Vorlesung

Dr. Harald Sack

Hasso-Plattner-Institut für Softwaresystemtechnik

Universität Potsdam

Wintersemester 2008/09

Blog zur Vorlesung: http://sw0809.blogspot.com/

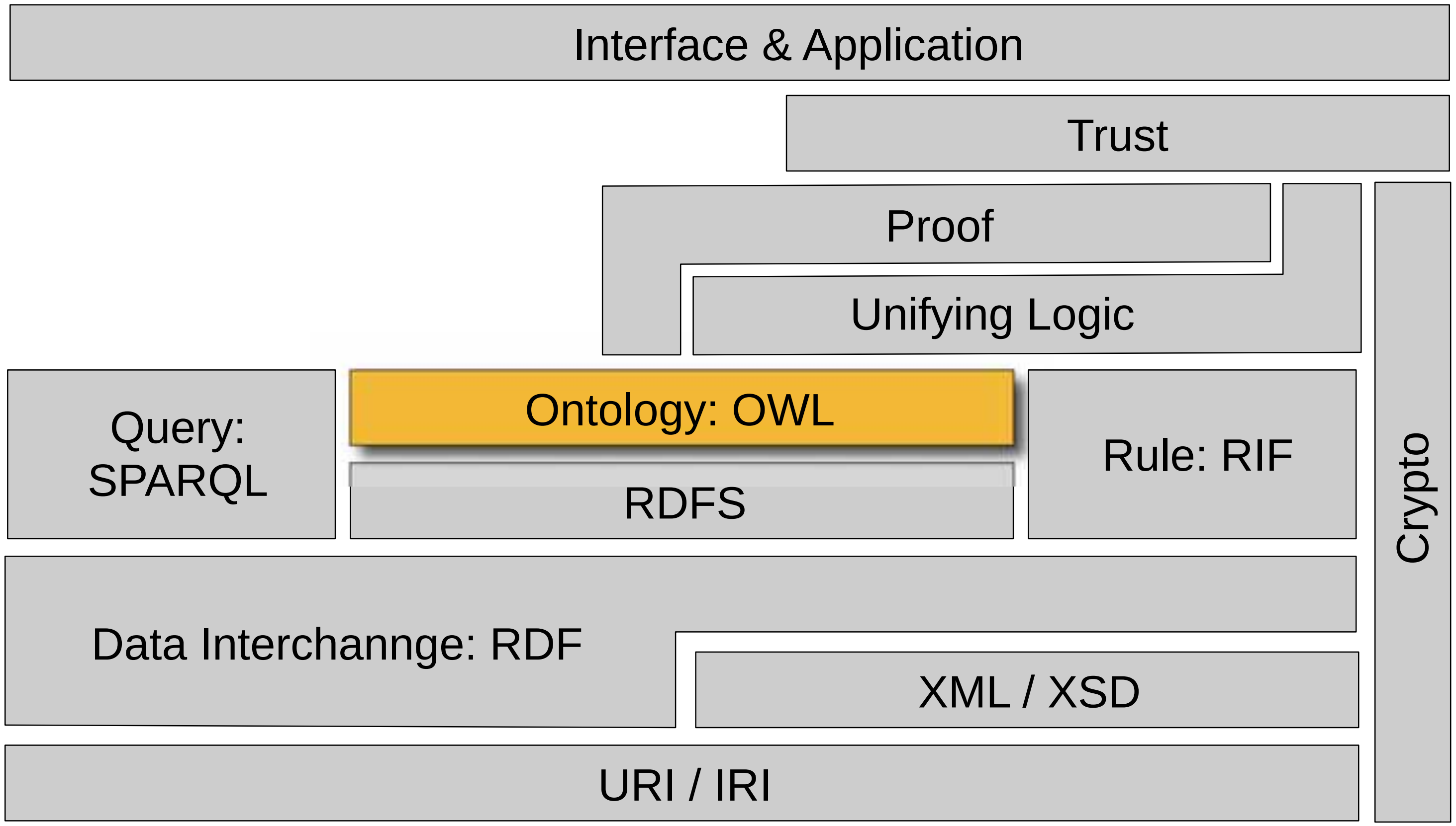
Semantic Web - Vorlesungsinhalt

1. Einführung
2. Die Sprachen des Semantic Web
3. **Wissensrepräsentation**
4. Web of Trust
5. Ontology Engineering
6. Semantic Web Anwendungen

1 2 3 4 5 6 7 8 9 **22.01.2009 – Vorlesung Nr. 10** 11 12 13

3. Wissensrepräsentationen

Semantic Web Architektur



Um mit OWL Schlussfolgerungen zu ziehen, brauchen wir eine modelltheoretische Semantik für OWL

3.7 Web Ontology Language (OWL)

Warum nicht FOL als Semantik Web Sprache?

- Warum nicht einfach FOL für Ontologien nehmen?
  - FOL kann alles
  - ..... Assembler auch
- FOL ist
  - sehr ausdrucksstark
  - deshalb unhandlich bei der Modellierung
  - schlecht geeignet um Konsens bei der Modellierung zu finden
  - Beweistheoretisch sehr komplex
  - FOL ist keine Markupsprache

