Roles, Concepts and their Interpretation

\( A \): Set of atomic concepts (variables: \( A, B \))
\( S \): Set of atomic roles (variable: \( S \))

Interpretation: \( \mathcal{I} = \langle D, [\cdot]^{\mathcal{I}} \rangle \) with \( D \) arbitrary set and Function:
\[
\begin{align*}
[\cdot]^{\mathcal{I}} : A & \rightarrow 2^{D} \\
S & \rightarrow [D]^{2}
\end{align*}
\]
\( [A]^{\mathcal{I}} \) is a concept extension
\( [S]^{\mathcal{I}}(d) \) is a role extension
\( [S]^{(d)}(d) \) is a set of role fillers of role \( S \) for \( d \)

\( R \): Set of role descriptions
\[
\begin{align*}
R & \rightarrow S \quad \text{atomic role} \\
R \cap R' & \rightarrow \text{role conjunction} \\
R_{c} & \rightarrow \text{role restriction}
\end{align*}
\]

The interpretation is defined inductively:
\[
\begin{align*}
[R \cap R']^{\mathcal{I}} &= [d \rightarrow ([R]^{\mathcal{I}}(d) \cap [R']^{\mathcal{I}}(d))] \\
[R_{c}]^{\mathcal{I}} &= [d \rightarrow ([R]^{\mathcal{I}}(d) \cap [C^{\mathcal{I}}](d))]
\end{align*}
\]

Terminologies, Models, Subsumption

A Terminology (\( T \)) is a set of equations and inequations with the following form:
\[
\begin{align*}
A \doteq C & \quad \text{defined concept} \\
A \equiv C & \quad \text{primitive concept} \\
S \doteq R & \quad \text{defined role} \\
S \equiv R & \quad \text{primitive role}
\end{align*}
\]

An interpretation \( \mathcal{I} \) fulfills such an equation (symbolically \( \models \mathcal{I} \)):
\[
\begin{align*}
\models_{\mathcal{I}} A \doteq C & \quad [A]^{\mathcal{I}} = [C]^{\mathcal{I}} \\
\models_{\mathcal{I}} A \equiv C & \quad [A]^{\mathcal{I}} \subseteq [C]^{\mathcal{I}} \\
\models_{\mathcal{I}} S \doteq R & \quad [S]^{(d)}(d) = [R]^{\mathcal{I}}(d) \text{ for all } d \in D \\
\models_{\mathcal{I}} S \equiv R & \quad [S]^{(d)}(d) \subseteq [R]^{\mathcal{I}}(d) \text{ for all } d \in D
\end{align*}
\]

\( \mathcal{I} \) is a model of \( T \) if and only if \( \mathcal{I} \) fulfills all equations.

Subsumption is defined as follows:
\( C \) is subsumed by \( D \) in \( T \) (\( C \subseteq_{T} D \)) if and only if
\[
[C]^{\mathcal{I}} \subseteq [D]^{\mathcal{I}} \text{ for all models } \mathcal{I} \text{ of } T.
\]
Example for an Initial Concept Taxonomy: NIKL

The Web Ontology Language OWL
(Ontology Web Language)

W3C has set up Web-Ontology (WebOnt) Working Group
- WebOnt developed OWL language
- OWL based on earlier languages OIL and DAML (Darpa Agent Markup Language)
  +OIL (Ontology Inference Layer)
- OWL now a W3C recommendation (i.e., a standard)
  • OIL, DAML+OIL and OWL based on Description Logics

The Web Ontology Language OWL (Ontology Web Language) Merges Three Language Families

Frame Languages
- Modeling Primitives
  - OKBC-Lite, KRL, FRL

Concept Languages/ Terminological Logics
- Formal Semantics
  - Subsumption, Inferences

OWL

Web Languages
- XML and RDF Syntax

Short History of Description Logics

Frame Languages
Modeling Primitives
OKBC-Lite, KRL, FRL

Concept Languages/ Terminological Logics
Formal Semantics
Subsumption, Inferences

The Ontology Web Language (OWL) Merges Three Language Families
Three Layers of Mark-Up Languages in the Semantic Web

