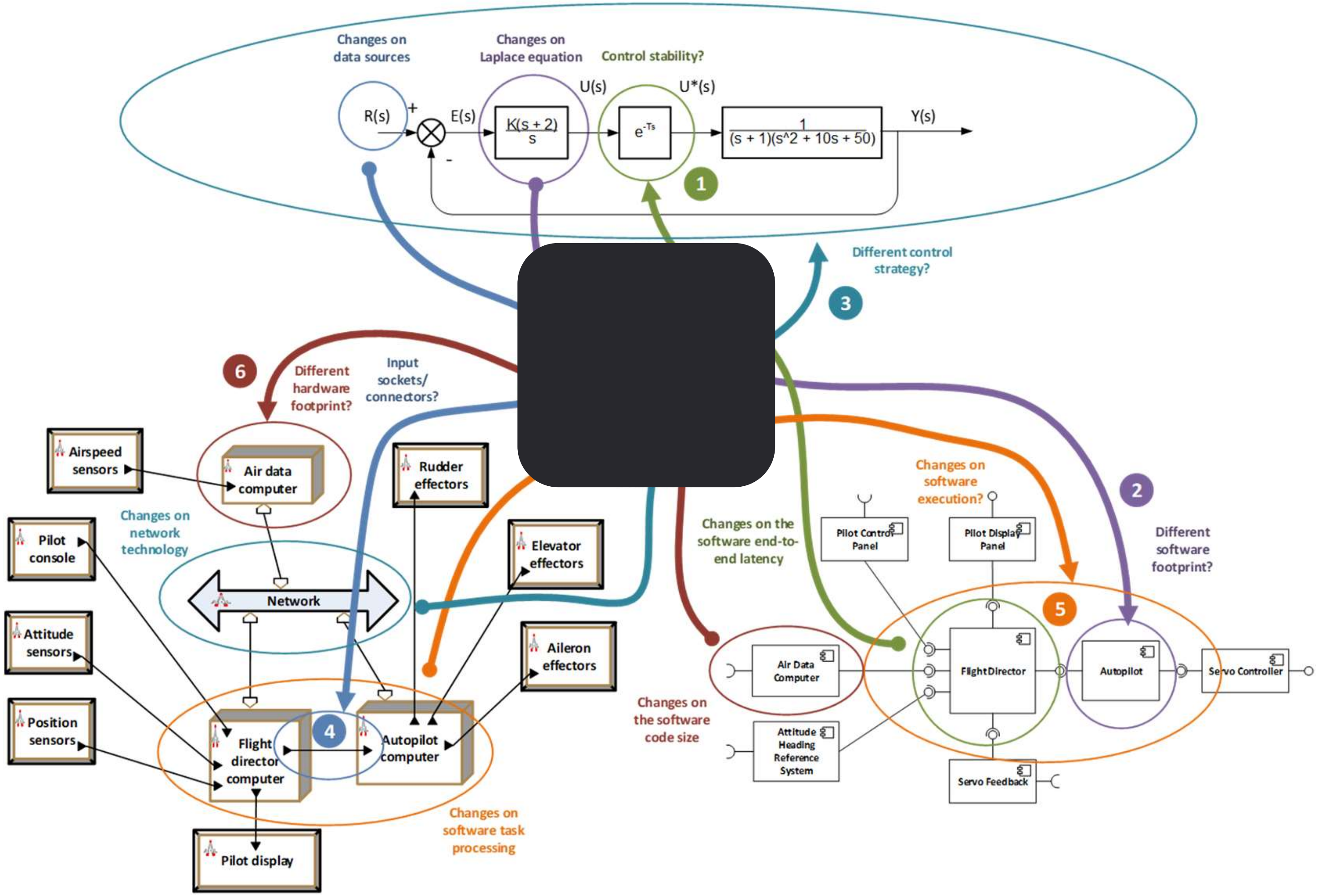
- Three stakeholders
  - Control engineer
  - Hardware engineer
  - Software engineer



- HIAS: AFCS or autopilot

Simplified Example of Aircraft Autopilot

1) SW-CW:
   Latency ⇒ Stability
2) CW-SW:
   Modelling ⇒ Footprint
3) HW-CW:
   Architecture ⇒ Strategy
4) CW-HW:
   Data ⇒ Socket
5) HW-SW:
   Processing ⇒ Execution
6) SW-HW:
   Coding ⇒ Footprint



Software Impact on Hardware and Control

Software Model Updates	Hardware Model Effects	Control Model Effects
End-to-end latency (QM)	SP: Execution SE: Task SC: Processing QF: Efficiency QC: Schedulability QM: Scheduling time	SP: Delay SE: Controller SC: Feedback control QF: Dependability QC: Responsiveness QM: Time response
	SP: Transmission SE: Packet SC: Communication QF: Efficiency QC: Responsiveness QM: Network latency	
Code size (QM)	SP: Footprint SE: Memory SC: Computation QF: Efficiency QC: Allocability QM: Hardware use	SP: Control parameter SE: Controller SC: Feedback control QF: Dependability QC: Stability QM: Root Locus, Bode margins
Data size (QM)	SP: Storage SE: Memory SC: Information QF: Efficiency QC: Mem Allocability QM: Mem use	
	SP: Bandwith SE: Network SC: Information QF: Efficiency QC: Network Usability QM: Network use	