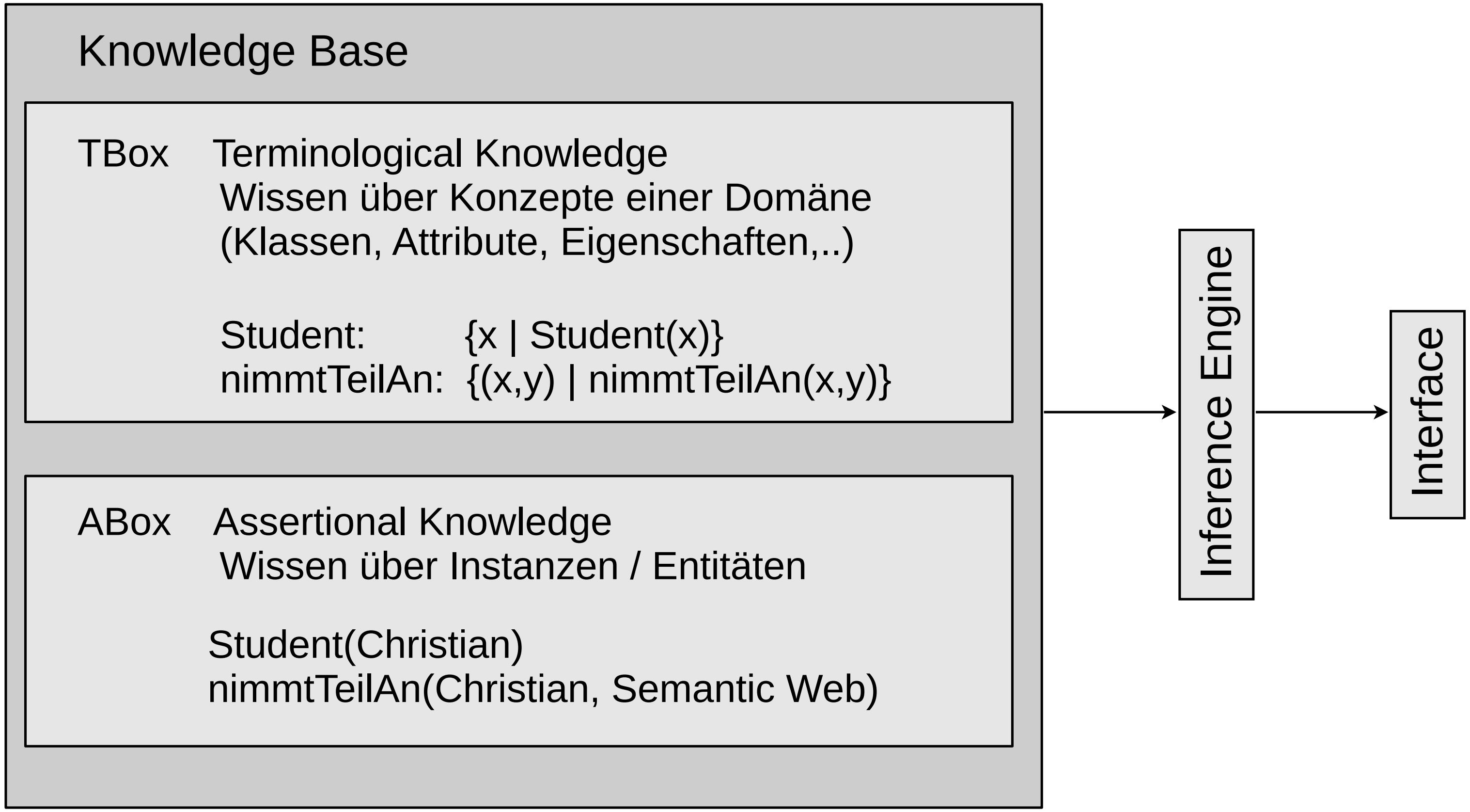
➡ Suche ein geeignetes Fragment von FOL

Beschreibungslogiken

- engl.: description logics (DLs)
- Fragmente von FOL
- entwickelt aus „*semantischen Netzwerken*“
- meist entscheidbar
- vergleichsweise ausdrucksstark
- enge Verwandtschaft mit Modallogiken

- W3C Standard OWL DL basiert auf der Beschreibungslogik
  $\mathcal{SHOIN}(D)$
- wir besprechen zunächst $\mathcal{ALC}$ (Basis für komplexere DLs)

Allgemeine DL Architektur



Allgemeiner DL Aufbau

- DLs sind eine Familie logikbasierter Formalismen zur Wissensrepräsentation
- Spezielle Sprachen u.a. charakterisiert durch:
  - Konstruktoren für komplexe Konzepte und Rollen aus einfacheren
  - Menge von Axiomen um Fakten über Konzepte, Rollen und Individuen auszudrücken
  - $\mathcal{ALC}$ (Attribute Language with Complement) ist die kleinste DL, die aussagenlogisch abgeschlossen ist
  - Konjunktion, Disjunktion, Negation sind Klassenkonstruktoren, geschrieben $\sqcap$, $\sqcup$, $\neg$
  - Quantoren schränken Rollenbereiche ein:

$$\text{Man} \sqcap \exists \text{hasChild.Female} \sqcap \exists \text{hasChild.Male} \sqcap \forall \text{hasChild.(Rich} \sqcup \text{Happy)}$$

Weitere DL Sprachmittel

- Andere Konstruktoren sind z.B.
  - Number restrictions (cardinality constraints) für Rollen: ≥ 3 hasChild, ≤ 1 hasMother
  - Qualified number restrictions: ≥ 2 hasChild.Female, ≤ 1 hasParent.Male
  - Nominals (definition by extension): {Italy, France, Spain}
  - Concrete domains (datatypes): hasAge.(≥ 21)

- Inverse roles: hasChild^{-} $\equiv$ hasParent
- Transitive roles: hasAncestor $\sqsubseteq^{+}$ hasAncestor
- Role composition: hasParent.hasBrother(uncle)

$\mathcal{ALC}$ Grundbausteine

- Grundbausteine:
  - Klassen
  - Rollen
  - Individuen

- Student(Christian)
  Individuum Christian ist in Klasse Student

- nimmtTeilAn(Christian, VorlesungSemanticWeb)
  Christian nimmt an der Vorlesung SemanticWeb teil

$\mathcal{ALC}$ Grundbausteine

- Professor $\sqsubseteq$ Fakultaetsmitglied
  - jeder Professor ist ein Fakultätsmitglied
  - entspricht $(\forall x)(Professor(x) \rightarrow Fakultaetsmitglied(x))$
  - entspricht owl:subClassOf
- Professor $\equiv$ Fakultaetsmitglied
  - Die Fakultätsmitglieder sind genau die Professoren
  - entspricht $(\forall x)(Professor(x) \leftrightarrow Fakultaetsmitglied(x))$
  - entspricht owl:equivalentClass

$\mathcal{ALC}$ komplexe Klassenbeziehungen

- Konjunktion $\sqcap$
- Disjunktion $\sqcup$
- Negation $\neg$

$$\text{Professor} \sqsubseteq (\text{Person} \sqcap \text{Unversitaetsangehoeriger}) \sqcup (\text{Person} \sqcap \neg\text{Student})$$

$$(\forall x)(\text{Professor}(x) \rightarrow ((\text{Person}(x) \land \text{Universitaetsangehoeriger}(x)) \lor (\text{Person}(x) \land \neg\text{Student}(x)))$$

$\mathcal{ALC}$ Quantoren auf Rollen

- Pruefung ⊑ ∀hatPruefer.Professor
- (∀x)(Pruefung(x) → (∀y)(hatPruefer(x,y) → Professor(y)))
- entspricht owl:allValuesFrom

- Pruefung ⊑ ∃hatPruefer.Person
- (∀x)(Pruefung(x) → (∃y)(hatPruefer(x,y) ∧ Person(y)))
- entspricht owl:someValuesFrom

