Technische Universität München

Management of Complex Product Ontologies Using a Web-Based Natural Language Processing Interface

Master Thesis Final Presentation
A B M Junaed, 11.07.2016

Software Engineering for Business Information Systems (sebis)
Department of Informatics
Technische Universität München, Germany

wwwmatthes.in.tum.de

Agenda

1. **Motivation**
   - **Background**
   - **Objectives**
2. **Research questions**
3. **Natural Language Interfaces to Knowledge Bases**
   - **Question-Answering Systems**
   - **Controlled Natural Language**
4. **Semantic wikis**
5. **Tool Comparison**
6. **Web-Based Natural Language Processing Interface**
7. **Evaluation**
8. **Future Work**

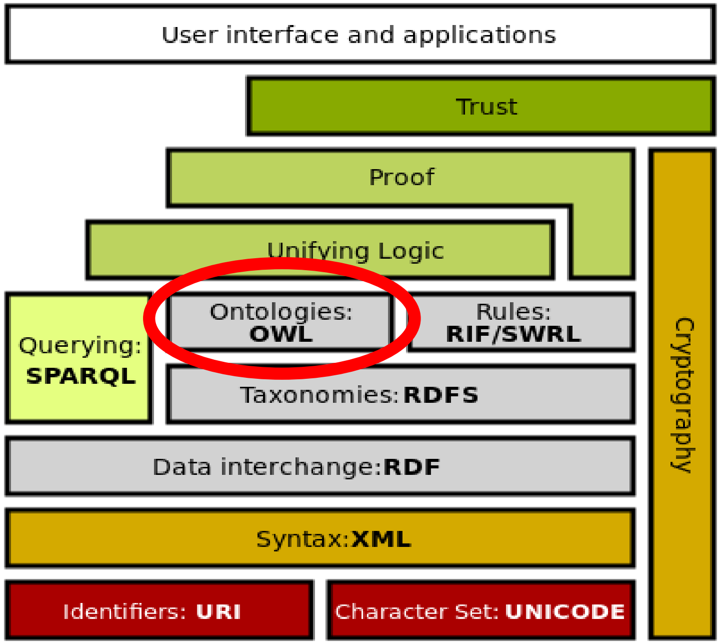
Overview

Complex Products require complex engineering processes

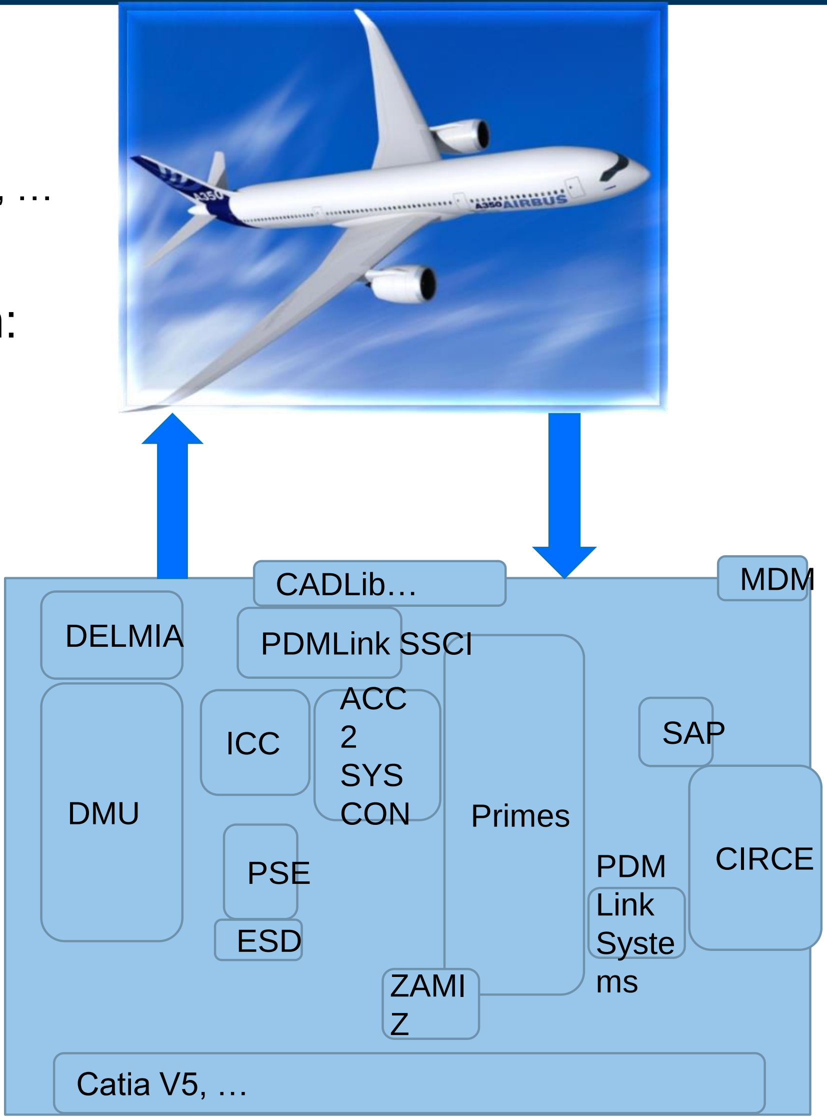
Data in different formats !!!

- Heterogeneous engineering tools => data in different formats !!!
  - Data formats: Relational Databases, XML, CSV, XLS, …
- No unique API to access data

- Key approach at Airbus to solve this problem:
  - Linked data and Semantic web technology



Semantic Web Stack

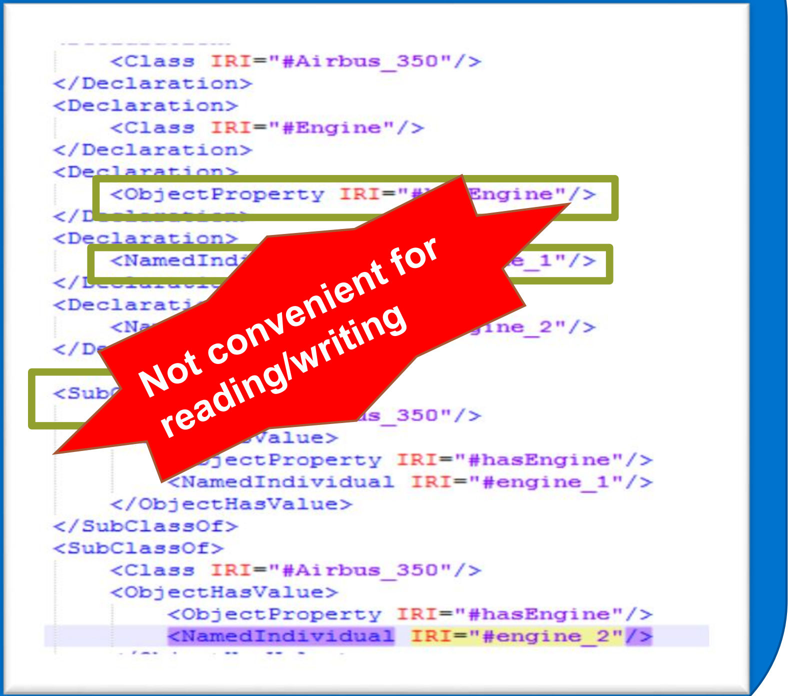


Various tools for aircraft engineering

OWL : Web Ontology Language

- OWL:
  - Used for knowledge representation (KR)
  - Includes descriptions of *classes*, *properties* and their *instances*
  - Based on description logic, so brings reasoning capability
- Very simple Example: Airbus 350 has two engines : engine 1 and engine 2

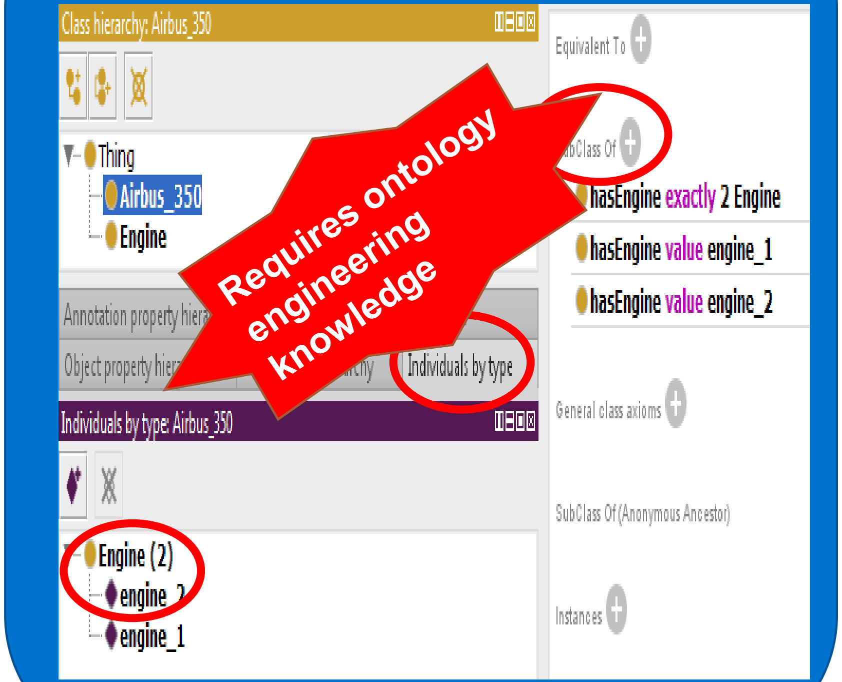
XML Based



```
    <Class IRI="#Airbus_350"/>
</Declaration>
<Declaration>
    <Class IRI="#Engine"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#      Engine"/>
</D
<Declaration>
    <NamedInd                  e_1"/>
</D
<Declarati
    <Na                     gine_2"/>
</De

<Sub
                           s_350"/>
                      Value>
           ectProperty IRI="#hasEngine"/>
        <NamedIndividual IRI="#engine_1"/>
    </ObjectHasValue>
</SubClassOf>
<SubClassOf>
    <Class IRI="#Airbus_350"/>
    <ObjectHasValue>
        <ObjectProperty IRI="#hasEngine"/>
        <NamedIndividual IRI="#engine_2"/>
```