<table>
<thead>
<tr>
<th>Content</th>
<th>OWL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structure</td>
<td>XML</td>
</tr>
<tr>
<td>Form</td>
<td>HTML</td>
</tr>
</tbody>
</table>

WWW Document

Content : Structure : Form = 1 : n : m

Many Practical Applications of OWL

- UK National Health Service £6 billion IT project use OWL
- US National Cancer Institute works on a 50.000 concepts thesaurus in OWL with >20 developers since 1999
- IBM, Oracle, HP and others are working on OWL products

Automated Services in the Semantic Web

OWL-S

1. Automatic Composition of Semantic Web Services
2. Automatic Execution Monitoring of Semantic Web Services
3. Automatic Invocation of Semantic Web Services
4. Automatic Retrieval and Selection of Semantic Web Services

OWL Class Constructors and their Description Logic (DL) Syntax

<table>
<thead>
<tr>
<th>Constructor</th>
<th>DL Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>intersectionOf</td>
<td>$C_1 \cap \ldots \cap C_n$</td>
<td>Human (\sqcap) Male</td>
</tr>
<tr>
<td>unionOf</td>
<td>$C_1 \cup \ldots \cup C_n$</td>
<td>Doctor (\sqcup) Lawyer</td>
</tr>
<tr>
<td>complementOf</td>
<td>$\sim C$</td>
<td>\sim Male</td>
</tr>
<tr>
<td>oneOf</td>
<td>${x_1} \cup \ldots \cup {x_n}$</td>
<td>{john} \cup {mary}</td>
</tr>
<tr>
<td>allValuesFrom</td>
<td>$\forall P.C$</td>
<td>\forallhasChild.Doctor</td>
</tr>
<tr>
<td>someValuesFrom</td>
<td>$\exists P.C$</td>
<td>\existshasChild.Lawyer</td>
</tr>
<tr>
<td>maxCardinality</td>
<td>$\leq n P$</td>
<td>$\leq 1\text{hasChild}$</td>
</tr>
<tr>
<td>minCardinality</td>
<td>$\geq n P$</td>
<td>$\geq 2\text{hasChild}$</td>
</tr>
</tbody>
</table>

- XMLS datatypes as well as classes in $\forall P.C$ and $\exists P.C$
  - E.g., $\exists\text{hasAge}.\text{nonNegativeInteger}$
- Arbitrarily complex nesting of constructors
  - E.g., Person $\sqcap \forall\text{hasChild}.\text{Doctor} \sqcup \exists\text{hasChild}.\text{Professor}$
OWL Class/Concept Constructors and First Order Predicate Logic (FOL) Notation

<table>
<thead>
<tr>
<th>Constructor</th>
<th>DL Syntax</th>
<th>Example</th>
<th>FOL Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>intersectionOf</td>
<td>$C_1 \cap \ldots \cap C_n$</td>
<td>Human $\cap$ Male</td>
<td>$C_1(x) \land \ldots \land C_n(x)$</td>
</tr>
<tr>
<td>unionOf</td>
<td>$C_1 \cup \ldots \cup C_n$</td>
<td>Doctor $\cup$ Lawyer</td>
<td>$C_1(x) \lor \ldots \lor C_n(x)$</td>
</tr>
<tr>
<td>complementOf</td>
<td>$\neg C$</td>
<td>$\neg C(x)$</td>
<td>$\neg C_1(x)$</td>
</tr>
<tr>
<td>oneOf</td>
<td>${x_1, \ldots, x_n}$</td>
<td>$x_1 \lor \ldots \lor x_n$</td>
<td>$\forall x. P(x)$</td>
</tr>
<tr>
<td>allValuesFrom</td>
<td>$\forall P.C$</td>
<td>$\forall P.C$</td>
<td>$\exists x. P(x)$</td>
</tr>
<tr>
<td>someValuesFrom</td>
<td>$\exists P.C$</td>
<td>$\exists P.C$</td>
<td>$\exists x. P(x)$</td>
</tr>
<tr>
<td>maxCardinality</td>
<td>$\leq n P$</td>
<td>$\leq n P$</td>
<td>$\leq n P$</td>
</tr>
<tr>
<td>minCardinality</td>
<td>$\geq n P$</td>
<td>$\geq n P$</td>
<td>$\geq n P$</td>
</tr>
</tbody>
</table>