Computation

Communication

SP: System Property
SE: System Element
SC: System Capability
QF: Quality Factor
QC: Quality Criteria
QM: Quality Metric

Model Analytics

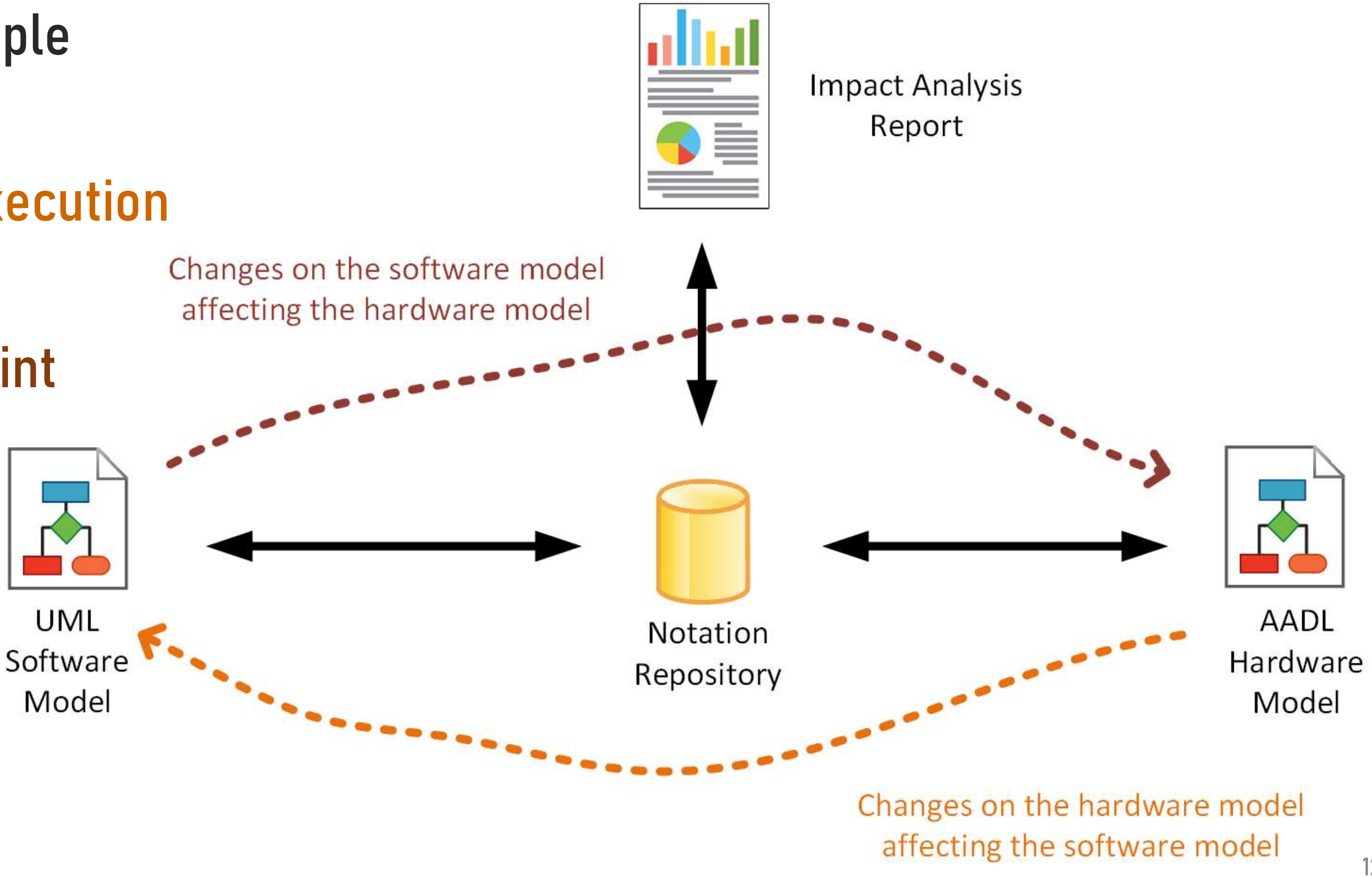
- HW-SW-HW Example

5) HW-SW:

   Processing ⇒ Execution

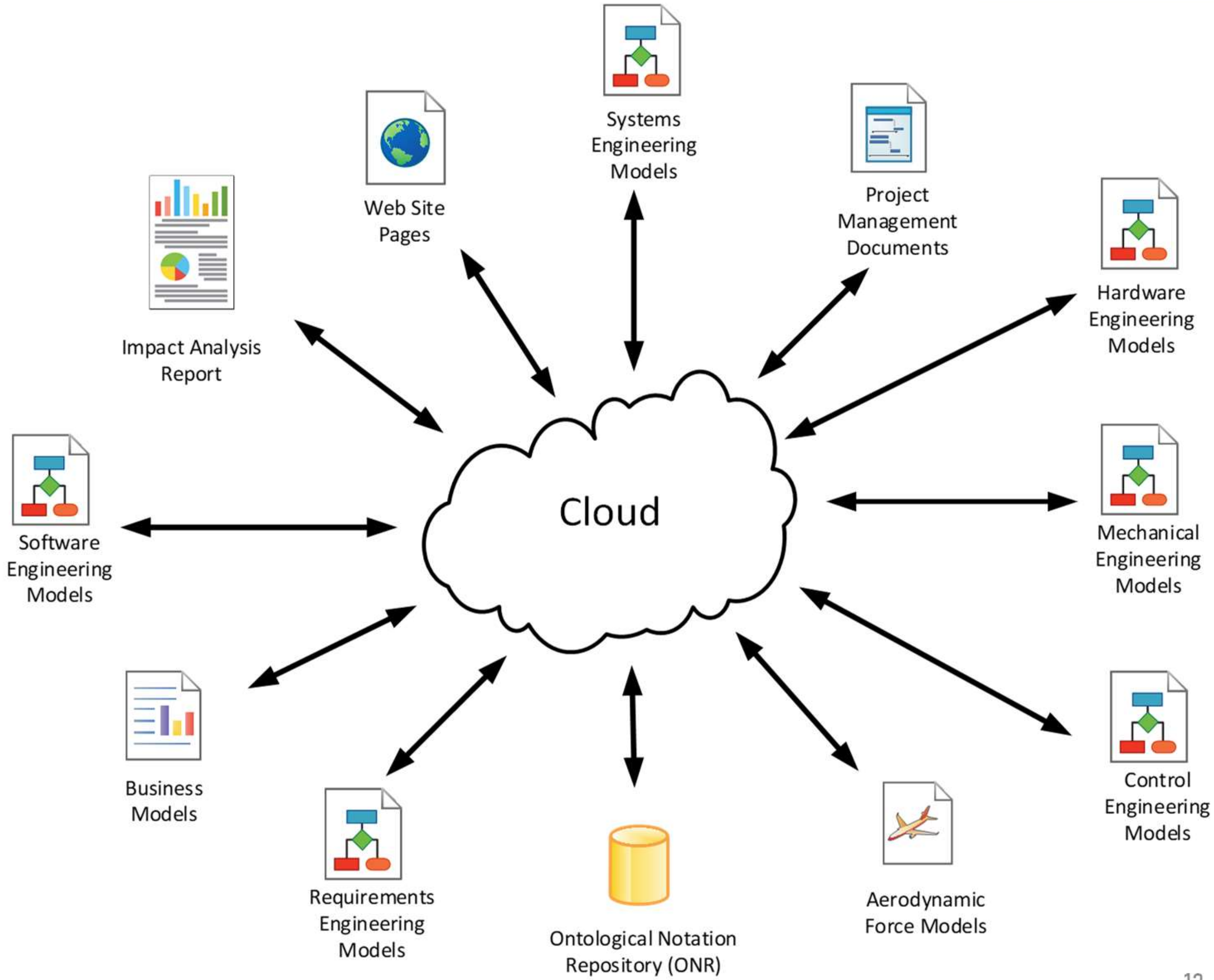
6) SW-HW:

   Coding ⇒ Footprint



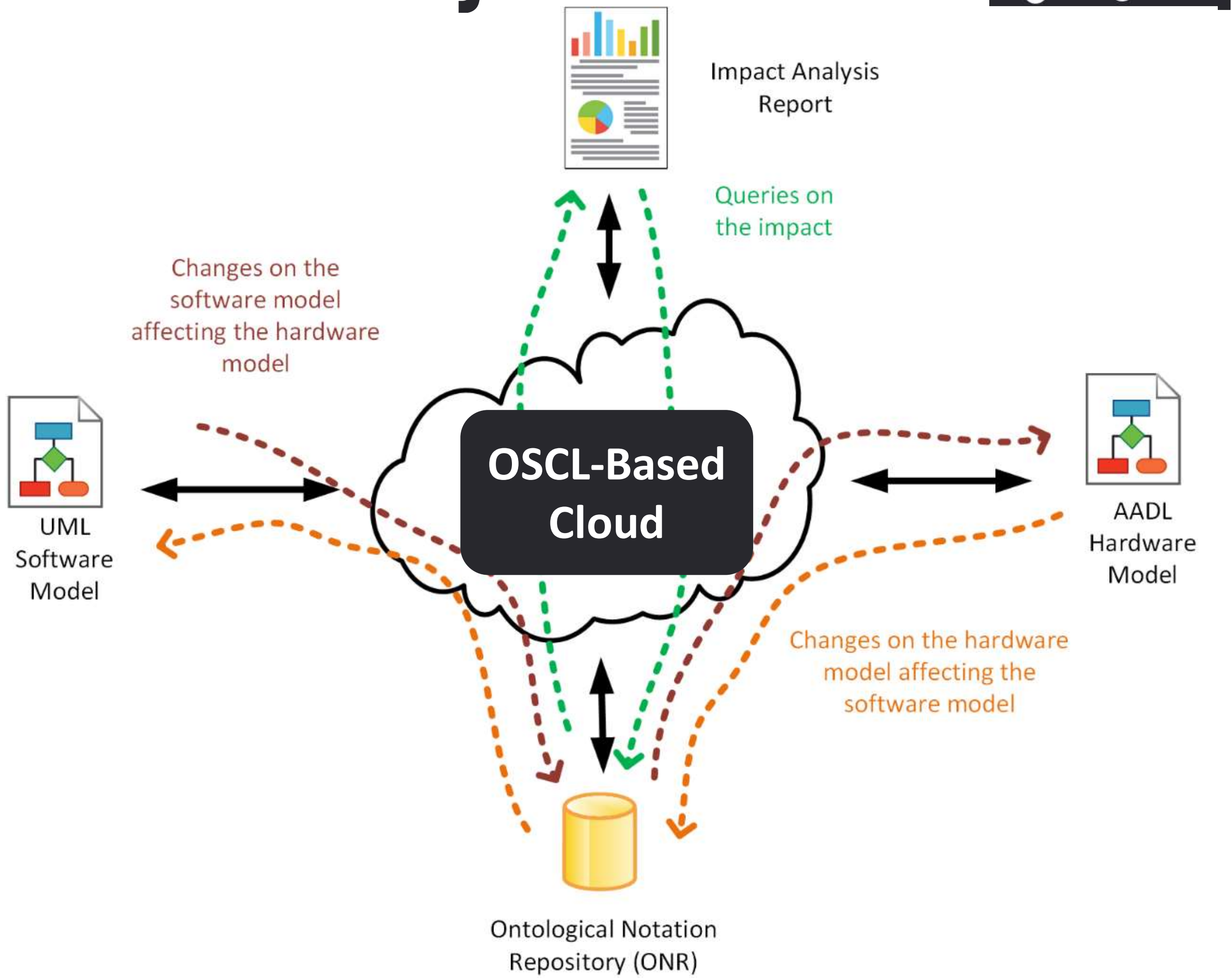
Cloud-Based Framework

- Implementing a cloud-based approach for the framework
- Merging the model notation (parameters) into a single repository for analysis
- Modelling notation includes the functional and non-functional requirements and constraints from different engineering disciplines.