Not convenient for reading/writing

Tool: e.g. Protégé



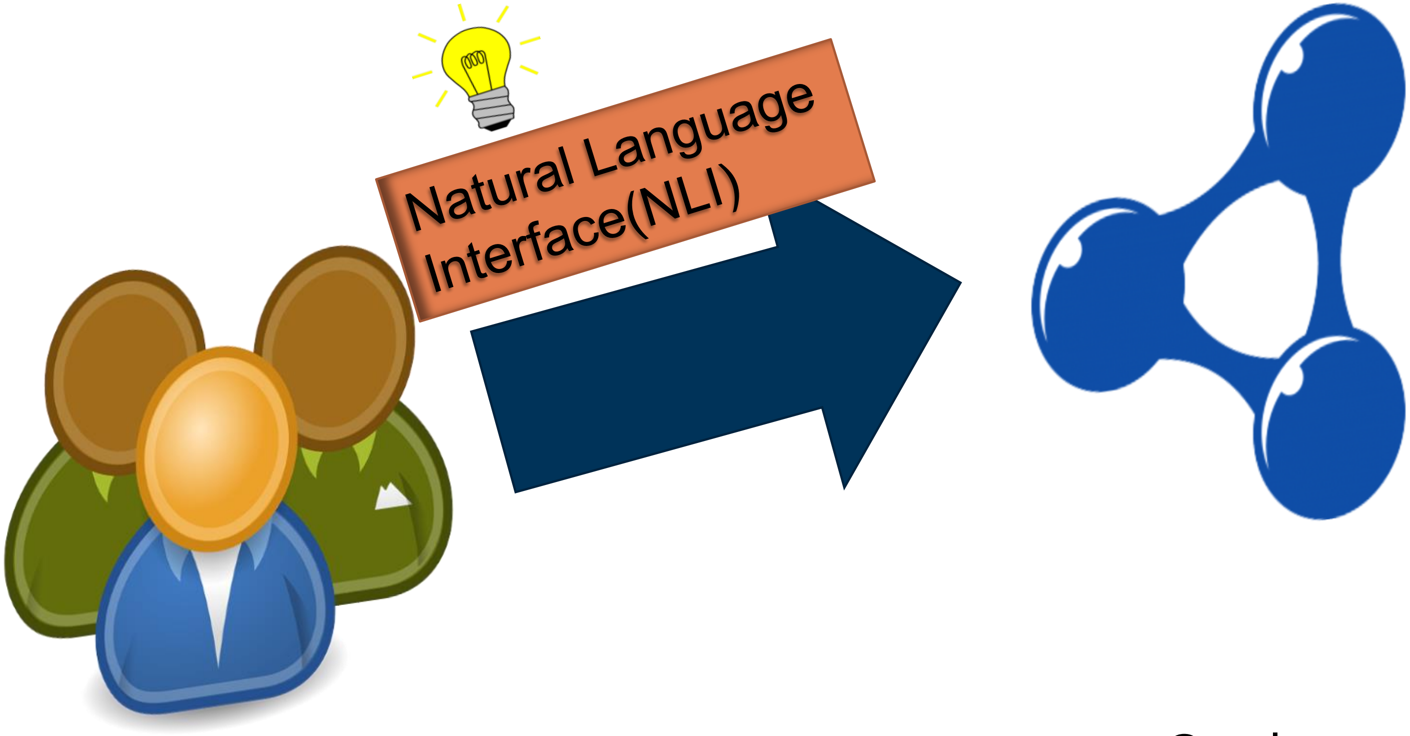
hasEngine exactly 2 Engine

hasEngine value engine_1

hasEngine value engine_2

Requires ontology engineering knowledge

Problems for Domain Experts

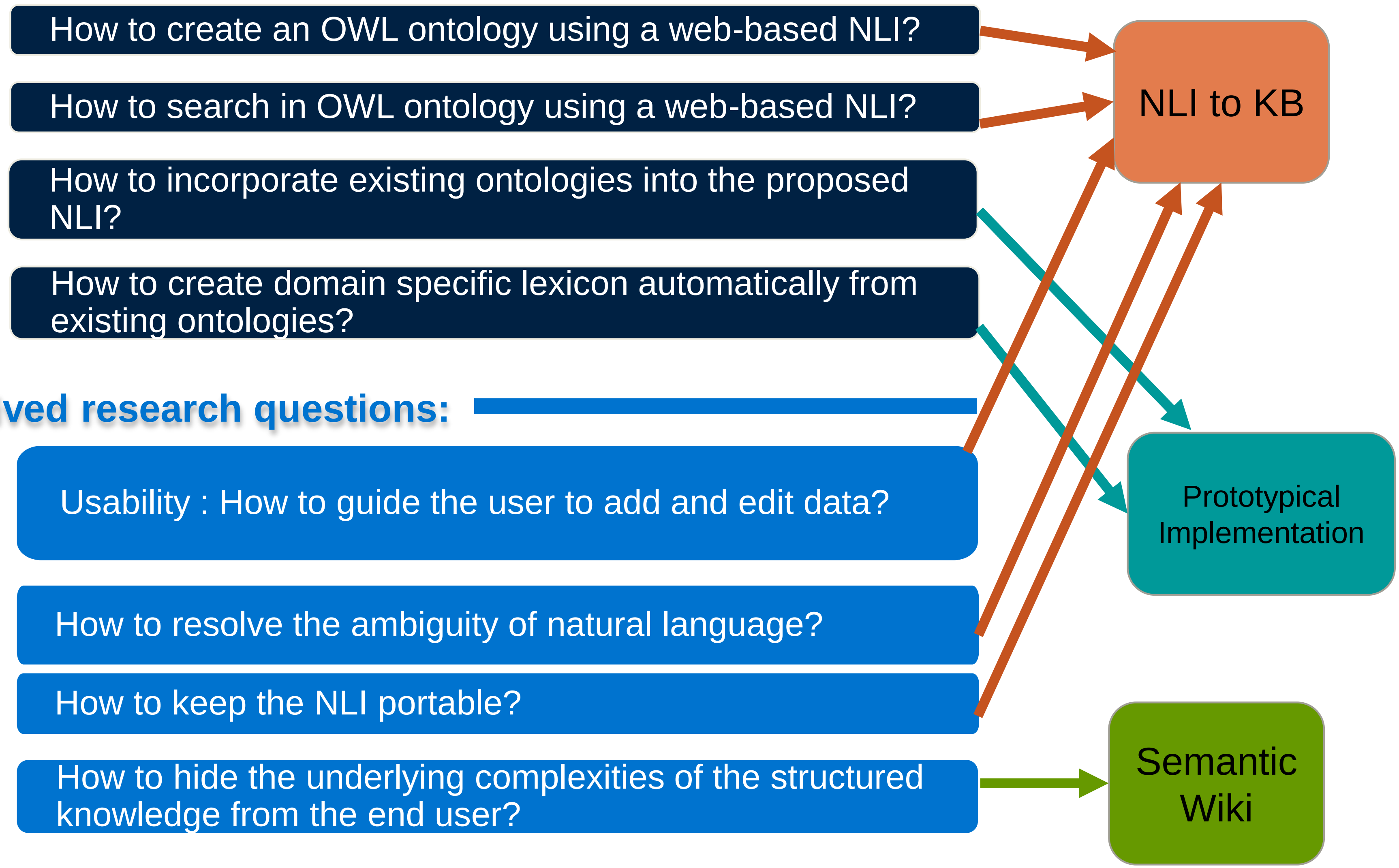


Domain expert

Ontology

Research Questions

- **Major research questions:**



How to create an OWL ontology using a web-based NLI?

How to search in OWL ontology using a web-based NLI?

How to incorporate existing ontologies into the proposed NLI?

How to create domain specific lexicon automatically from existing ontologies?

- **Derived research questions:**

Usability : How to guide the user to add and edit data?

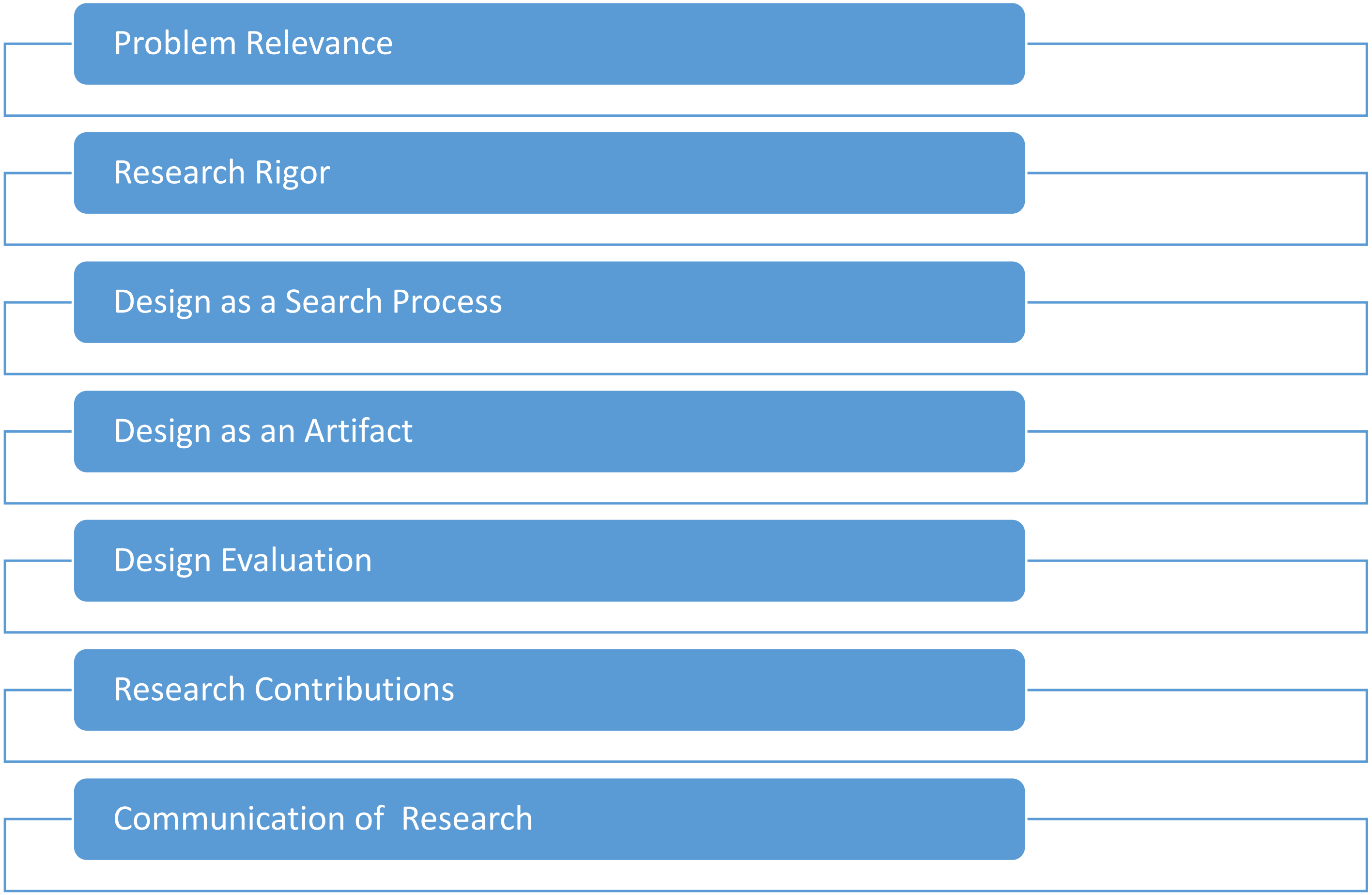
How to resolve the ambiguity of natural language?

How to keep the NLI portable?

How to hide the underlying complexities of the structured knowledge from the end user?

Research Method

- Followed the information systems (IS) research framework by Hevner et al.
  - Set of seven research guidelines



Problem Relevance

Research Rigor

Design as a Search Process

Design as an Artifact

Design Evaluation

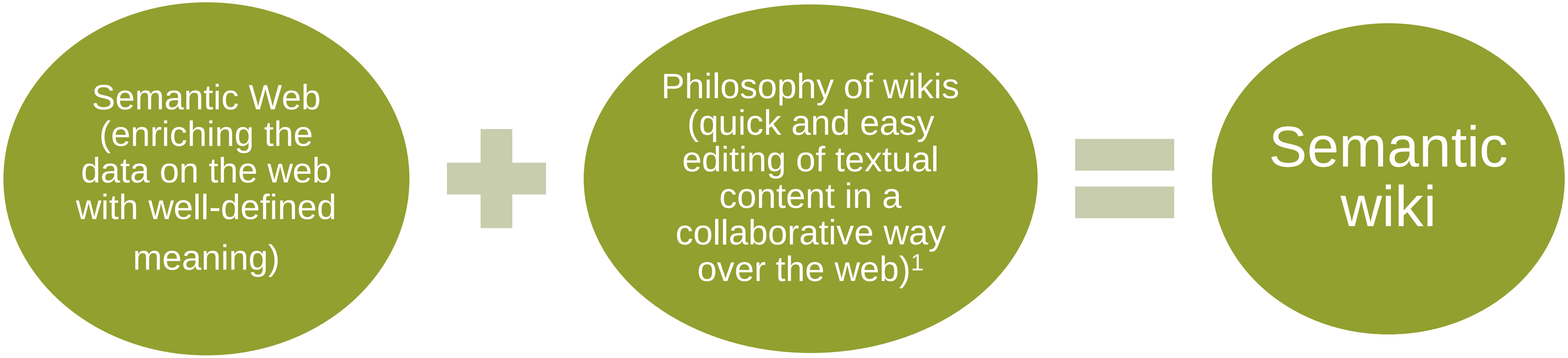
Research Contributions

Communication of Research

Natural Language Interfaces to Knowledge Bases

- Two broad categories:
  1. Question Answering (QA) systems
     - Translate NL into formal query language e.g. SPARQL
     - E.g. Aqualog, NLP-Reduce, FREyA, AutoSparql
  2. Controlled Natural Language (CNL) to work with OWL
     - Grammar and vocabulary are restricted to eliminate or reduce ambiguity
     - E.g. Attempto Controlled English (ACE), Rabbit to OWL Ontology Authoring (ROO)

Semantic wikis



Semantic Web (enriching the data on the web with well-defined meaning)