$\mathcal{ALC}$ Modellierung

- owl:nothing: $\bot \equiv C \sqcap \neg C$
- owl:thing: $\top \equiv C \sqcup \neg C$
- owl:disjointWith: $C \sqcap D \equiv \bot$

  oder gleichbedeutend: $C \sqsubseteq \neg D$
- rdfs:range: $\top \sqsubseteq \forall R.C$
- rdfs:domain: $\exists R.\top \sqsubseteq C$

ALC - formale Syntax

- Folgende Syntaxregeln erzeugen Klassen in *ALC,* dabei ist A eine atomare Klasse, C und D **komplexe Klassen** und R eine Rolle:

  - C,D ::= A | $\top$ | $\bot$ | $\neg$C | C $\sqcap$ D | C $\sqcup$ D | $\exists$R.C | $\forall$R.C

- Eine *ALC* TBox besteht aus Aussagen der Form C $\sqsubseteq$ D und C $\equiv$ D, wobei C, D komplexe Klassen sind.
- Eine *ALC* ABox besteht aus Aussagen der Form C(a) und R(a,b), wobei C eine komplexe Klasse, R eine Rolle und a,b Individuen sind.
- Eine *ALC*-Wissensbasis besteht aus einer ABox und einer TBox.

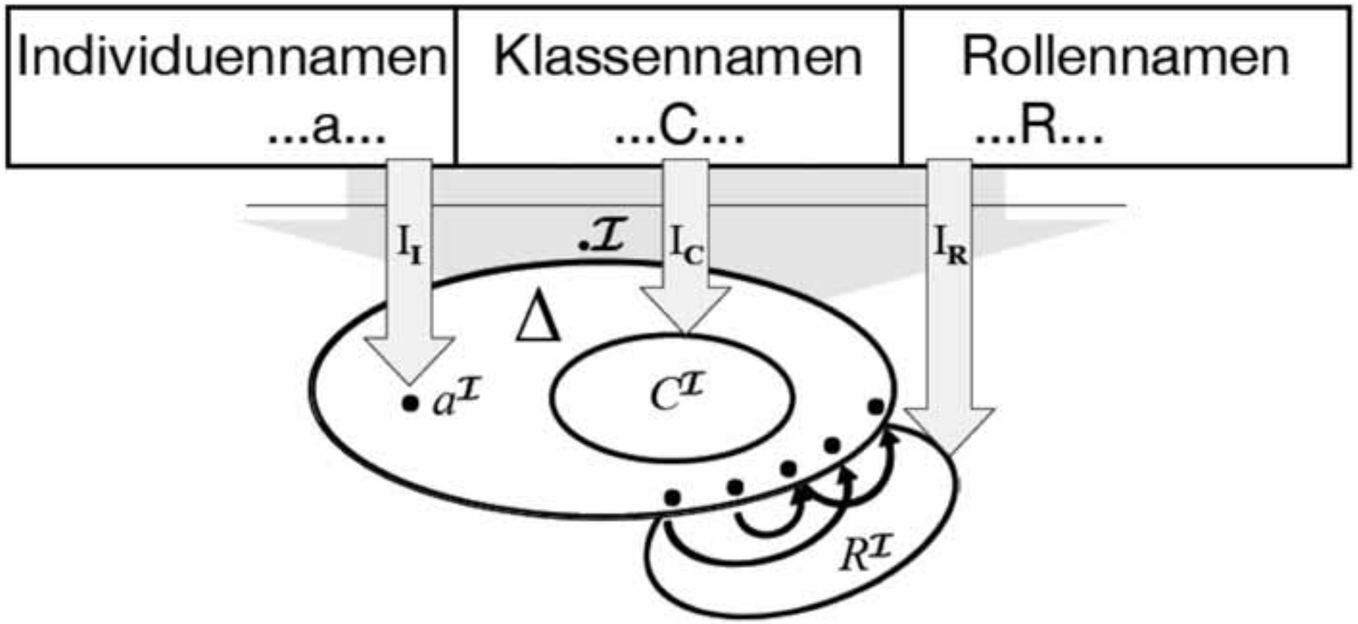
$\mathcal{ALC}$ - Semantik (Interpretationen)

- wir definieren eine modelltheoretische Semantik für $\mathcal{ALC}$ (d.h. Folgerung wird über Interpretationen definiert)

- eine Interpretation $I=(\Delta^I,\cdot^I)$ besteht aus
  - einer Menge Δ^I, genannt Domäne und
  - einer Funktion $\cdot^I$, die abbildet von
    - Individuennamen a auf Domänenelemente $a^I \in \Delta^I$
    - Klassennamen C auf Mengen von Domänenelementen $C^I \subseteq \Delta^I$
    - Rollennamen R auf Mengen von Paaren von Domänenelementen $R^I \subseteq \Delta^I \times \Delta^I$

$\mathcal{ALC}$ - Semantik (Interpretationen)

- schematisch:



$\mathcal{ALC}$ - Semantik

- wird auf komplexe Klassen erweitert:
  - $\top^I = \Delta^I$
  - $(C \sqcup D)^I = C^I \cap D^I$
  - $(\neg C)^I = \Delta^I \setminus C^I$
  - $\forall R.C = \{ x \mid \forall(x,y) \in R^I \rightarrow y \in C^I\}$
  - $\exists R.C = \{ x \mid \exists(x,y) \in R^I \text{ mit } y \in C^I\}$
  - $\bot^I = \varnothing$
  - $(C \sqcap D)^I = C^I \cup D^I$
- und auf Axiome:
  - $C(a)$ gilt, wenn $a^I \in C^I$
  - $R(a,b)$ gilt, wenn $(a^I,b^I) \in R^I$
  - $C \sqsubseteq D$ gilt, wenn $C^I \subseteq D^I$
  - $C \equiv D$ gilt, wenn $C^I = D^I$

$\mathcal{ALC}$ - alternative Semantik

- Übersetzung in die Prädikatenlogik mittels der Abbildung π
- ABox:
  π(C(a))=C(a)
  π(R(a,b))=R(a,b)
- TBox:
  rekursive Definition →
- Dabei sind C,D komplexe Klassen, R eine Rolle und A eine atomare Klasse.