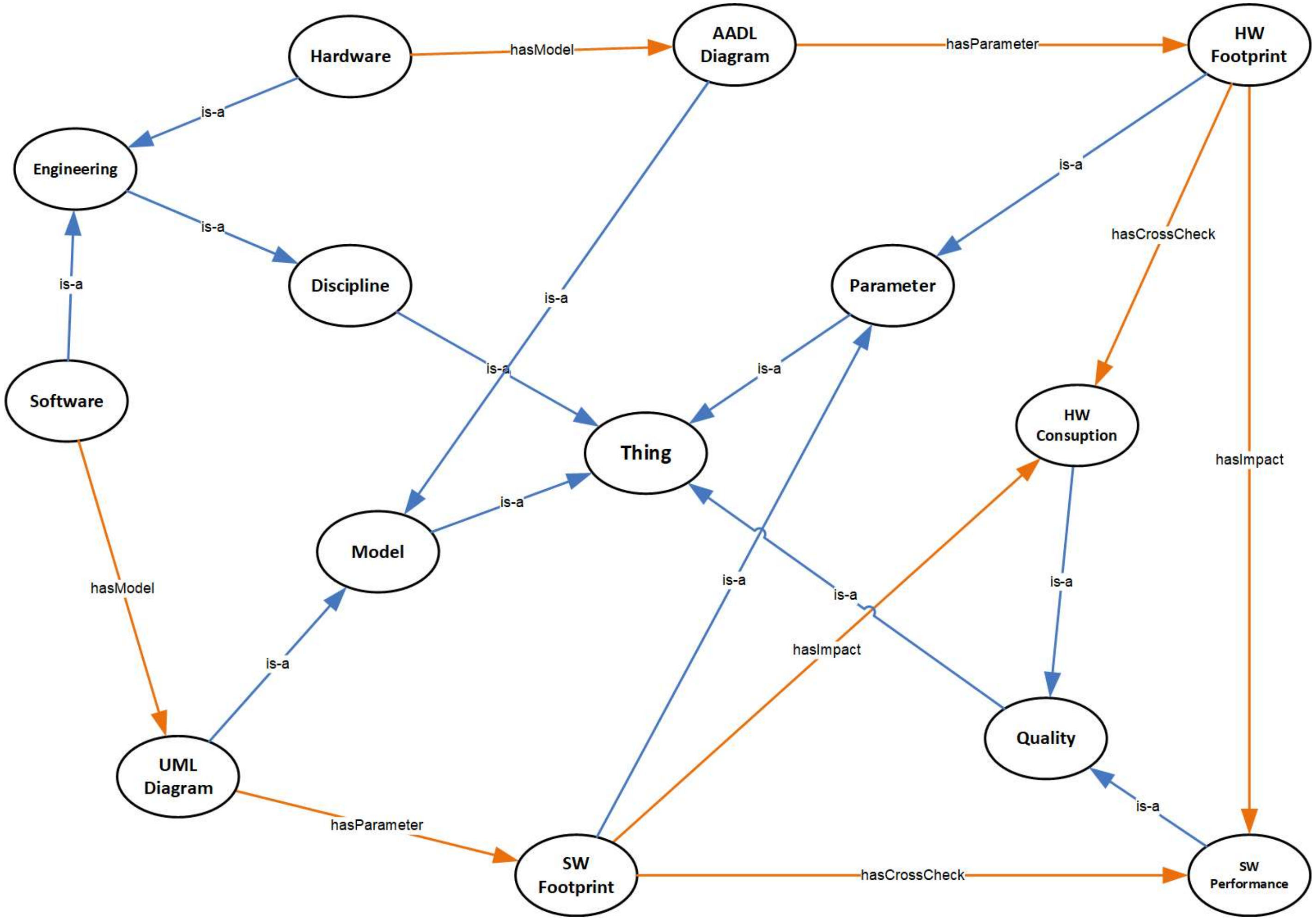
OSCL Cloud-Based Model Analytics

- Relating models
- Cross-checked modelling
- OSCL-based cloud to interconnect models



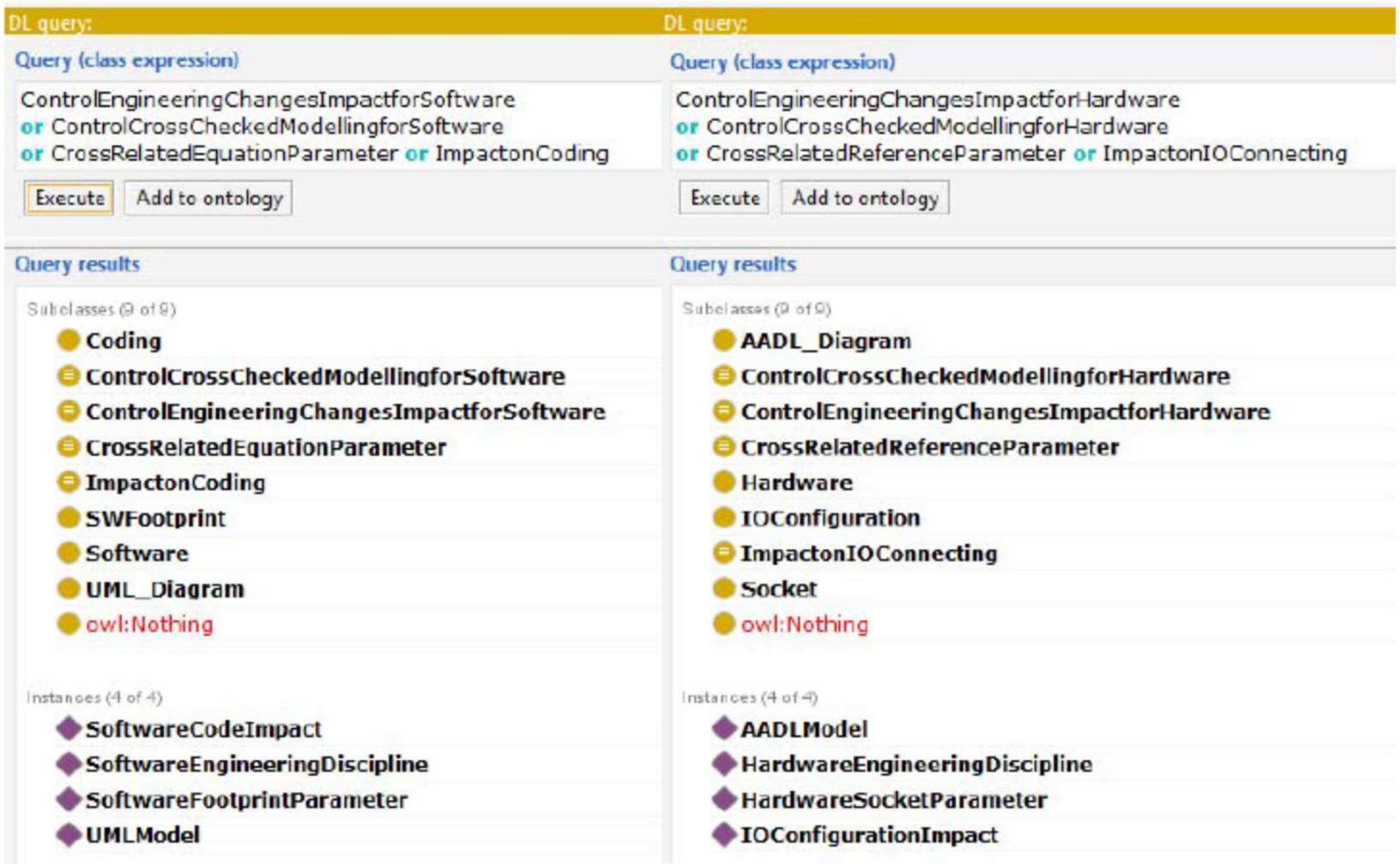
Ontological Notation Repository

- Ontologies make use of **semantic diagrams** to easily realize concepts the ontology and the connections between concepts.