Philosophy of wikis (quick and easy editing of textual content in a collaborative way over the web)[1]

Semantic wiki

[1] Tobias Kuhn, 2010

Tool Comparison

	Approach	User guidance	Domain Independence	OWL → NL conversion	NL → OWL conversion	Adding data	Updating data	Search	Automatic ambiguity Resolution
AquaLog	QA	-	+/-	-	-	-	-	+	+/-
NLP-Reduce	QA	-	-	-	-	-	-	+	+/-
AutoSPARQL	QA	-	-	-	-	-	-	+	+/-
FREyA	QA	+/-	+	-	-	-	-	+	+/-
ROO	CNL	-	+	-	-	+	+	-	+
ACE	CNL	+ (AceWiki)	+	+/- (OWL-Verbalizer)	+ (AceWiki)	+	+	+ (AceWiki)	+

+ : supported, +/- : partly supported, - : not supported

Solution Approach

OWL-Verbalizer

- OWL→ACE translation

AceWiki

- Provides web-based interface
- ACE as CNL

Limitations of OWL-Verbalizer

- Not compatible with all OWL axioms e.g. Annotation, FunctionalDataProperty …
- Can not handle more than two classes in a DisjointClasses block

Limitations of AceWiki

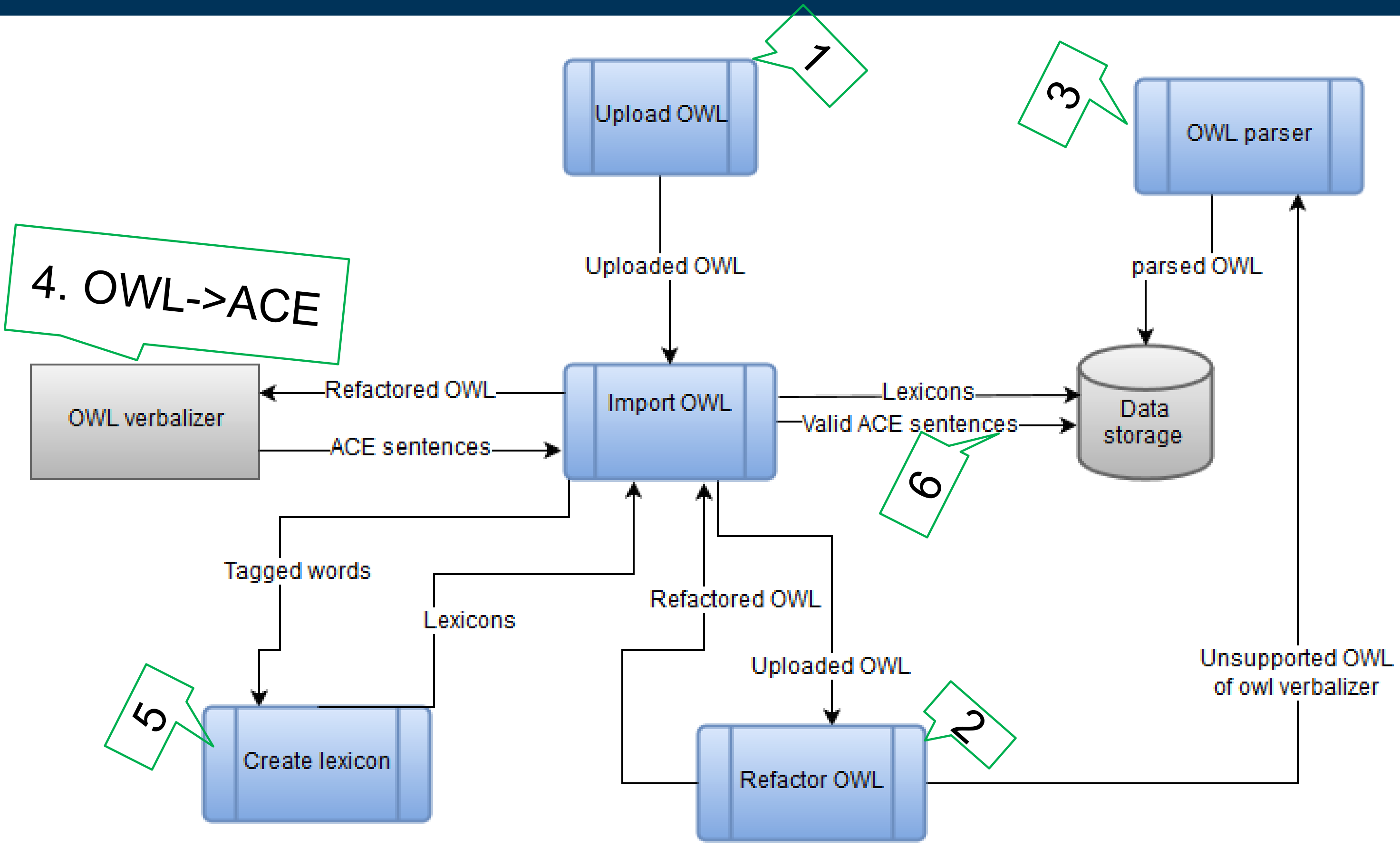
- No import functionality
- Wrong URI
- Floating point numbers not supported
- All ACE sentences are not supported
- Labels and comments from OWL ontology are lost

Implemented New Features

- Based on the limitations of OWL-Verbalizer and AceWiki

- Import functionality
- Auto lexicon creation
- Change grammar to support floating point numbers
- Rewrite DisjointClasses blocks
- Store rdfs:Labels and export them
- Store rdfs:comments and export them
- Store right URI
- Support more data formats

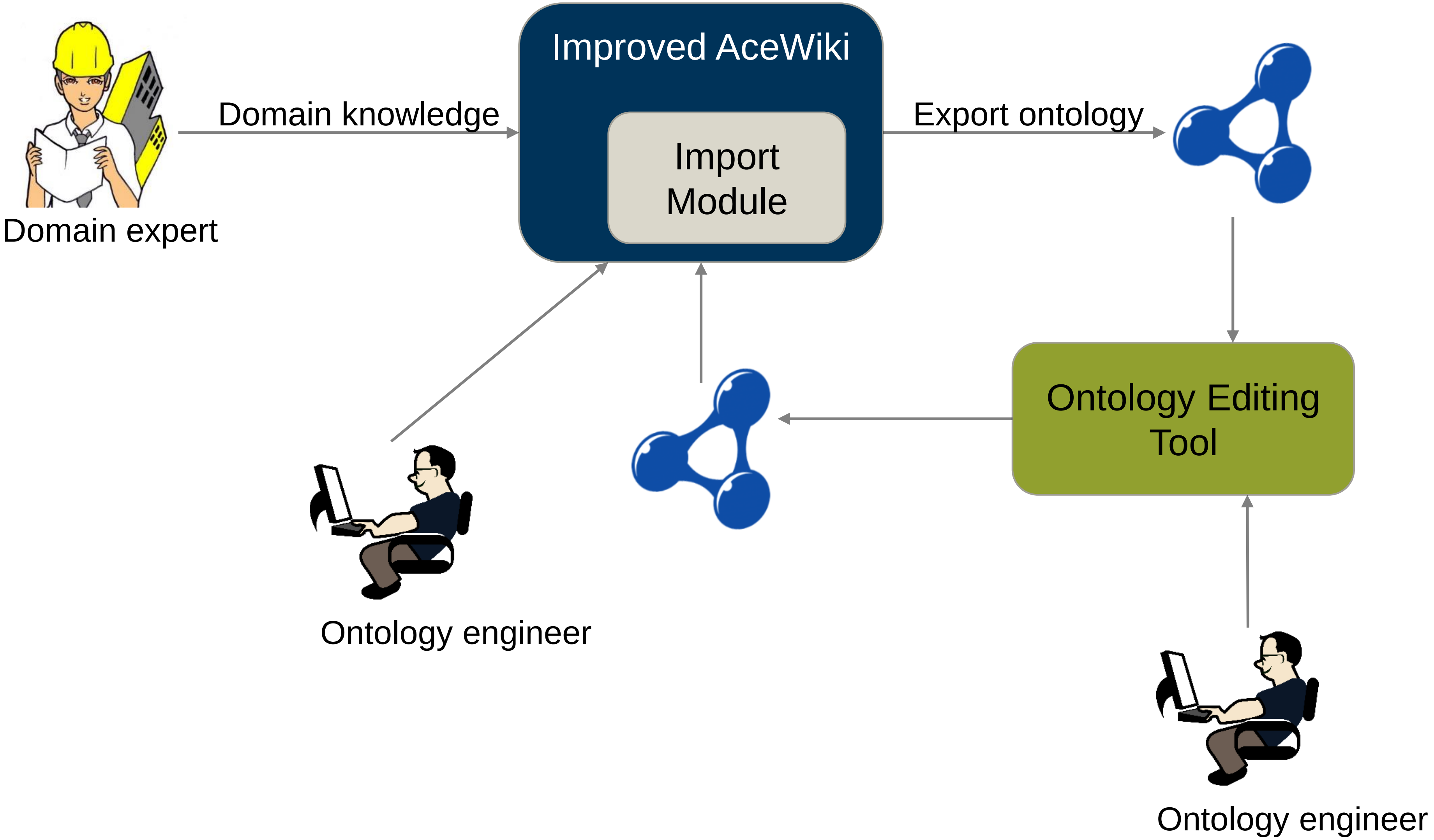
Data Flow Diagram of Implemented Solution



🟦 Implemented components ▭ ACE components

Figure: Level 1 data flow diagram for import functionality and lexicon creation

Ontology Management Workflow

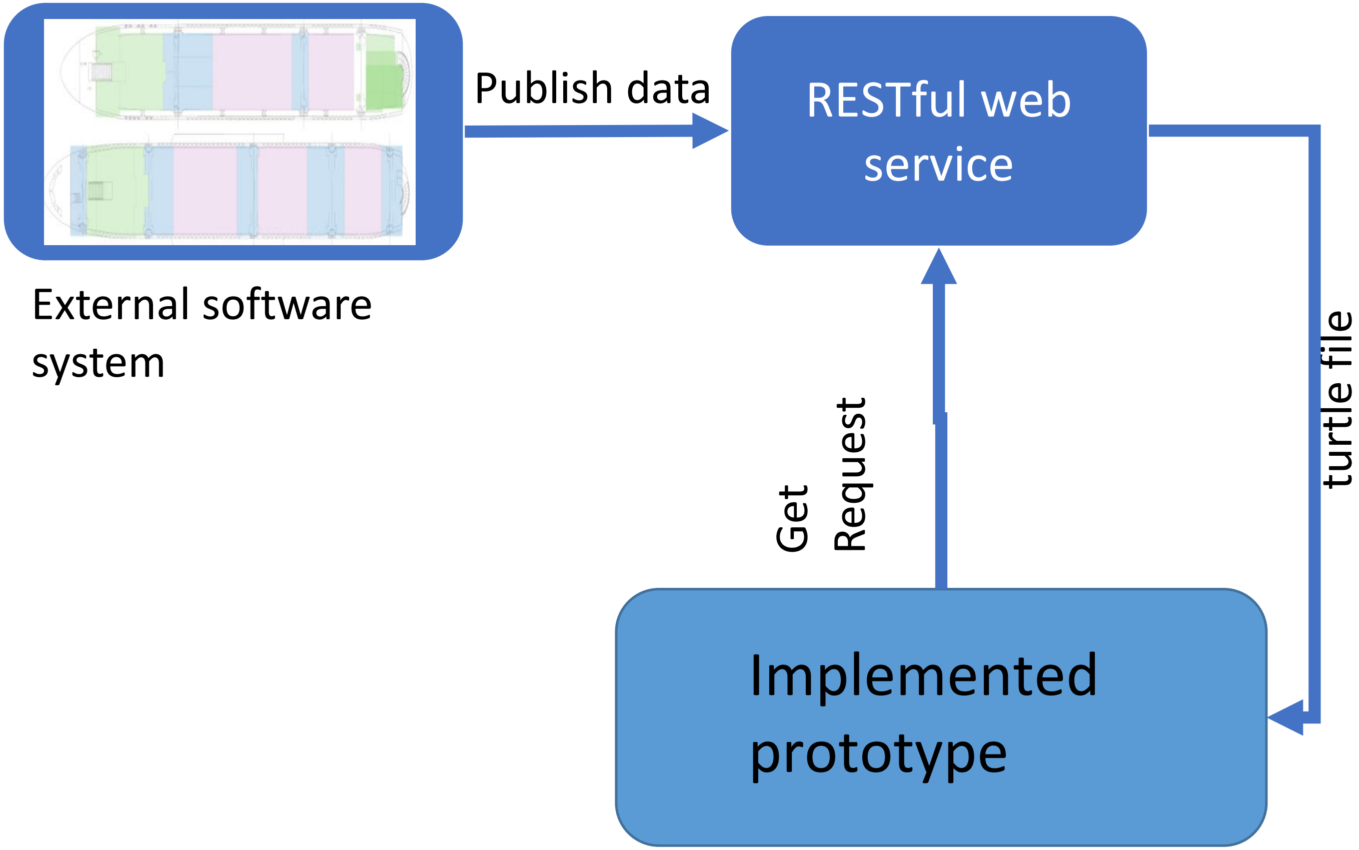


Demo

Functional Evaluation: Results of Functionality and Portability Test

- Successfully handled all the ACE sentences for which we added support
- The prototype is portable
  - No customization is required to work with different OWL ontologies.