$$
\begin{aligned}
\pi(C \sqsubseteq D) &= (\forall x)(\pi_x(C) \rightarrow \pi_x(D)) \\
\pi(C \equiv D) &= (\forall x)(\pi_x(C) \leftrightarrow \pi_x(D)) \\
\pi_x(A) &= A(x) \\
\pi_x(\neg C) &= \neg \pi_x(C) \\
\pi_x(C \sqcap D) &= \pi_x(C) \wedge \pi_x(D) \\
\pi_x(C \sqcup D) &= \pi_x(C) \vee \pi_x(D) \\
\pi_x(\forall R.C) &= (\forall y)(R(x,y) \rightarrow \pi_y(C)) \\
\pi_x(\exists R.C) &= (\exists y)(R(x,y) \wedge \pi_y(C)) \\
\pi_y(A) &= \boldsymbol{A(y)} \\
\pi_y(\neg C) &= \neg \pi_y(C) \\
\pi_y(C \sqcap D) &= \pi_y(C) \wedge \pi_y(D) \\
\pi_y(C \sqcup D) &= \pi_y(C) \vee \pi_y(D) \\
\pi_y(\forall R.C) &= (\forall x)(R(y,x) \rightarrow \pi_x(C)) \\
\pi_y(\exists R.C) &= (\exists x)(R(y,x) \wedge \pi_x(C))
\end{aligned}
$$

$\mathcal{ALC}$ - Wissensbasis

- Terminologisches Wissen (TBox)
  Axiome, die die Struktur der zu modellierenden Domäne beschreiben (konzeptionelles Schema):
  - Human $\sqsubseteq$ $\exists$parentOf.Human
  - Orphan $\equiv$ Human $\sqcap$ $\neg\exists$hasParent.Alive
- Assertionales Wissen (ABox)
  Axiome, die konkrete Situationen (Daten) beschreiben:
  - Orphan(harrypotter)
  - hasParent(harrypotter,jamespotter)
- Semantik und logische Konsequenzen klar, da übersetzbar nach FOL

Beschreibungslogiken

<table>
<tr><th>Operator/Construktor</th><th>Syntax</th><th colspan="2">Sprache</th></tr>
<tr><td>Konjunktion</td><td>$A \sqcap B$</td><td rowspan="3">FL</td><td rowspan="10">S*</td></tr>
<tr><td>Wertrestriktion</td><td>$\forall R.C$</td></tr>
<tr><td>Existenzquantor</td><td>$\exists R$</td></tr>
<tr><td>Top</td><td>$\top$</td><td rowspan="7">AL*</td></tr>
<tr><td>Bottom</td><td>$\bot$</td></tr>
<tr><td>Negation</td><td>$\neg A$</td></tr>
<tr><td>Disjunktion</td><td>$A \sqcup B$</td></tr>
<tr><td>Existentielle Restriktion</td><td>$\exists R.C$</td></tr>
<tr><td>Zahlenrestriktion</td><td>$(\leq nR)$ $(\geq nR)$</td></tr>
<tr><td>Menge von Individuen</td><td>$\{a_1,\ldots,a_2\}$</td></tr>
<tr><td>Beziehungshierarchie</td><td>$R \sqsubseteq S$</td><td colspan="2">H</td></tr>
<tr><td>inverse Beziehung</td><td>R^{-1}</td><td colspan="2">I</td></tr>
<tr><td>Qualifizierte Zahlenrestriktion</td><td>$(\leq nR.C)$ $(\geq nR.C)$</td><td colspan="2">Q</td></tr>
</table>

Beschreibungslogiken

- $\mathcal{ALC}$: Attribute Language with Complement
- $\mathcal{S}$: $\mathcal{ALC}$ + Rollentransitivität
- $\mathcal{H}$: Subrollenbeziehung
- $\mathcal{O}$: abgeschlossene Klassen
- $\mathcal{I}$: inverse Rollen
- $\mathcal{N}$: Zahlenrestriktionen ≤n R etc.
- $\mathcal{Q}$: Qualifizierende Zahlenrestriktionen ≤n R.C etc.
- $(\mathcal{D})$: Datentypen
- $\mathcal{F}$: Funktionale Rollen

- OWL DL ist $\mathcal{SHOIN}(\mathcal{D})$
- OWL Lite ist $\mathcal{SHIF}(\mathcal{D})$

OWL und $\mathcal{ALC}$

- Folgende OWL DL Sprachelemente sind in $\mathcal{ALC}$ repräsentierbar:
  - Klassen, Rollen, Individuen
  - Klassenzugehörigkeit, Rolleninstanzen
  - owl:Thing und owl:Nothing
  - Klasseninklusion, -äquivalenz, -disjunktheit
  - owl:intersectionOf, owl:unionOf
  - owl:complementOf
  - owl:allValuesFrom, owl:someValuesFrom
  - rdfs:range und rdfs:domain

OWL als $\mathcal{SHOIN}(\mathcal{D})$

- owl:sameAs
  - gibt an dass zwei Individuennamen dasselbe Element bezeichnen
  - DL: $a=b$
  - FOL: Erweiterung durch Gleichheitsprädikat
- owl:differentFrom
  - gibt an dass zwei Individuennamen unterschiedliche Elemente bezeichnen
  - DL: $a\neq b$
  - FOL: $\neg(a=b)$

OWL als *SHOIN(D)*

- Abgeschlossene Klassen
  - owl:oneOf
    - definiert eine Klasse durch vollständige Aufzählung ihrer Instanzen
    - DL: $C \equiv \{a,b,c\}$
    - FOL: $(\forall x)\ (C(x) \leftrightarrow (x=a \lor x=b \lor x=c))$
  - owl:hasValue
    - „erzwingt“ Rolle zu einem bestimmten Individuum
    - darstellbar mittels owl:someValuesFrom und owl:oneOf