Ontological Proof of Concept

- Protégé user interface for the ontology (ONR)
- Relating models and cross-checked modelling
- OSCL-based cloud to interconnect models



Software Model Service

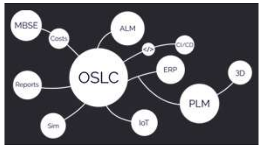
- Application scenario:
  - Software impacts on Hardware

- Software model on the Autonomous Model Analytics (AMA) application

- OSLC service interface for the software model
  - Creation of service
  - Resource shape
  - Query capability
    - Query resource

```
<oslc:service>
  <oslc:Service>
    <oslc:domain rdf:resource="http://open-services.net/ns/am#"/>
      <oslc:creationFactory>
        <oslc:CreationFactory>
          <dcterms:tittle>Creation of Software Model Service</dcterms:tittle>
          <!-- Creation of new resource (the software model)>
          <oslc:creation rdf:resource="http://host/creation/swmodelnotation"/>
          <!-- Metadata of the XML resource>
          <oslc:resourceShape rdf:resource="http://host/shapes/swnotationshape"/>
          <oslc:usage rdf:resource="http://open-services/ns/core#default"/>
          <oslc:resourceType rdf:resource="http://open-services/ns/am#Resource"/>
        </oslc:CreationFactory>
      </oslc:creationFactory>
      <oslc:queryCapability>
        <oslc:QueryCapability>
          <oslc:queryBase rdf:resource="http://host/query"/>
          <oslc:resourceShape rdf:resource="http://host/shapes/ swhwlinkqueryshape "/>
        </oslc:QueryCapability>
      <oslc:queryCapability>
   </oslc:Serice>
</oslc:service>
```

Software Model Resource Shape



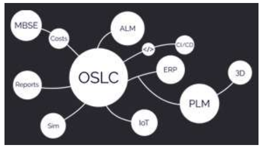

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dcterms="http://purl.org/dc/terms/"
         xmlns:foaf="http://xmlns.com/foaf/0.1/"
         xmlns:oslc="http://open-services.net/ns/core#">

<oslc:ResourceShape rdf:about="http://host/shapes/swnotationshape">
  <dcterms:title>Shape for the software model notation</dcterms:title>
  <oslc:name>SWNotation</oslc:name>
  <oslc:describes rdf:resource="http://host/services/swm#Notation"/>

  <!–Resource shape for the software footprint parameter>
  <oslc:property>
    <oslc:Property>
      <oslc:name>SWF</oslc:name>
      <dcterms:title>Software Footprint</dcterm:title>
      <oslc:propertyDefinition rdf:resource="http://host/services/swm#SWF"/>
      <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
      <oslc:occurs rdf:resource="http://open-services.net/ns/core#Zero-or-one"/>
      <dcterms:description>Use of hardware based on the software size</dcterms:description>
    </oslc:Property>
  </oslc:property>
```