Functional Evaluation: Integration With Other Business Solutions



Publish data

RESTful web service

External software system

Get Request

turtle file

Implemented prototype

Qualitative Evaluation: Results of Expert Interview using Questionnaire

Feedback for the Prototype

- Intuitive import functionality
- Search options are helpful
- User guidance: Can be improved by auto-completion

Potential Use Cases

- Managing requirements: Importing verbalized ontology is very helpful
- To quickly create a generic ontology

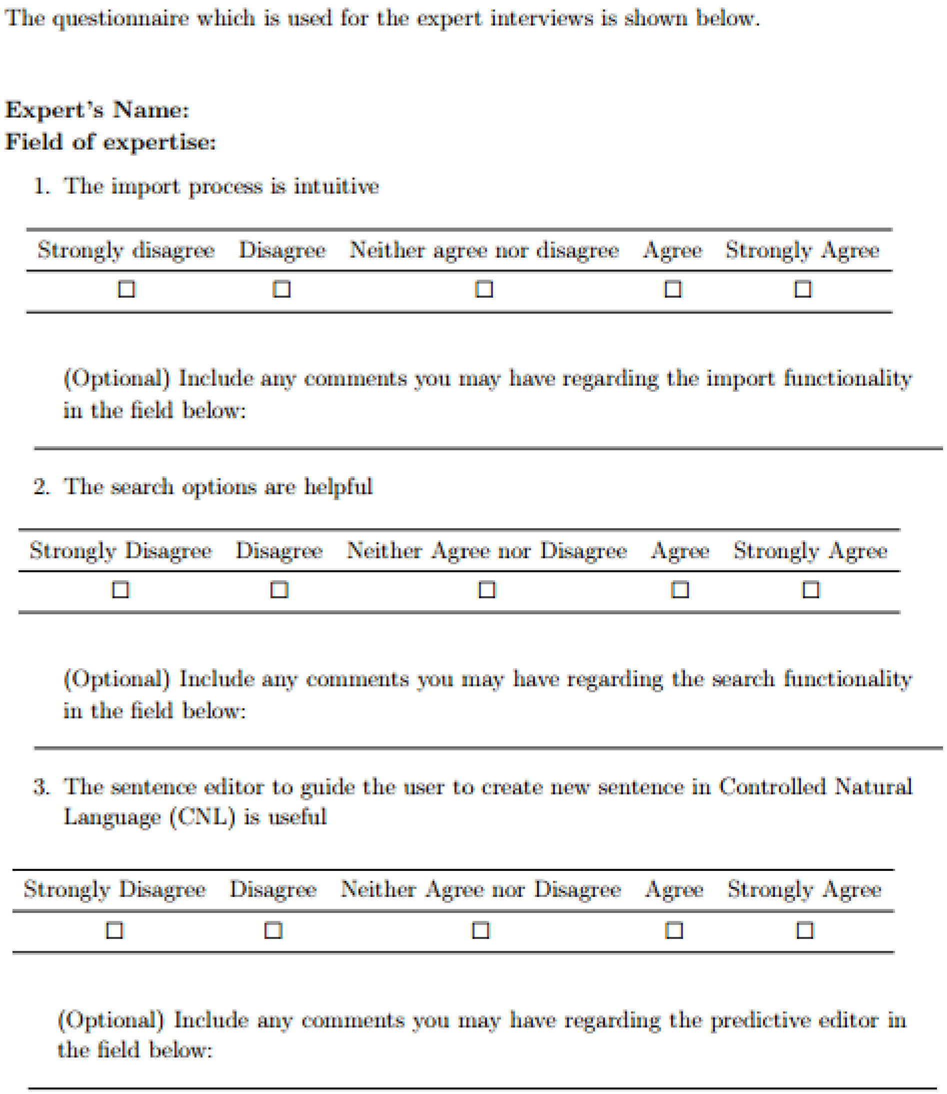
Future Work

- User management and activity logging
- Morphological improvement
- Improving OWL-verbalizer
- Auto-completion
- Potential use cases
  - e.g., managing requirements, model management
  - Prototype can be tailored to work with those use cases in future.

Questions?

Backup slides

Questionnaire



The questionnaire which is used for the expert interviews is shown below.

Expert's Name:
Field of expertise:

1. The import process is intuitive

Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly Agree
□	□	□	□	□

(Optional) Include any comments you may have regarding the import functionality in the field below:

2. The search options are helpful

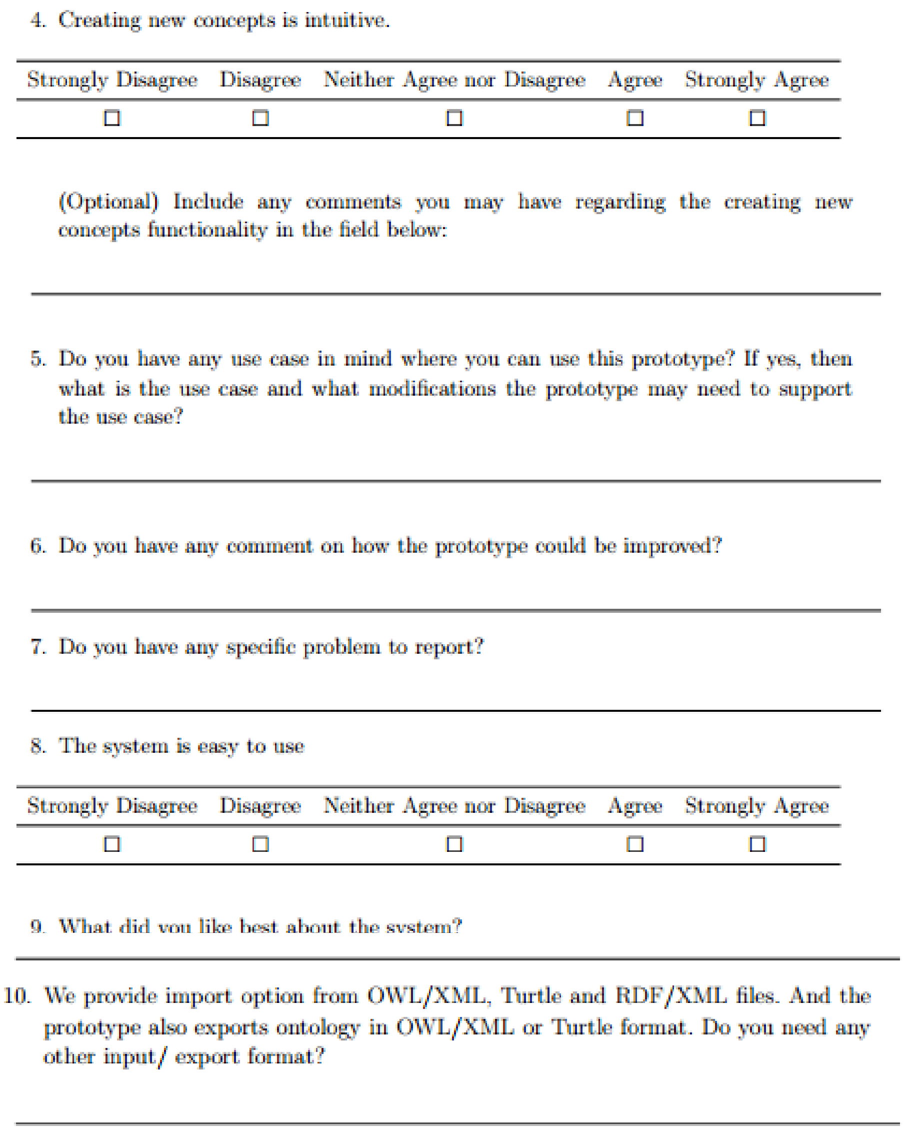
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
□	□	□	□	□

(Optional) Include any comments you may have regarding the search functionality in the field below:

3. The sentence editor to guide the user to create new sentence in Controlled Natural Language (CNL) is useful

Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
□	□	□	□	□

(Optional) Include any comments you may have regarding the predictive editor in the field below:



4. Creating new concepts is intuitive.

Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
□	□	□	□	□

(Optional) Include any comments you may have regarding the creating new concepts functionality in the field below:

5. Do you have any use case in mind where you can use this prototype? If yes, then what is the use case and what modifications the prototype may need to support the use case?

6. Do you have any comment on how the prototype could be improved?

7. Do you have any specific problem to report?

8. The system is easy to use

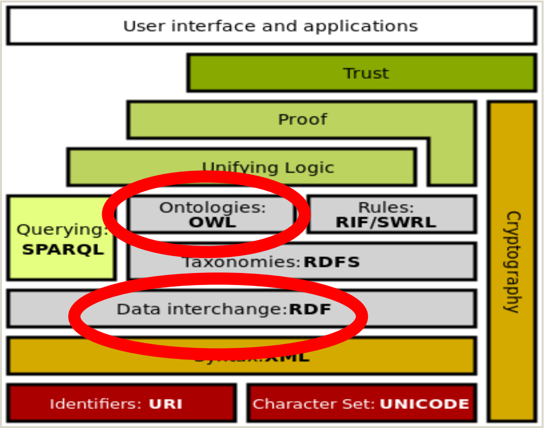
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
□	□	□	□	□

9. What did you like best about the system?

10. We provide import option from OWL/XML, Turtle and RDF/XML files. And the prototype also exports ontology in OWL/XML or Turtle format. Do you need any other input/ export format?