OWL als *SHOIN(D)*

- Zahlenrestriktionen mittels Gleichheitsprädikat

```
<owl:Class rdf:about="Pruefung">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="hatPruefer"/>
      <owl:maxCardinality>2</owl:maxcardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

- Eine Prüfung kann höchstens zwei Prüfer haben.
- DL: Pruefung $\sqsubseteq$ ≤ 2 hatPruefer
- FOL: (P... Prüfung, h...hatPruefer)
  - $(\forall x)(P(x) \rightarrow \neg(\exists x1)(\exists x2)(\exists x3)$
    $(x1 \neq x2 \land x2 \neq x3 \land x1 \neq x3 \land h(x,x1) \land h(x,x2) \land h(x,x3)))$

OWL als *SHOIN(D)*

- Rollenkonstruktoren
  - rdfs:subPropertyOf
    - DL: $R \sqsubseteq S$
    - FOL: $(\forall x)\forall(y)(R(x,y) \rightarrow S(x,y))$
    - Entsprechend Rollenäquivalenz
  - Inverse Rollen: $R \equiv S^-$
    - FOL: $(\forall x)(\forall y)(R(x,y) \leftrightarrow S(y,x))$
  - Transitive Rollen: $R \sqsubseteq_+ R$
    - FOL: $(\forall x)(\forall y)(\forall z)(R(x,y) \wedge R(y,z) \rightarrow R(x,z))$
  - Symmetrie: $R \equiv R-$
  - Funktionalität: $\top \sqsubseteq \leq 1\ R$
  - Inverse Funktionalität: $\top \sqsubseteq \leq 1\ R^-$

OWL als *SHOIN(D)*

- Für OWL DL ist erlaubt:
  - *ALC*
  - Gleichheit und Ungleichheit zwischen Individuen
  - Abgeschlossene Klassen
  - Zahlenrestriktionen
  - Subrollen und Rollenäquivalenz
  - Inverse und transitive Rollen
  - Datentypen
    - Erlaubt ist die Verwendung von Datentypen im zweiten Argument konkreter Rollen in der ABox.
    - Eine Menge konkreter Daten kanneine abgeschlossene Klasse bilden.
    - Datentypen lassen sich nicht ohne Weiteres in FOL ausdrücken. Man kann die FOL Semantik aber entsprechend erweitern.

OWL DL Syntax - Übersicht

	Concepts	
ALC	Atomic	A, B
ALC	Not	$\neg C$
ALC	And	$C \sqcap D$
ALC	Or	$C \sqcup D$
ALC	Exists	$\exists R.C$
ALC	For all	$\forall R.C$
Q(N)	At least	$\geq n\ R.C\ (\geq n\ R)$
Q(N)	At most	$\leq n\ R.C\ (\leq n\ R)$
O	Nominal	$\{i_1,\ldots i_n\}$

	Roles	
	Atomic	Roles
I	Inverse	R^-

Concept Axioms (TBox)	
Subclass	$C \sqsubseteq D$
Equivalent	$C \equiv D$

	Role Axioms (TBox)	
H	Subrole	$R \sqsubseteq S$
S	Transitivity	$R+$

Assertional Axioms (ABox)	
Instance	$C(a)$
Role	$R(a,b)$
Same	$a=b$
Different	$a\neq b$

OWL DL Syntax - Klassenkonstruktoren

Konstruktor	DL Syntax	Beispiel	FOL Syntax
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	Person $\sqcap$ Male	$C_1(x) \wedge \ldots \wedge C_n(x)$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Student $\sqcup$ Professor	$C_1(x) \vee \ldots \vee C_n(x)$
complementOf	$\neg C$	$\neg$ Male	$\neg C(x)$
oneOf	$\{x_1\} \sqcup \ldots \sqcup \{x_n\}$	{john} $\sqcup$ {mary}	$x=x_1 \vee \ldots \vee x=x_n$
allValuesFrom	$\forall P.C$	nimmtTeil.Seminar	$\forall y, P(x,y) \rightarrow C(y)$
someValuesFrom	$\exists P.C$	nimmtTeil.Seminar	$\exists y, P(x,y) \rightarrow C(y)$
maxCardinality	$\leq nP$	≤ 2 nimmtTeil	$\exists^{\leq n} y, P(x,y)$
minCardinality	$\geq nP$	≥ 1 nimmtTeil	$\exists^{\geq n} y, P(x,y)$

OWL DL Syntax - Axiome

Axiome	DL Syntax	Beispiel
subClassOf	$C_1 \sqsubseteq C_2$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
equivalentClass	$C_1 \equiv C_2$	Woman $\equiv$ Human $\sqcap$ Female
disjointWith	$C_1 \sqsubseteq \neg C_2$	Male $\sqsubseteq \neg$ Female
sameAs	$\{x_1\} \equiv \{x_2\}$	{JamesBond} $\equiv$ {007}
differentFrom	$\{x_1\} \sqsubseteq \neg\{x_2\}$	{John} $\sqsubseteq \neg${Mary}
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter $\sqsubseteq$ hasChild
equivalentProperty	$P_1 \equiv P_2$	cost $\equiv$ price
inverseOf	$P_1 \equiv P_2^-$	hasChild $\equiv$ hasParent$^-$
transitiveProperty	$P_1^+ \sqsubseteq P_2$	hasAncester$^+$ $\sqsubseteq$ hasAncester
functionalProperty	$\top \sqsubseteq \leq 1P$	$\top \sqsubseteq \leq$1hasMother
inverseFunctionalProperty	$\top \sqsubseteq \leq 1P^-$	$\top \sqsubseteq \leq$1isMotherOf$^-$

OWL DL Syntax

- Beliebig komplexes Schachteln von Konstruktoren erlaubt:
  - Person $\sqcap$ $\forall$hasChild.(Doctor $\sqcup$ $\exists$hasChild.Doctor)
- General Class Inclusion ($\sqsubseteq$) genügt:
  - $C \equiv D$ gdw. ($C \sqsubseteq D$ und $D \sqsubseteq C$)
- Offensichtliche FOL-Äquivalenzen
  - $C \equiv D \longleftrightarrow (\forall x)\ (C(x) \leftrightarrow D(x))$
  - $C \sqsubseteq D \longleftrightarrow (\forall x)\ (C(x) \rightarrow D(x))$

OWL DL und OWL RDF Syntax

- Person ⊓ ∀hasChild.(Doctor ⊔ ∃hasChild.Doctor)

```
<owl:Class>
  <owl:intersectionOf rdf:parseType="collection">
    <owl:Class rdf:about="Person"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="hasChild"/>
      <owl:allValuesFrom>
        <owl:unionOf rdf:parseType="collection">
          <owl:Class rdf:about="Doctor"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="hasChild"/>
            <owl:someValuesFrom rdf:resource="Doctor"/>
          </owl:Restriction>
        </owl:unionOf>
      </owl:allValuesFrom>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Open World vs Closed World Assumption

- OWA: Open World Assumption
  Die Existenz von weiteren Individuen ist möglich, sofern sie nicht explizit ausgeschlossen wird.
- CWA: Closed World Assumption
  Es wird angenommen, dass die Wissensbasis alle Individuen enthält.

	are all children of Bill male?	*no idea since we do not know all children of Bill*	*if we assume that we know everything about Bill then all of his children are male*
child(Bill,Bob) Man(Bob)	$? \models \forall child.Man(Bill)$	DL answers don't know	PROLOG answers yes
$\leq$ 1 child.$\top$(Bill)	$? \models \forall child.Man(Bill)$	yes	*now we know everything about Bill's children*

Wichtige Inferenzprobleme

- Globale Konsistenz der Wissensbasis — KB $\models$ false?
  - ist Wissensbasis sinnvoll?
- Klassenkonsistenz — $C \equiv \bot$?
  - Muss Klasse C leer sein?
- Klasseninklusion (Subsumption) — $C \sqsubseteq D$?
  - Strukturierung der Wissensbasis
- Klassenäquivalenz — $C \equiv D$?
  - Sind zwei Klassen eigentlich dieselbe?
- Klassendisjunktheit — $C \sqcap D = \bot$?
  - Sind zwei Klassen disjunkt?
- Klassenzugehörigkeit — C(a)?
  - Ist Individuum a in der Klasse C?
- Instanzgenerierung (Retrieval) „alle x mit C(x) finden“
- Finde alle (bekannten!) Individuen zur Klasse C.