```xml
    <!–Resource shape for the software performance quality>
  <oslc:property>
    <oslc:Property>
      <oslc:name>SWP</oslc:name>
      <dcterms:title>Software Performance</dcterm:title>
      <oslc:propertyDefinition rdf:resource="http://host/services/swm#SWP"/>
      <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
      <oslc:occurs rdf:resource="http://open-services.net/ns/core#Zero-or-one"/>
      <dcterms:description>Software performance based on hardware 
capacity</dcterms:description>
    </oslc:Property>
  </oslc:property>
</oslc:ResoruceShape>
</rdf:RDF>
```

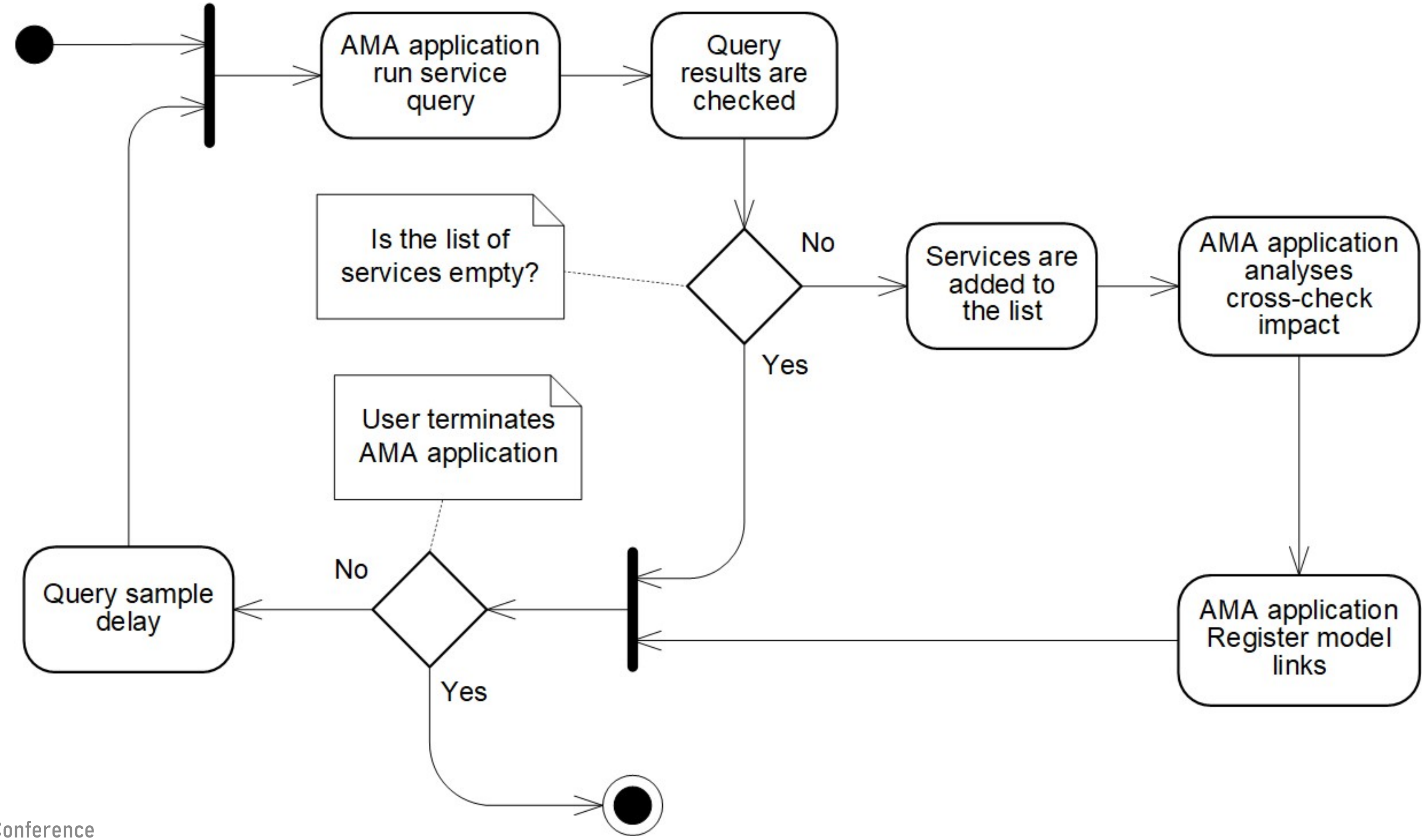
Software Model Query



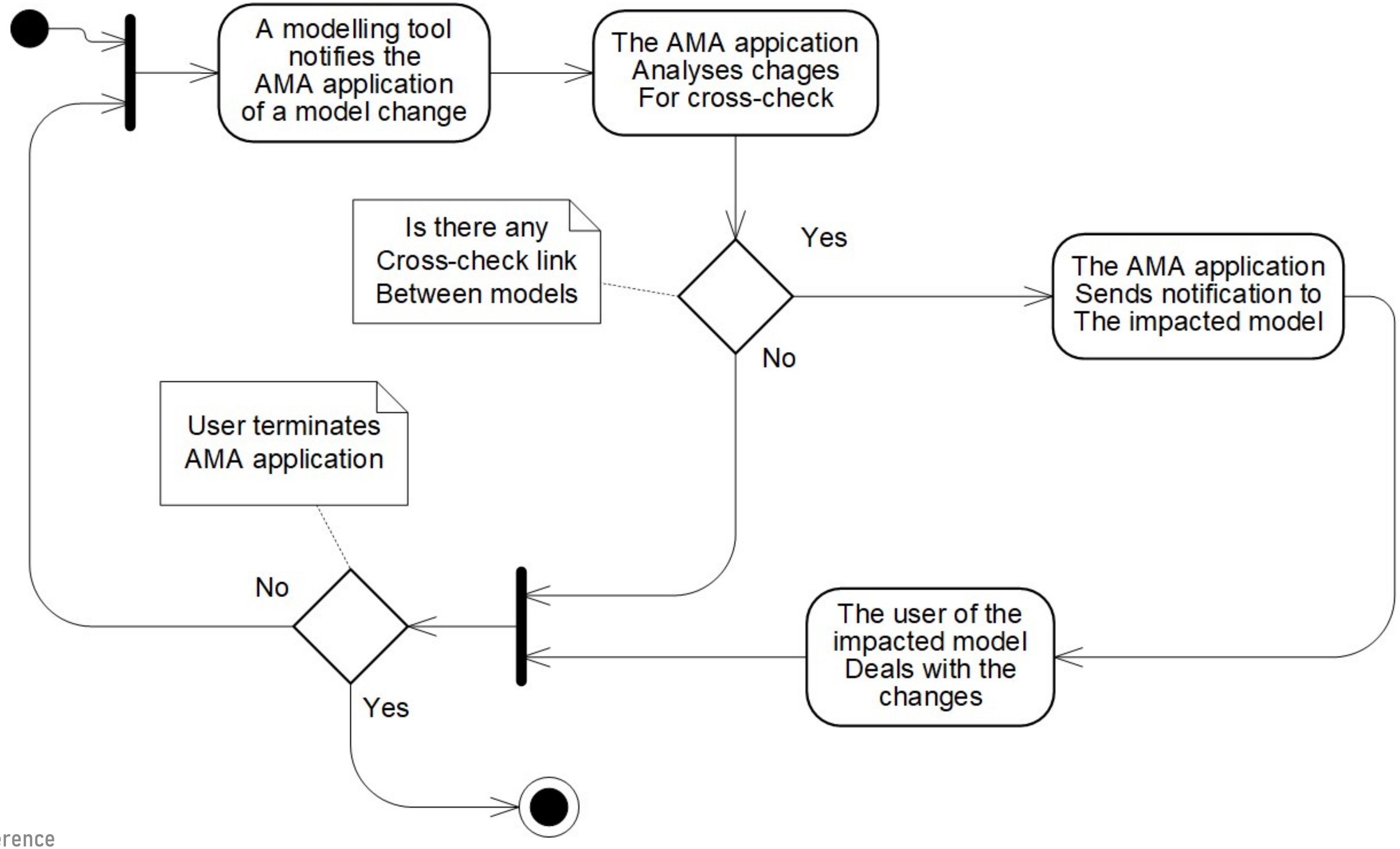