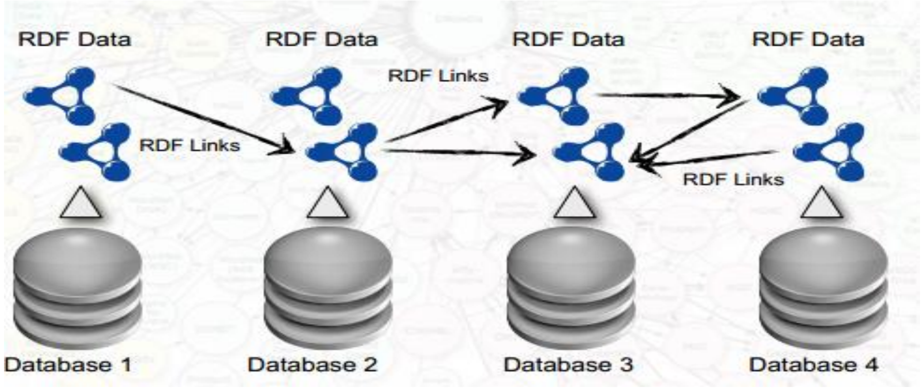
Overview of Semantic web and Linked Data

- Apply Semantic web technologies :
  - To publish data (in RDF format)
  - To draw connections between data sources



Semantic Web Stack

- **Linked Data**
- **Accessible via same kind of API**



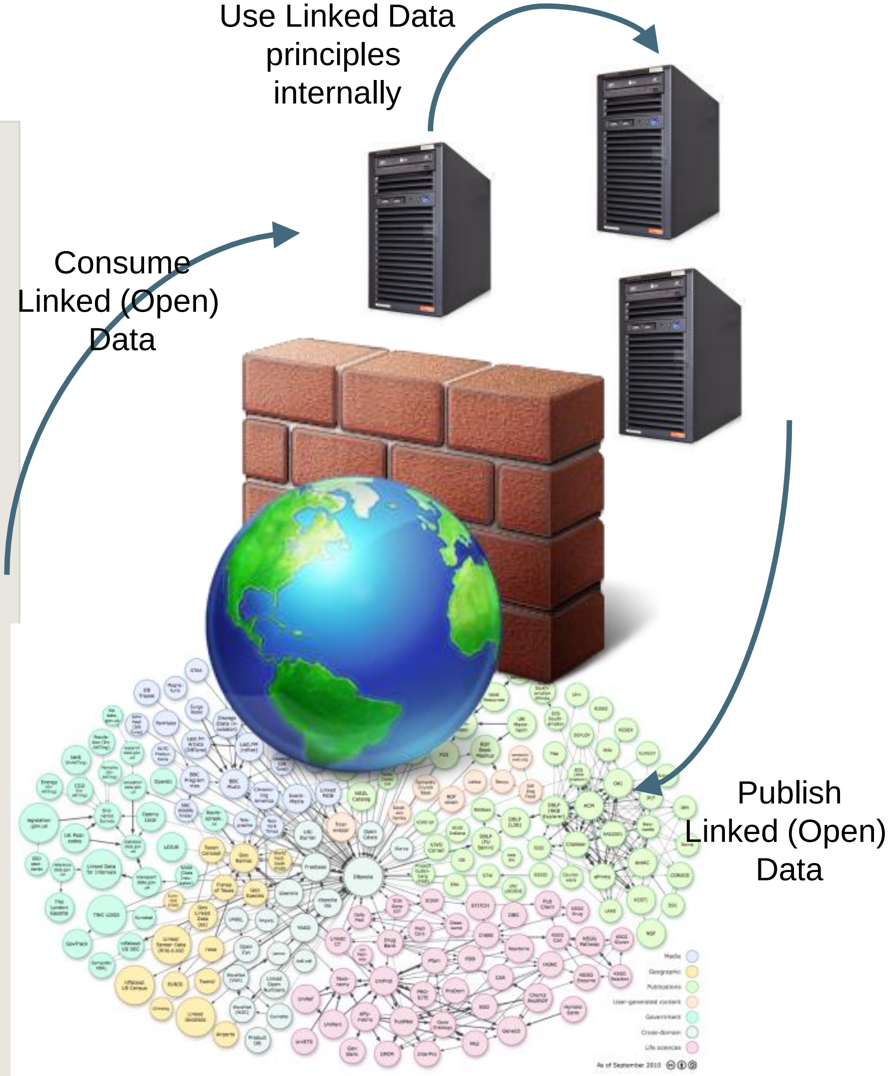
Linked Data is an architectural style for integrating data in the enterprise

1.Standard data access mechanism: **HTTP**

2.Standard address & identifier mechanism: **URIs**

3.Standard data model: **RDF(resurouce description framework)**

4.Include links to other URIs, to discover more things.



Use Linked Data principles internally

Consume Linked (Open) Data

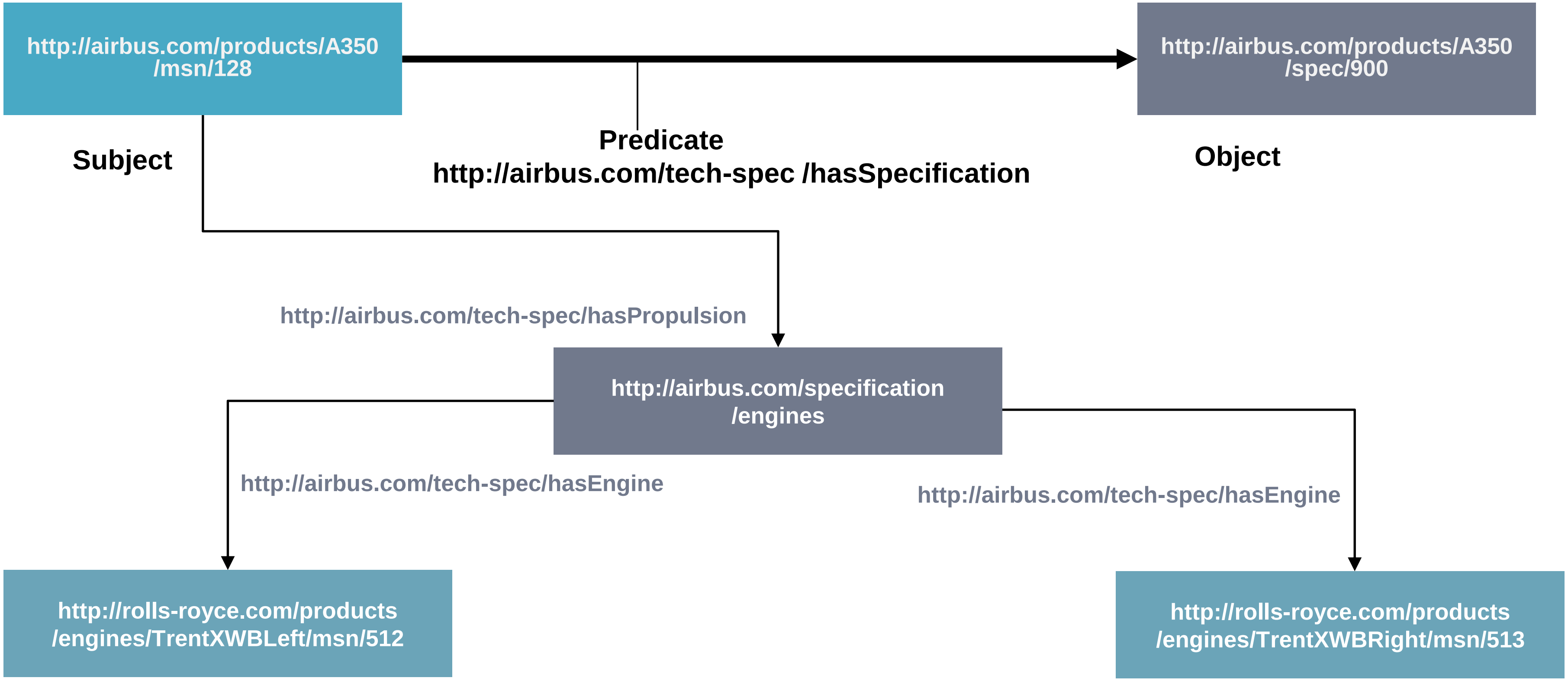
Publish Linked (Open) Data

RDF

Statements (Triple format): Subject + Predicate + Object

- How to present: Airbus A350 with the MSN 128 has the specification 900

- Airbus A350 has two engines, 512 and 513 : manufactured by Rolls Royce with the



http://airbus.com/products/A350 /msn/128
http://airbus.com/products/A350 /spec/900
Subject
Predicate
http://airbus.com/tech-spec /hasSpecification
Object
http://airbus.com/tech-spec/hasPropulsion
http://airbus.com/specification /engines
http://airbus.com/tech-spec/hasEngine
http://airbus.com/tech-spec/hasEngine
http://rolls-royce.com/products /engines/TrentXWBLeft/msn/512
http://rolls-royce.com/products /engines/TrentXWBRight/msn/513

Benefits of Linked Enterprise Data complementary to PRIMES

sebis

Flexibility and Agility

- Schema modifications, e.g. an additional column of RDBMS take months to authorize; adding a triple is simple w/ RDF
- Works in an incremental fashion
- Easy integration of new concepts

Links and URIs

- Universal Identification through global identifier
- „Foreign keys“ to tables out of authorization

Scalability

- Planetary scale (see the LOD cloud)
- Management of billions of data triples
- Cooperation w/o coordination

RDF (graphs) as data model

- General method for conceptual description and modeling of information
- Don’t confuse data models w/ data serialization formats!

Economic aspects

- Costs for functional updates …
- Independence of proprietary technologies and data formats
- Sustainability of the web technology approach (tools are changing, www basics probably not)
- All the needed technology is already in place and tested on a larger scale
- Global approach not limited to a specific step in a product lifecycle management

Knowledge Generation

- Generation of implicit knowledge through meta data
- Generation of automated rule checks

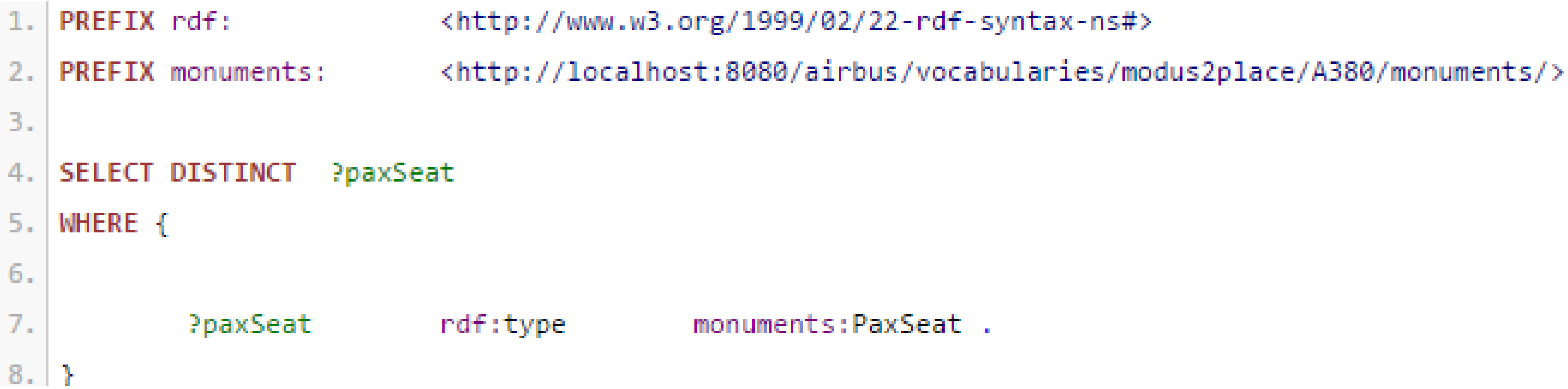
Networking

- Content negotiation for different roles
- Authentication, access control and secure communication through standard web technologies
- Event notification based on standard enterprise communication (E-Mail, etc.)

Search

- **SPARQL**:
  - To query OWL
  - We need to query also !

Example: extract all Passenger Seats:

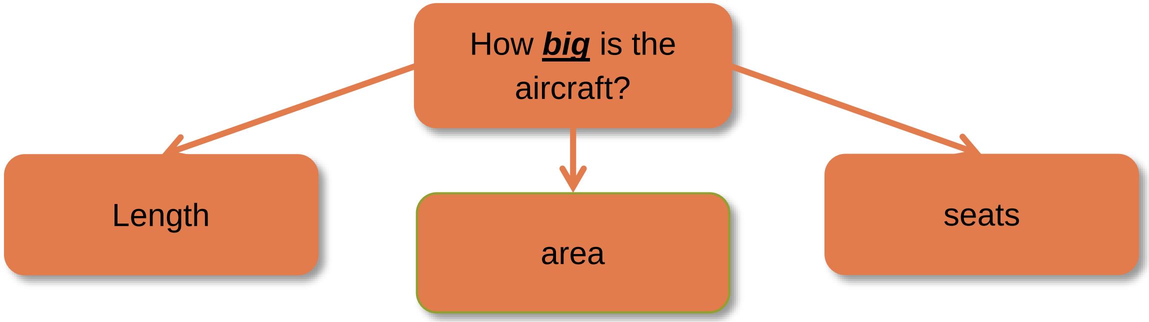


```
PREFIX rdf:             <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX monuments:       <http://localhost:8080/airbus/vocabularies/modus2place/A380/monuments/>

SELECT DISTINCT  ?paxSeat
WHERE {

        ?paxSeat        rdf:type        monuments:PaxSeat .
}
```

But again, **not convenient for end users, they have to learn SPARQL!**

Challenges of NLI: **Ambiguity**

- **Ambiguity**: One query, different meanings
  - depending on:
    - » context
    - » also on ontology structure.