Entscheidbarkeit und OWL DL

- Entscheidbarkeit:
  zu jedem Inferenzproblem gibt es einen immer terminierenden Algorithmus
  - OWL DL ist Fragment von FOL, also könnten (im Prinzip) FOL-Inferenzalgorithmen (Resolution, Tableaux) verwendet werden.
  - Diese terminieren aber nicht immer!

- Problem: Finde immer terminierende Algorithmen!
  - Keine „naiven“ Lösungen in Sicht!

Entscheidbarkeit und OWL DL

- Tableaux- und Resolutionsverfahren müssen für OWL DL abgewandelt werden
- Wir werden uns (zuerst) auf $\mathcal{ALC}$ beschränken
- Tableaux- und Resolutionsverfahren zeigen Unerfüllbarkeit einer Theorie
- Rückführung der Inferenzprobleme auf das Finden von Inkonsistenten in der Wissensbasis, d.h. zeigen der Unerfüllbarkeit der Wissensbasis!

Rückführung der Inferenz auf Erfüllbarkeit / Konsistenz

- Klassenkonsistenz $C \equiv \bot$ gdw
  - KB $\sqcup$ {C(a)} unerfüllbar (a neu)
- Klasseninklusion (Subsumption) $C \sqsubseteq D$ gdw
  - KB $\sqcup$ {C $\sqcap$ ¬D(a)} unerfüllbar (a neu)
- Klassenäquivalenz $C \equiv D$ gdw
  - $C \sqsubseteq D$ und $D \sqsubseteq C$
- Klassendisjunktheit $C \sqcap D = \bot$ gdw
  - KB $\sqcup$ {(C $\sqcap$ D)(a)} unerfüllbar (a neu)
- Klassenzugehörigkeit C(a) gdw
  - KB $\sqcup$ {¬C(a)} unerfüllbar (a neu)
- Instanzgenerierung (Retrieval) alle C(X) finden
- Prüfe Klassenzugehörigkeit für alle Individuen.
- Effiziente Implementation problematisch....

3.7 Web Ontology Language (OWL)

3.7.1 Motivation

3.7.2 OWL - Übersicht

3.7.3 OWL Syntax

3.7.4

3.7.5

Einschub:
Tableauverfahren (Tableaux-Verfahren)

Tableau - Verfahren für OWL DL

- **Tableaux Verfahren in PL und FOL**
- Wissensbasis in Negationsnormalform (NNF)
- Tableaux-Erweiterungsregeln für OWL DL
- Tableaux Terminierungsproblem

Tableaux Verfahren in der Aussagenlogik (PL)

- syntaktisches Verfahren zum Prüfen der Konsistenz logischer Ausdrücke
- **Grundidee** (ähnlich Resolution):
    - Beweisverfahren, mit dem eine Formel dadurch bewiesen wird, dass ihre Negation als widersprüchlich abgeleitet wird (*proof by refutation*).
    - Tableaux basieren auf einer Darstellung von Formeln in disjunktiver Normalform (*Resolution: konjunktive Normalform*)
    - Konstruiere **Baum**, in dem jeder Knoten mit einer Formel markiert ist. Ein Pfad von der Wurzel zu einem Blatt stellt die Konjunktion aller Formeln der Knoten entlang des Pfads dar; eine Verzweigung stellt eine Disjunktion dar.
    - Der Baum wird durch sukzessive Anwendung der **Tableaux-Erweiterungsregeln** aufgebaut.
    - Ein Pfad in einem Tableaux ist **abgeschlossen**, wenn entlang des Pfads sowohl X wie ¬X für eine Formel X auftreten, oder wenn F auftritt (X muss nicht atomar sein.).

Tableaux Verfahren in der Aussagenlogik (PL)

- **Grundidee** (Fortsetzung):
  - Ein Tableaux heißt abgeschlossen, wenn alle seine Pfade abgeschlossen sind.
  - Ein Tableaux-Beweis für eine Formel X ist ein abgeschlossenes Tableaux für ¬X.
  - Die Auswahl der Regeln bei der Erweiterung eines Tableaus ist nichtdeterministisch.
  - Für aussagenlogische Tableaux kann die Auswahl etwas eingeschränkt werden

Tableaux Erweiterungsregeln

- für Aussagenlogik:

$$\frac{\neg\neg X}{X} \qquad \frac{\neg W}{F} \qquad \frac{\neg F}{W}$$

- für konjunktive Formeln (α-Regeln):

$$\frac{\alpha}{\begin{array}{c}\alpha_1\\ \alpha_2\end{array}} \qquad \frac{X \wedge Y}{\begin{array}{c}X\\ Y\end{array}} \qquad \frac{\neg(X \vee Y)}{\begin{array}{c}\neg X\\ \neg Y\end{array}} \qquad \frac{\neg(X \Rightarrow Y)}{\begin{array}{c}X\\ \neg Y\end{array}}$$

- für disjunktive Formeln (β-Regeln):

$$\frac{\beta}{\beta_1 \mid \beta_2} \qquad \frac{X \vee Y}{X \mid Y} \qquad \frac{\neg(X \wedge Y)}{\neg X \mid \neg Y} \qquad \frac{(X \Rightarrow Y)}{\neg X \mid Y}$$

Beispiel(1):

zu zeigen: $((q \wedge r) \Rightarrow (\neg q \vee r))$

(1) $\neg((q \wedge r) \Rightarrow (\neg q \vee r))$

(2) α,1: $(q \wedge r)$

(3) α,1: $\neg(\neg q \vee r)$

(4) α,2: q

(5) α,2: r

(6) α,3: $\neg\neg q$

(7) α,3: $\neg r$

Beispiel(2):

zu zeigen : $(p \Rightarrow (q \Rightarrow r)) \Rightarrow ((p \Rightarrow q) \Rightarrow (p \Rightarrow r))$

(1) $\neg((p \Rightarrow (q \Rightarrow r)) \Rightarrow ((p \Rightarrow q) \Rightarrow (p \Rightarrow r)))$