```xml
<?xml version=”1.0” encoding=”UTF-8”>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dcterms="http://purl.org/dc/terms/"
         xmlns:oslc="http://open-services.net/ns/core#">
<oslc:ResourceShape rdf:about="http://host/shapes/swhwlinkqueryshape">
  <dcterms:title>Shape for the software-hardware link</dcterms:title>
  <oslc:type rdf:resource="http://open-services.net/ns/core#ResourceShape"/>
  <oslc:name>SwHwLinkQuery</oslc:name>
  <oslc:describes rdf:resource="http://open-services.net/ns/swm#"/>

  <!–Resource shape for the software-hardware link query>
  <oslc:property>
    <oslc:Property>
      <oslc:name>SWNotation</oslc:name>
      <oslc:occurs rdf:resource="http://open-services.net/ns/core#Zero-or-many"/>
      <oslc:valueShape rdf:resource="http://host/shapes/swnotationshape"/>
      <oslc:propertyDefinition rdf:resource="http://host/services/swm#Notation"/>
      <oslc:isMemberProperty>true</oslc:isMemberProperty>
    </oslc:Property>
  </oslc:property>
</oslc:ResourceShape>
</rdf:RDF>
```

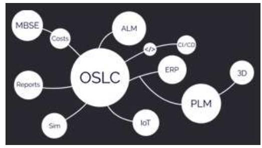
Discovery & Retrieve of Service Info



Analysis of Model Information



Reflective Remarks



- Impacts from model updates on other models cannot necessarily to implement in real time
- Benefits from the model cross-checking framework but is complex by nature; toward process automation
- Model impacts can be also used for performance assessment
- OSCL is a good driver for the framework. However, it will add some complexity to the framework.
- OSCL facilities the model interconnections for cross-checking impact but a lot of work on producing the XML files
- Need for an automated process for the generation of OSLC interfaces for the framework
- Future work: development of framework prototype

References

[1] E. Palachi, C. Cohen, S. Takashi, "Simulation of Cyber-Physical Models using SysML and Numerical Solvers", 6th IEEE International Systems Conference, Orlando, FL, USA, pp. 164-169, Apr 2013.

[2] G. Karsai, J. Sztipanovits, A. Ledeczi, T. Bapty, "Model-Integrated Development of Embedded Software", Proceedings of the IEEE, vol. 91, issue 1, pp. 145-164, 2003.

[3] The Compositional Interchange Format for hybrid systems, available at http://cif.se.wtb.tue.nl.

[4] Modelica, available at https://www.modelica.org.

[5] The 20sim tool, available at http://www.20sim.com.

[6] M. A. Groothuis, and J. F. Broenink, "Multi-view Methodology for the Design of Embedded Mechatronic Control Systems", IEEE Conf. on Computer Aided Control Systems Design, Munich, Germany, 2006.

[7] Architecture Analysis & Design Language (AADL) - Modeling Language for Safety-Critical Systems, available at http://www.aadl.info.

[8] Object Management Group (OMG), A UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded systems, Report, 2008.

[9] COMPASS project, available at http://www.compass-research.eu.

[10] CRYSTAL project, available at http://www.crystal-artemis.eu.

[11] International Standard ISO/IEC/IEEE 42010, Systems and Software Engineering - Architecture Description, 1st Ed., 2011.