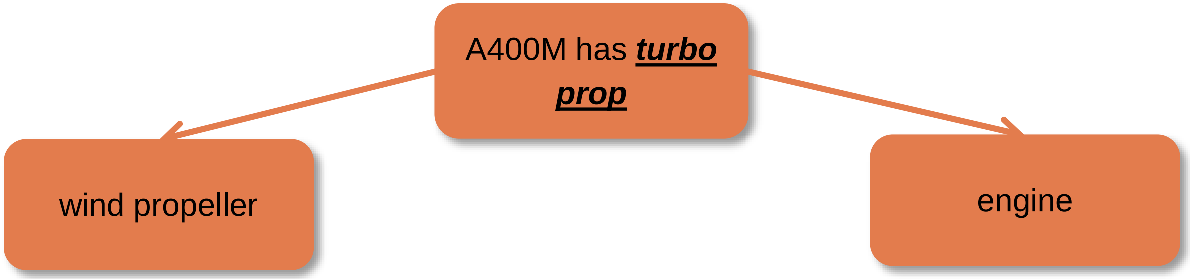


How ***big*** is the aircraft?

Length

area

seats



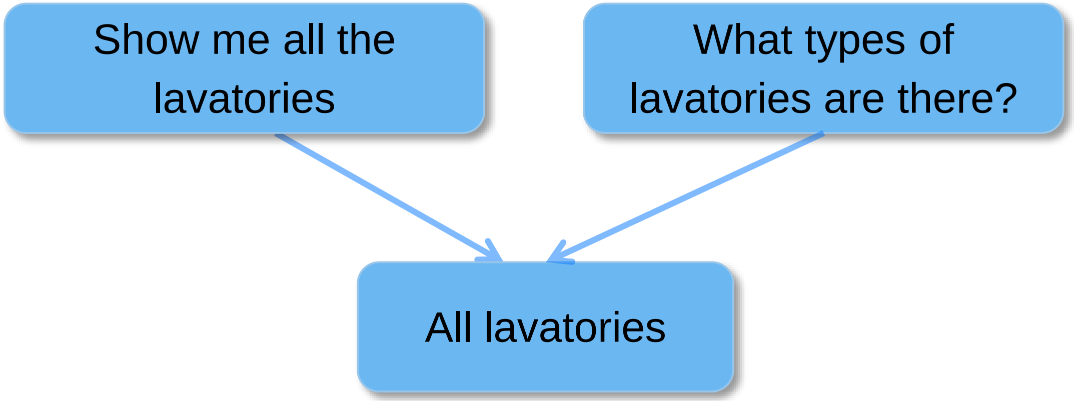
A400M has ***turbo prop***

wind propeller

engine

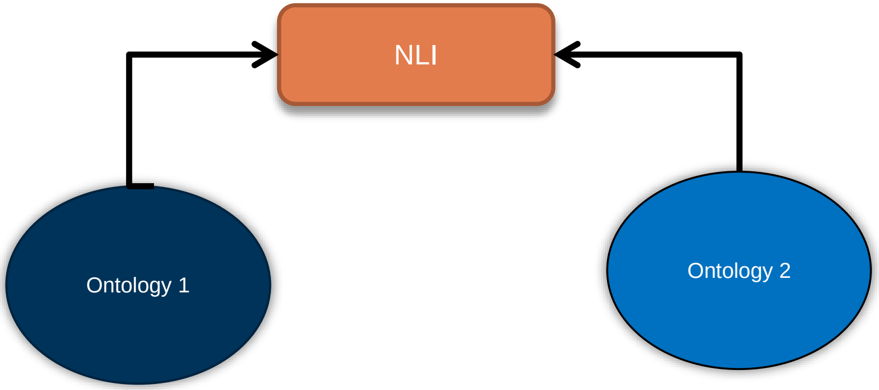
Challenges of NLI: **Expressiveness**

- **Expressiveness**/ **Robustness**:
  - Same meaning, different sentences


Show me all the lavatories
What types of lavatories are there?
All lavatories

Challenges of NLI: **Portability**

- **Portability**: To easily port new ontologies



Challenges of NLI: **Other**

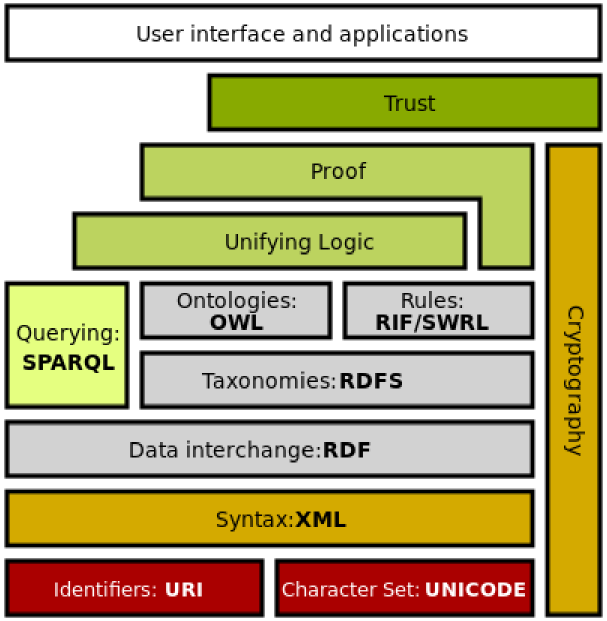
- **Guiding** the user through the process of formulating queries.

- Keeping the supported language **intuitive**.
- **Hiding complexities**: Showing results without imposing underlying complexities of the structured knowledge to user

Semantic web Basics

- **Semantic web standards**
  - Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries.

- RDF(Resource Description Framework):
  - to create in triples statements
  - to represent information about resources in the form of graph
- RDF Schema (RDFS):
  - possible to create hierarchies of classes and properties.
- Web Ontology Language (OWL):
  - extends RDFS to describe semantics
    - such as cardinality, restrictions of values, or characteristics of properties such as transitivity.
    - based on description logic, so brings reasoning power
- SPARQL:
  - to query RDF-based data (i.e., including RDFS and OWL)



User interface and applications

Trust

Proof

Unifying Logic

Querying: **SPARQL**

Ontologies: **OWL**

Rules: **RIF/SWRL**

Taxonomies: **RDFS**

Data interchange: **RDF**

Syntax: **XML**

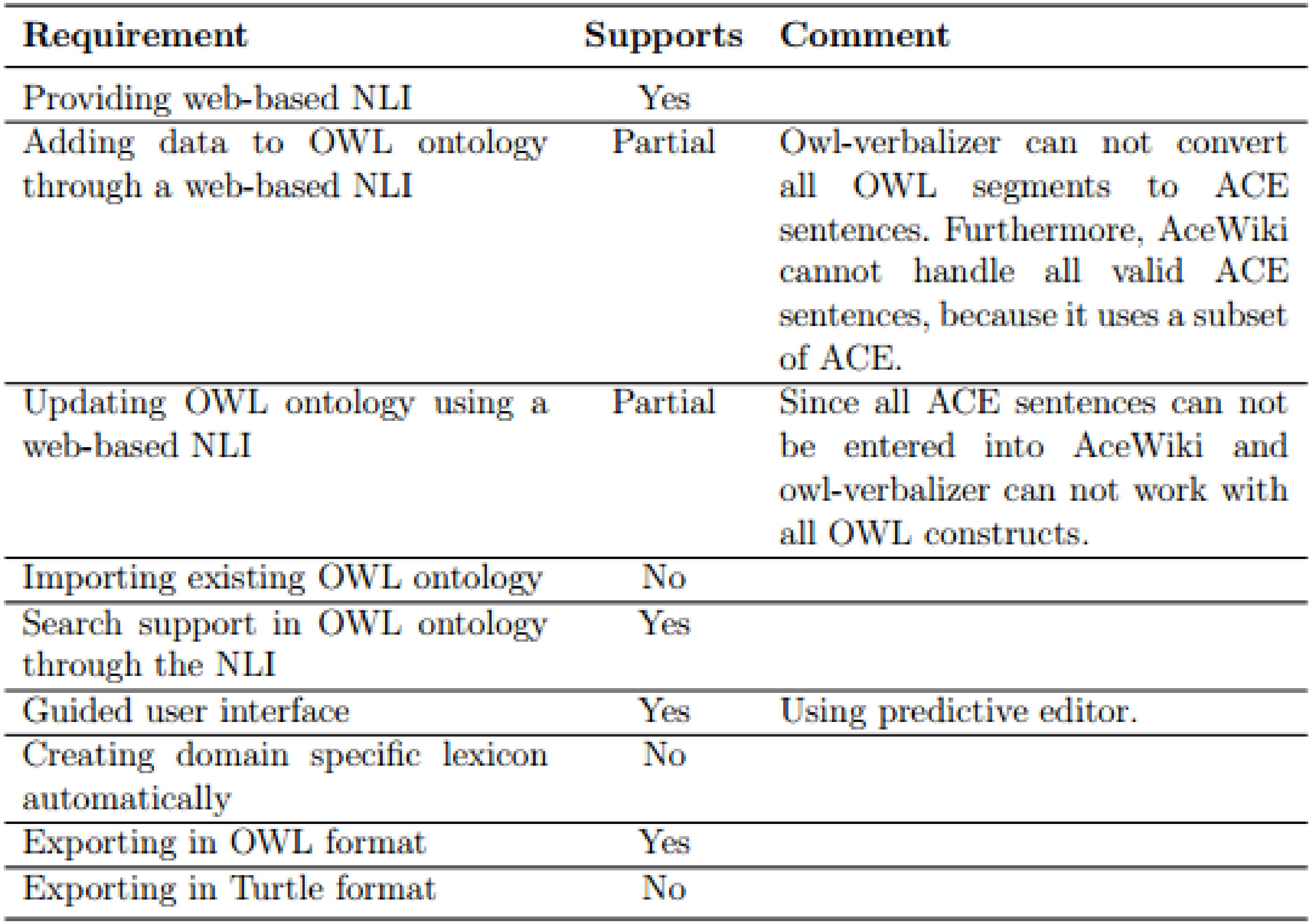
Identifiers: **URI**

Character Set: **UNICODE**

Cryptography

Semantic Web Stack

6. Implementation Details: Evaluation of AceWiki



Requirement	Supports	Comment
Providing web-based NLI	Yes	
Adding data to OWL ontology through a web-based NLI	Partial	Owl-verbalizer can not convert all OWL segments to ACE sentences. Furthermore, AceWiki cannot handle all valid ACE sentences, because it uses a subset of ACE.
Updating OWL ontology using a web-based NLI	Partial	Since all ACE sentences can not be entered into AceWiki and owl-verbalizer can not work with all OWL constructs.
Importing existing OWL ontology	No	
Search support in OWL ontology through the NLI	Yes	
Guided user interface	Yes	Using predictive editor.
Creating domain specific lexicon automatically	No	
Exporting in OWL format	Yes	
Exporting in Turtle format	No	

Limitations of the Solution Approach

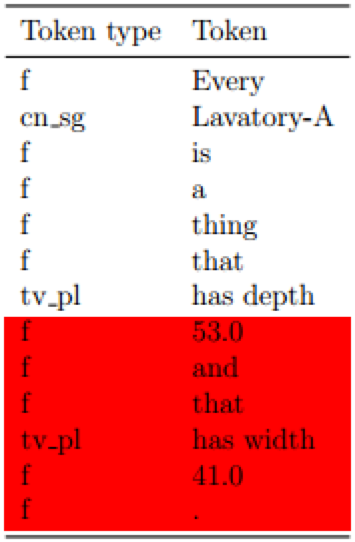
1. No import functionality in AceWiki
2. Automatic lexicon creation is not supported
3. AceWiki can not work with floating point numbers
4. OWL-Verbalizer can not handle more than two classes in a DisjointClasses block
5. OWL-Verbalizer can not verbalize labels and comments from OWL ontology are not verbalized and are not stored in Acewiki
6. Wrong URI: If there is an import statement in OWL ontology, then URI for the imported classes are not the same as the base URI of initial ontology, but AceWiki has no way to define different URI for those imported classes
7. All ACE sentences are not supported in AceWiki