- C is a concept (class); P is a role (property); x is an individual name
- XML Schema datatypes as well as classes in $\forall P.C$ and $\exists P.C$
  - Restricted form of DL concrete domains

An Example of the RDFS Syntax for OWL

```xml
<owl:Class>
  <owl:intersectionOf rdf:parseType="collection">
    <owl:Class rdf:about="#Person"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:toClass>
        <owl:unionOf rdf:parseType="collection">
          <owl:Class rdf:about="#Doctor"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#hasChild"/>
            <owl:hasClass rdf:resource="#Professor"/>
          </owl:Restriction>
        </owl:unionOf>
      </owl:toClass>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Person $\cap \forall$hasChild.Doctor $\cap \exists$hasChild.Professor:

OWL Axioms

<table>
<thead>
<tr>
<th>Axiom</th>
<th>DL Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>subClassOf</td>
<td>$C_1 \sqsubseteq C_2$</td>
<td>Human $\sqsubseteq$ Animal $\sqsubseteq$ Biped</td>
</tr>
<tr>
<td>equivalentClass</td>
<td>$C_1 \equiv C_2$</td>
<td>Man $\equiv$ Human $\equiv$ Male</td>
</tr>
<tr>
<td>disjoint</td>
<td>$C_1 \sqcap C_2$</td>
<td>Male $\sqcap$ Female</td>
</tr>
<tr>
<td>sameIndividualAs</td>
<td>${x_1} \equiv {x_2}$</td>
<td>${\text{President.Bush}} \equiv {\text{G.W. Bush}}$</td>
</tr>
<tr>
<td>differentFrom</td>
<td>${x_1} \not\equiv {x_2}$</td>
<td>${\text{john}} \not\equiv {\text{peter}}$</td>
</tr>
<tr>
<td>subPropertyOf</td>
<td>$P_1 \sqsubseteq P_2$</td>
<td>hasDaughter $\sqsubseteq$ hasChild</td>
</tr>
<tr>
<td>equivalentProperty</td>
<td>$P_1 \equiv P_2$</td>
<td>hasDaughter $\equiv$ hasChild</td>
</tr>
<tr>
<td>inverseOf</td>
<td>$P_1 \sqsubseteq P_2^\top$</td>
<td>hasChild $\sqsubseteq$ hasParent $\top$</td>
</tr>
<tr>
<td>transitiveProperty</td>
<td>$P_1 \sqsubseteq P_2$</td>
<td>ancestor $\sqsubseteq$ ancestor</td>
</tr>
<tr>
<td>functionalProperty</td>
<td>$\top \sqsubseteq P$</td>
<td>$\top \sqsubseteq 1P$</td>
</tr>
<tr>
<td>inverseFunctionalProperty</td>
<td>$\top \sqsubseteq P$</td>
<td>$\top \sqsubseteq 1P$</td>
</tr>
</tbody>
</table>

Tools and Infrastructure for OWL

- Editors/Browsers/Testing environments
  - Oiled, Protégé, Swoop, Construct, Ontotrack, …
Protégé OWL Plugin

- Extension of Protégé for handling OWL ontologies
- Features
  - Loading and saving OWL files & databases
  - Graphical editors for class expressions
  - Access to description logics reasoners
  - Powerful platform for hooking in custom-tailored components

OWL Plugin Architecture

- OWL Extension APIs (OWL-S, etc.)
- Protégé OWL API (Logical class definitions, restrictions, etc.)
- Protégé API (Classes, properties, individuals, etc.)
- Protégé GUI (Expression Editor, Conditions Widget, etc.)
- Protégé OWL GUI (Protégé OWL API, Protégé GUI, Jena API, Protégé Core System)
- Jena API (Parsing, Reasoning)

Visualization with OWLVis
Tools and Infrastructure for OWL

- **Editors/environments**
  - Oiled, Protégé, Swoop, Construct, Ontotrack, …
- **Reasoning systems**
  - Cerebra, FaCT++, Kaon2, Pellet, Racer, …

Reasoning Engines for Description Logics