(2|α aus 1) $(p \Rightarrow (q \Rightarrow r))$

(3|α aus 1) $\neg((p \Rightarrow q) \Rightarrow (p \Rightarrow r))$

(4|α aus 3) $(p \Rightarrow q)$

(5|α aus 3) $\neg(p \Rightarrow r)$

(6|α aus 5) p

(7|α aus 5) $\neg r$

(8|β aus 2) $\neg q$ | (9|β aus 2) $(q \Rightarrow r)$

(10|β aus 9) $\neg q$ | (11|β aus 9) r

(12|β aus 4) $\neg p$ | (13|β aus 4) q

Tableaux Erweiterungsregeln für FOL

- wie für Aussagenlogik - in den Regeln stehen X und Y dann für beliebige (prädikatenlogische) Formeln:
- Zusätzlich folgende Regeln für die Behandlung quantifizierter Formeln :

$$\frac{\gamma}{\gamma[t]} \qquad \frac{\delta}{\delta[c]}$$

- für γ universell quantifizierte Formel, δ existenziell quantifizierte Formel, mit:

γ	$\gamma[t]$
$\forall x.\Phi$	$\Phi[x \leftarrow t]$
$\neg\exists x.\Phi$	$\neg\Phi[x \leftarrow t]$

δ	$\delta[c]$
$\exists x.\Phi$	$\Phi[x \leftarrow c]$
$\neg\forall x.\Phi$	$\neg\Phi[x \leftarrow c]$

- t ist Grundterm (d.h. enthält keine in Φ gebundenen Variablen), c ist eine „neue“ Konstante

Beispiel(3):

zu zeigen: $(\forall x.P(x) \vee Q(x)) \Rightarrow (\exists x.P(x)) \vee (\forall x.Q(x))$

(1) $\neg((\forall x.P(x) \vee Q(x)) \Rightarrow (\exists x.P(x)) \vee (\forall x.Q(x)))$

(2|α aus 1) $(\forall x.P(x) \vee Q(x))$

(3|α aus 1) $\neg((\exists x.P(x)) \vee (\forall x.Q(x)))$

(4|α aus 3) $\neg(\exists x.P(x))$

(5|α aus 3) $\neg(\forall x.Q(x))$

(6|δ aus 5) $\neg Q(c)$

(7|γ aus 4) $\neg P(c)$

(8|γ aus 2) $P(c) \vee Q(c)$

(9|β aus 8) $P(c)$ | (10|β aus 8) $Q(c)$

Tableau - Verfahren für OWL DL

- Tableaux Verfahren in PL und FOL
- **Wissensbasis in Negationsnormalform (NNF)**
- Tableaux-Erweiterungsregeln für OWL DL
- Tableaux Terminierungsproblem

Tableaux Transformation in Negationsnormalform

- Gegeben eine Wissensbasis W.
  - Ersetze $C \equiv D$ durch $C \sqsubseteq D$ und $D \sqsubseteq C$
  - Ersetze $C \sqsubseteq D$ durch $\neg C \sqcup D$.
  - Wende die Regeln auf der folgenden Folie an, bis es nicht mehr geht
- Resultierende Wissensbasis: NNF(W)
  - Negationsnormalform von W.
  - Negation steht nur noch direkt vor atomaren Klassen

Tableaux Transformation in Negationsnormalform

- NNF Transformationen

$$\begin{aligned}
&\mathrm{NNF}(C) = C, \text{ falls } C \text{ atomar ist}\\
&\mathrm{NNF}(\neg C) = \neg C, \text{ falls } C \text{ atomar ist}\\
&\mathrm{NNF}(\neg\neg C) = \mathrm{NNF}(C)\\
&\mathrm{NNF}(C \sqcup D) = \mathrm{NNF}(C) \sqcup \mathrm{NNF}(D)\\
&\mathrm{NNF}(C \sqcap D) = \mathrm{NNF}(C) \sqcap \mathrm{NNF}(D)\\
&\mathrm{NNF}(\neg(C \sqcup D)) = \mathrm{NNF}(\neg C) \sqcup \mathrm{NNF}(\neg D)\\
&\mathrm{NNF}(\neg(C \sqcap D)) = \mathrm{NNF}(\neg C) \sqcap \mathrm{NNF}(\neg D)\\
&\mathrm{NNF}(\forall R.C) = \forall\, R.\mathrm{NNF}(C)\\
&\mathrm{NNF}(\exists R.C) = \exists\, R.\mathrm{NNF}(C)\\
&\mathrm{NNF}(\neg\forall R.C) = \exists R.\mathrm{NNF}(\neg C)\\
&\mathrm{NNF}(\neg\exists R.C) = \forall R.\mathrm{NNF}(\neg C)
\end{aligned}$$

- W und NNF(W) sind logisch äquivalent.

Tableaux Transformation in Negationsnormalform

- Beispiel: $P \sqsubseteq (E \sqcap U) \sqcup \neg(\neg E \sqcap D)$
- In NNF: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$.

Tableau - Verfahren für OWL DL

- Tableaux Verfahren in PL und FOL
- Wissensbasis in Negationsnormalform (NNF)
- **Tableaux-Erweiterungsregeln für OWL DL**
- Tableaux Terminierungsproblem

Tableaux Erweiterungsregeln für OWL DL

Auswahl	Aktion
C(a)∈W (ABox)	Füge C(a) hinzu
R(a,b)∈W (ABox)	Füge R(a,b) hinzu
C∈W (TBox)	Füge C(a) für ein bekanntes Individuum a hinzu
(C⊓D)(a)∈A	Füge C(a) und D(a) hinzu
(C⊔D)(a)∈A	Splitte den Zweig. Füge zu (1) C(a) und zu (2) D(a) hinzu
(∃R.C)(a)∈A	Füge (Ra,b) und C(b) für neues Individuum b hinzu
(∀R.C)(a)∈A	Falls R(a,b)∈A, dann füge C(b) hinzu

- Ist das resultierende Tableaux abgeschlossen, so ist die ursprüngliche Wissensbasis unerfüllbar.
- Man wählt dabei immer nur solche Elemente aus, die auch wirklich zu neuen Elementen im Tableaux führen. Ist dies nicht möglich, so terminiert der Algorithmus und die Wissensbasis ist erfüllbar.

Beispiel:

- P ... Professor
- E ... Person
- U ... Universitätsangehöriger
- D ... Doktorand

- Wissensbasis: $P \sqsubseteq (E \sqcap U) \sqcup (E \sqcap \neg D)$
- Ist $P \sqsubseteq E$ logische Konsequenz?