Limitations of OWL-verbalizer

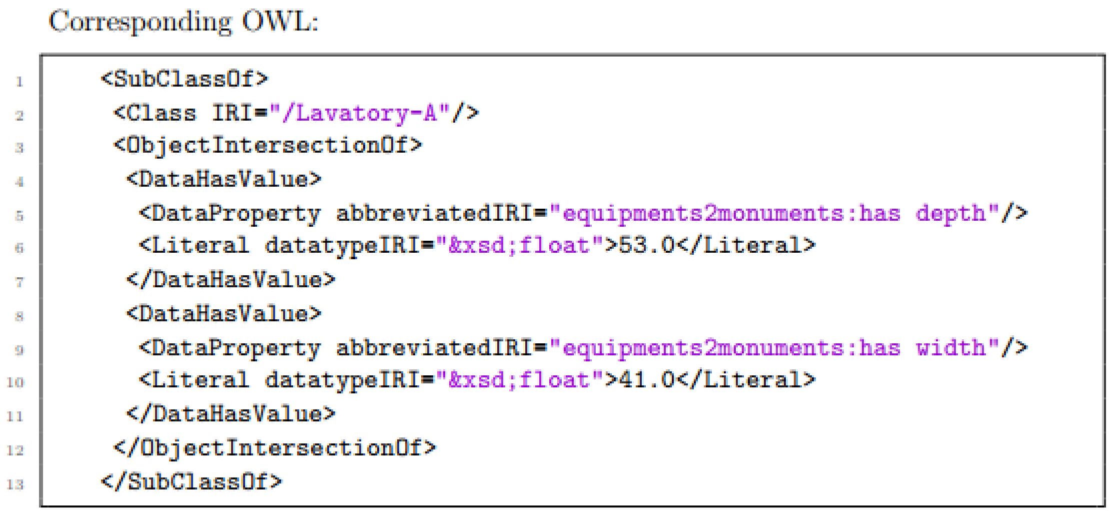
- not compatible with all OWL axioms
  - For this reason, some of the OWL axioms could not be converted to ACE sentence
- owl properties which OWL-verbalizer can not handle:
  - SubDataPropertyOf
  - FunctionalDataProperty
  - DataPropertyRange
  - DLSafeRule
  - DatatypeDefinition
  - ObjectIntersectionOf
  - DataAllValuesFrom
  - DataOneOf
  - DataExactCardinality
  - EquivalentClasses
  - Annotation

Unsuppoerted ACE sentences in AceWiki

1. Unsupported ACE sentences in AceWiki. From the red portion, it is not possible to write the sentence in AceWiki since AceWiki does not support floating point number.



Token type	Token
f	Every
cn_sg	Lavatory-A
f	is
f	a
f	thing
f	that
tv_pl	has depth
f	53.0
f	and
f	that
tv_pl	has width
f	41.0
f	.

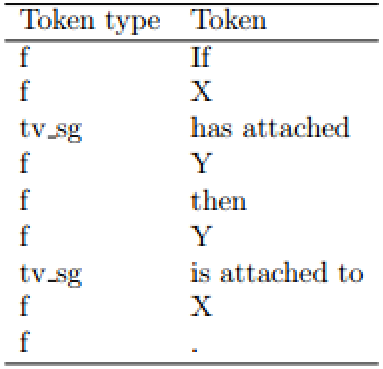


Corresponding OWL:

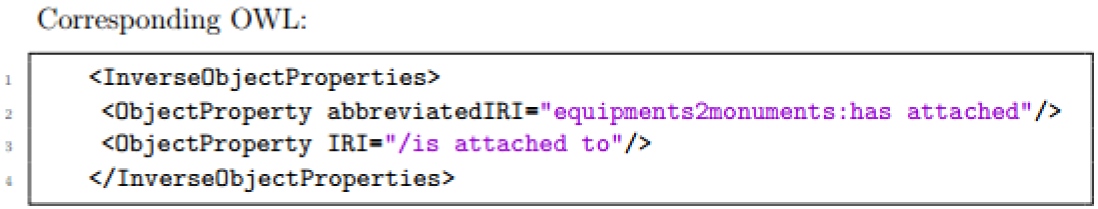
```
<SubClassOf>
 <Class IRI="/Lavatory-A"/>
 <ObjectIntersectionOf>
  <DataHasValue>
   <DataProperty abbreviatedIRI="equipments2monuments:has depth"/>
   <Literal datatypeIRI="&xsd;float">53.0</Literal>
  </DataHasValue>
  <DataHasValue>
   <DataProperty abbreviatedIRI="equipments2monuments:has width"/>
   <Literal datatypeIRI="&xsd;float">41.0</Literal>
  </DataHasValue>
 </ObjectIntersectionOf>
</SubClassOf>
```

Unsupported ACE sentences in AceWiki

2.. Conditional sentence is not supported in AceWiki



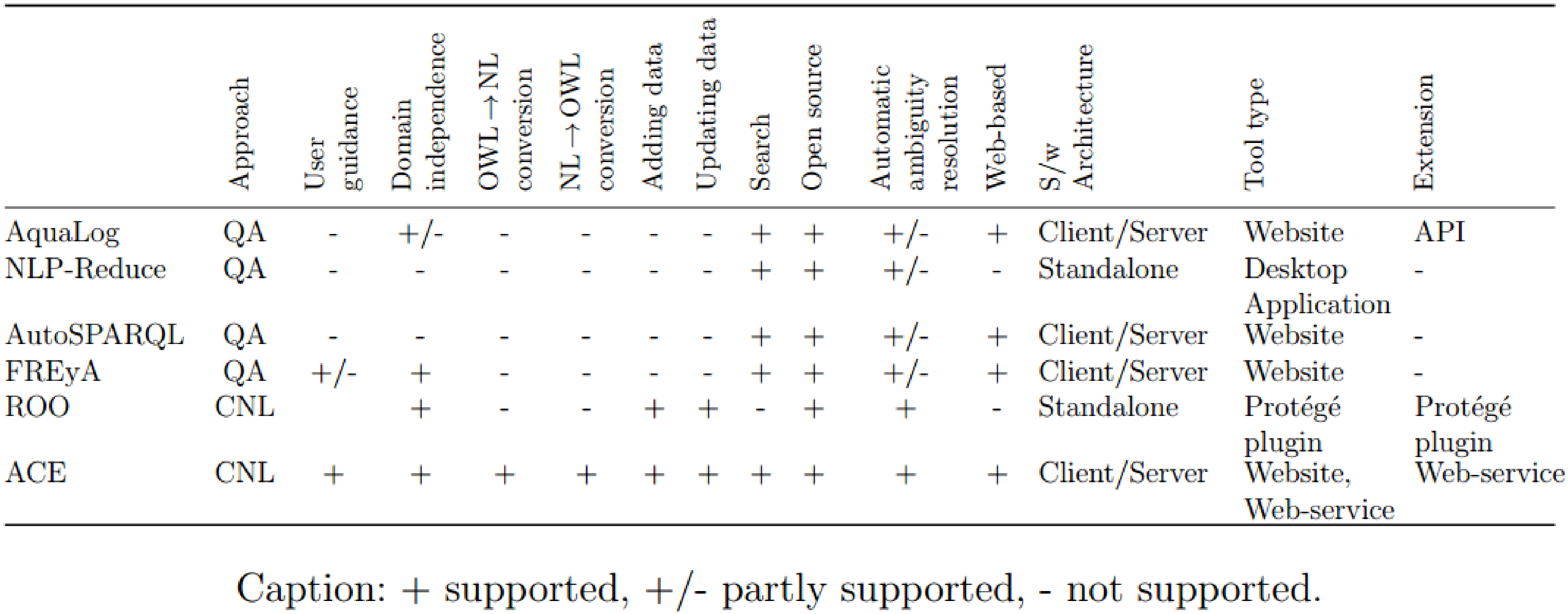
Token type	Token
f	If
f	X
tv_sg	has attached
f	Y
f	then
f	Y
tv_sg	is attached to
f	X
f	.



Corresponding OWL:

```
<InverseObjectProperties>
 <ObjectProperty abbreviatedIRI="equipments2monuments:has attached"/>
 <ObjectProperty IRI="/is attached to"/>
</InverseObjectProperties>
```

Tool Comparison



	Approach	User guidance	Domain independence	OWL→NL conversion	NL→OWL conversion	Adding data	Updating data	Search	Open source	Automatic ambiguity resolution	Web-based	S/w Architecture	Tool type	Extension
AquaLog	QA	-	+/-	-	-	-	-	+	+	+/-	+	Client/Server	Website	API
NLP-Reduce	QA	-	-	-	-	-	-	+	+	+/-	-	Standalone	Desktop Application	-
AutoSPARQL	QA	-	-	-	-	-	-	+	+	+/-	+	Client/Server	Website	-
FREyA	QA	+/-	+	-	-	-	-	+	+	+/-	+	Client/Server	Website	-
ROO	CNL		+	-	-	+	+	-	+	+	-	Standalone	Protégé plugin	Protégé plugin
ACE	CNL	+	+	+	+	+	+	+	+	+	+	Client/Server	Website, Web-service	Web-service

Caption: + supported, +/- partly supported, - not supported.

7. Evaluation Methodology

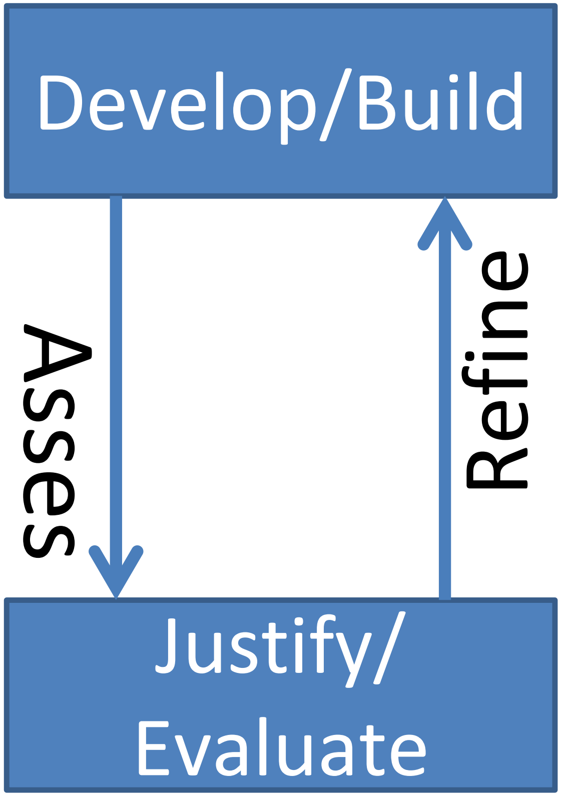


Figure: Develop/Build and Justify/Evaluate cycles within the research group to build the final artifact

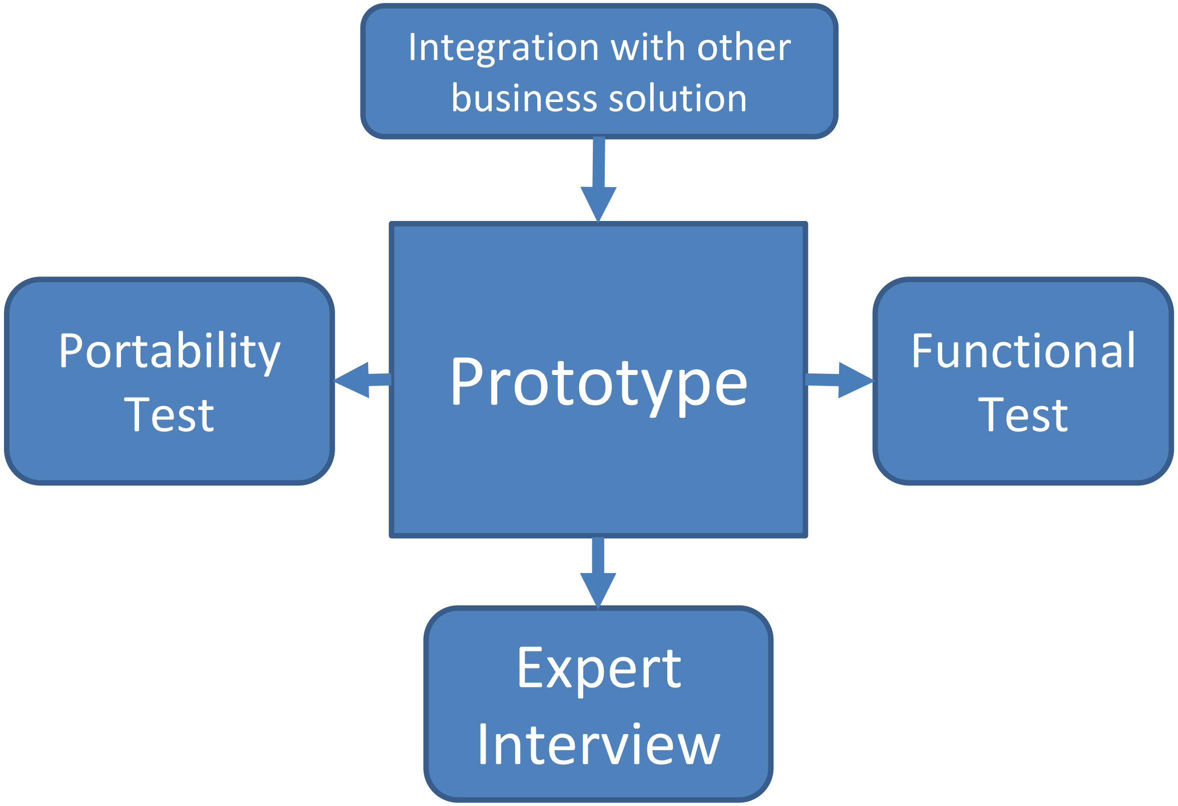
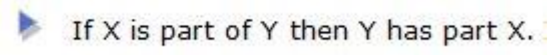


Figure: Final evaluation conducted by

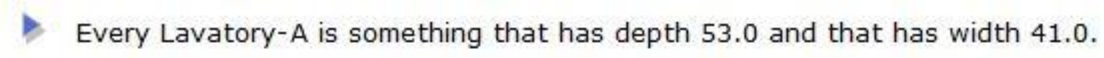
- five expert interviews,
- functional test
- portability test and
- integrating with other business solutions

Screenshot of the Prototype



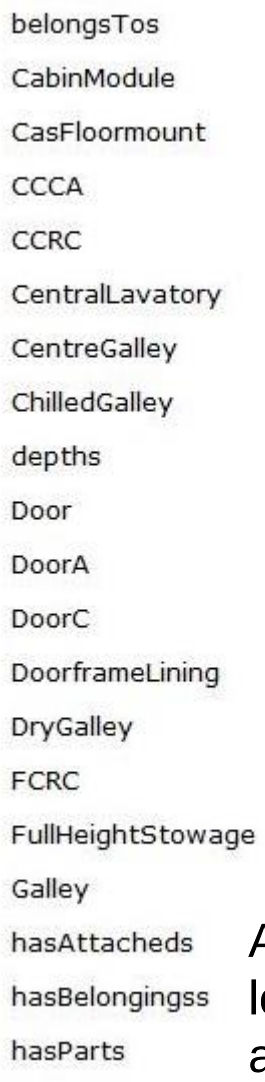
If X is part of Y then Y has part X.

Our prototype supports conditional sentences



Every Lavatory-A is something that has depth 53.0 and that has width 41.0.

Our prototype supports floating point numbers



belongsTos

CabinModule

CasFloormount

CCCA

CCRC

CentralLavatory

CentreGalley

ChilledGalley

depths

Door

DoorA

DoorC

DoorframeLining

DryGalley

FCRC

FullHeightStowage

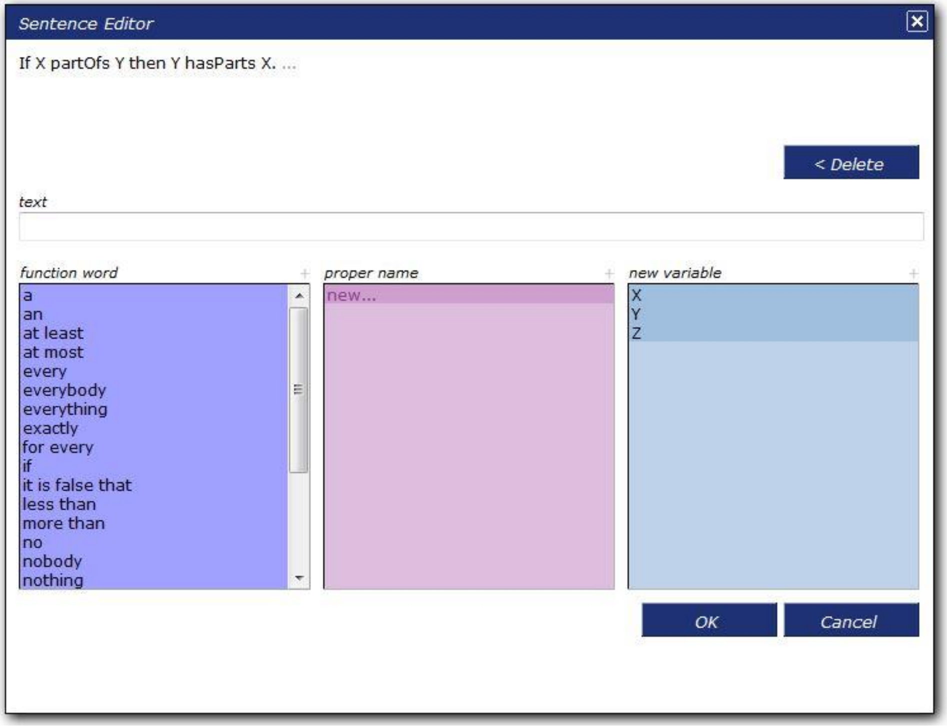
Galley

hasAttacheds

hasBelongingss

hasParts

A screenshot of the list of lexicons which are created automatically while importing an ontology

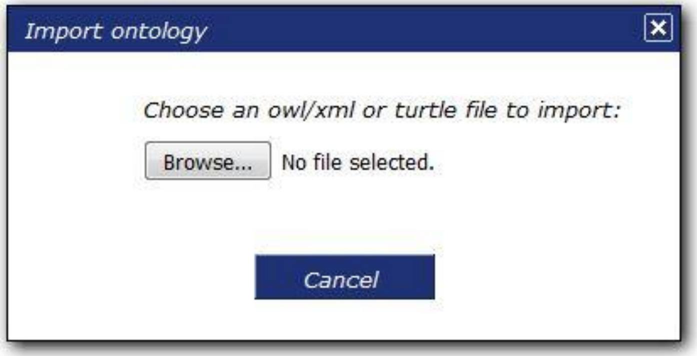


If X partOfs Y then Y hasParts X. ...

function word	proper name	new variable
a	new...	X
an		Y
at least		Z
at most		
every		
everybody		
everything		
exactly		
for every		
if		
it is false that		
less than		
more than		
no		
nobody		
nothing		

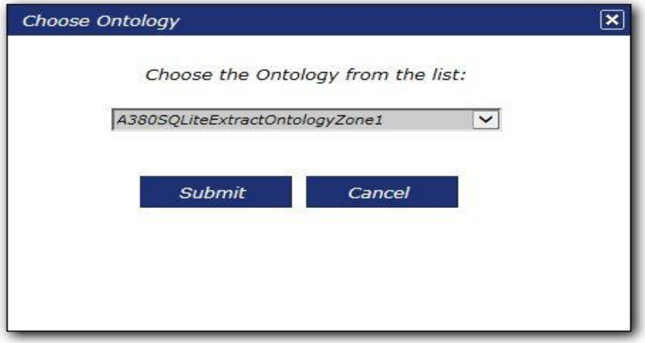
Screenshots showing the improved predictive editor which supports if-then and floating point numbers

Screenshots of the Prototype



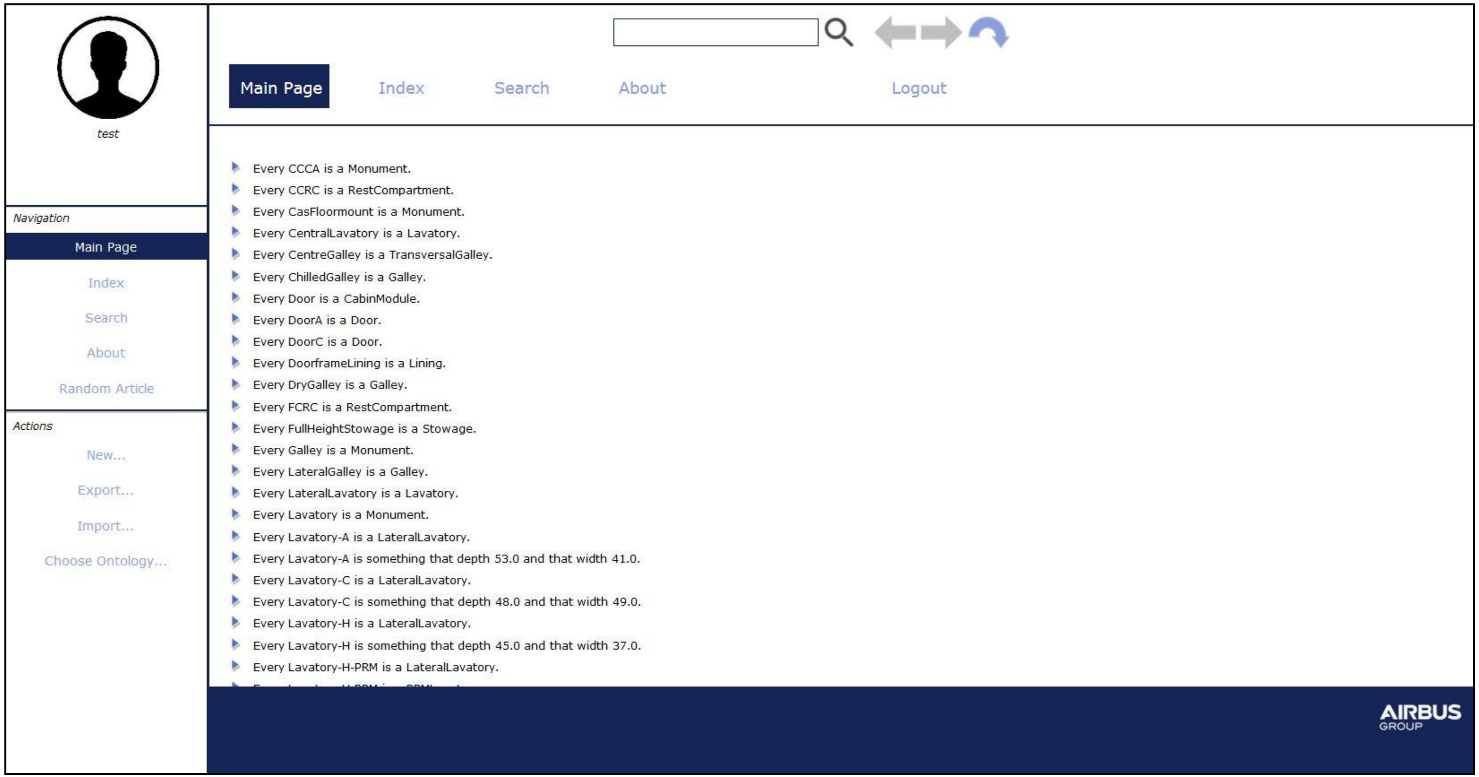
Choose an owl/xml or turtle file to import:

No file selected.



Choose the Ontology from the list:

A380SQLiteExtractOntologyZone1



- Every CCCA is a Monument.
- Every CCRC is a RestCompartment.
- Every CasFloormount is a Monument.
- Every CentralLavatory is a Lavatory.
- Every CentreGalley is a TransversalGalley.
- Every ChilledGalley is a Galley.
- Every Door is a CabinModule.
- Every DoorA is a Door.
- Every DoorC is a Door.
- Every DoorframeLining is a Lining.
- Every DryGalley is a Galley.
- Every FCRC is a RestCompartment.
- Every FullHeightStowage is a Stowage.
- Every Galley is a Monument.
- Every LateralGalley is a Galley.
- Every LateralLavatory is a Lavatory.
- Every Lavatory is a Monument.
- Every Lavatory-A is a LateralLavatory.
- Every Lavatory-A is something that depth 53.0 and that width 41.0.
- Every Lavatory-C is a LateralLavatory.
- Every Lavatory-C is something that depth 48.0 and that width 49.0.
- Every Lavatory-H is a LateralLavatory.
- Every Lavatory-H is something that depth 45.0 and that width 37.0.
- Every Lavatory-H-PRM is a LateralLavatory.