- Wissensbasis (mit Anfrage) in NNF:
  $\{\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D), (P \sqcap \neg E)(a)\}$

Beispiel (Fortsetzung):

- TBox: $\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)$
- Tableaux:

(1) $(P \sqcap \neg E)(a)$ (aus Wissensbasis)

(2|α aus 1) $P(a)$

(3|α aus 1) $\neg E(a)$

(4) $(\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D))(a)$ (δ aus Wissensbasis)

(5) $\neg P(a)$ | (6) $((E \sqcap U) \sqcup (E \sqcap \neg D))(a)$

(7) $(E \sqcap U)(a)$ | (8) $(E \sqcap \neg D)(a)$

(9) $E(a)$ (10) $E(a)$

(11) $U(a)$ (12) $\neg D(a)$

Tableau - Verfahren für OWL DL

- Tableaux Verfahren in PL und FOL
- Wissensbasis in Negationsnormalform (NNF)
- Tableaux-Erweiterungsregeln für OWL DL
- **Tableaux Terminierungsproblem und Blocking**
- Tableaux für OWL DL

3. Wissensrepräsentationen

Einschub: Tableaux-Beweiser / Terminierungsproblem

- Wissensbasis: $\neg$Person $\sqcup$ $\exists$hasParent.Person
- abzuleiten: $\neg$Person(Bill)

Person(Bill)

($\neg$Person $\sqcup$ $\exists$hasParent.Person)(Bill)

$\neg$Person(Bill) | $\exists$hasParent.Person(Bill) ← $\sqcup$

STOP

hasParent(Bill,x_1) ← $\exists$

Person(x_1)

($\neg$Person $\sqcup$ $\exists$hasParent.Person)(x_1)

$\neg$Person(x_1) | $\exists$hasParent.Person(x_1) ← $\sqcup$

STOP

hasParent(x_1,x_2) ← $\exists$

Person(x_2)

($\neg$Person $\sqcup$ $\exists$hasParent.Person)(x_2)

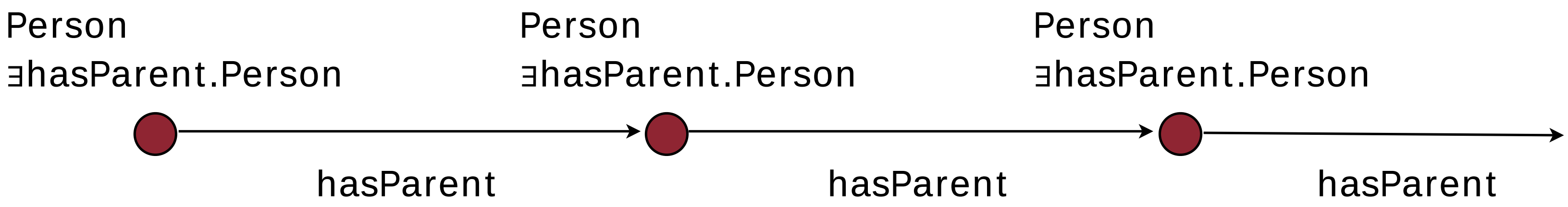
$\neg$Person(x_2) | $\exists$hasParent.Person(x_2) ← $\sqcup$

STOP

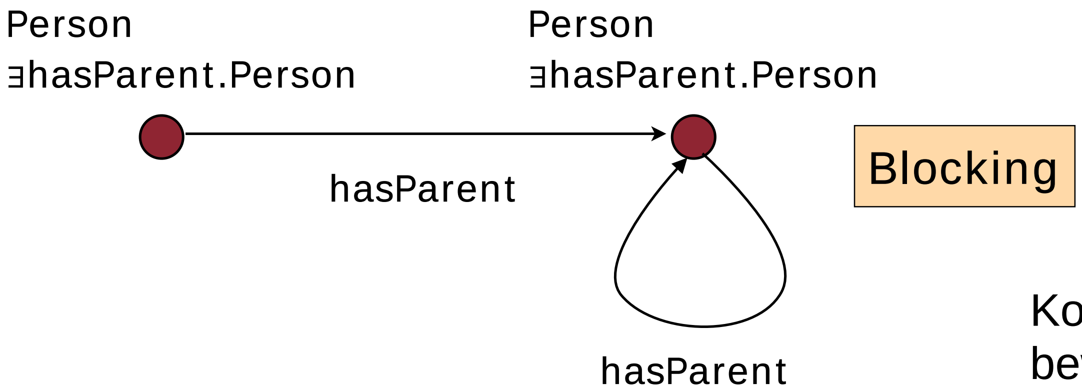
Problem tritt bei Existenzquantoren auf bzw. minCardinality

Idee des Blocking

- wir hatten folgendes konstruiert:



- Idee: Wiederverwendung alter Knoten



Korrektheit muss natürlich bewiesen werden...

3. Wissensrepräsentationen

Einschub: Tableaux-Beweiser / Terminierungsproblem

- Wissensbasis: $\neg \text{Person} \sqcup \exists \text{hasParent.Person}$
- abzuleiten: $\neg \text{Person(Bill)}$

Person(Bill)

$(\neg \text{Person} \sqcup \exists \text{hasParent.Person})(\text{Bill})$

$\neg \text{Person(Bill)}$ STOP

$\exists \text{hasParent.Person(Bill)}$ ← $\sqcup$

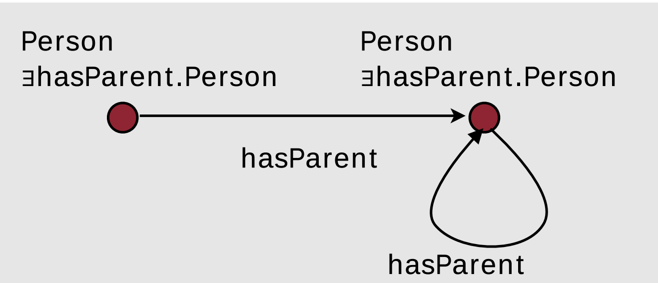
$\text{hasParent}(\text{Bill}, x_1)$ ← $\exists$

$\text{Person}(x_1)$

$(\neg \text{Person} \sqcup \exists \text{hasParent.Person})(x_1)$

$\neg \text{Person}(x_1)$ STOP

$\exists \text{hasParent.Person}(x_1)$ ← $\sqcup$



$\sigma(\text{Bill}) = \{\text{Person}, \neg \text{Person} \sqcup \exists \text{hasParent.Person}, \exists \text{hasParent.Person}\}$

$\sigma(x_1) = \{ \text{Person}, \neg \text{Person} \sqcup \exists \text{hasParent.Person}, \exists \text{hasParent.Person} \}$

$\sigma(x_1) \subseteq \sigma(\text{Bill})$, so Bill blocks x_1 STOP

Tableaux Blocking Definition

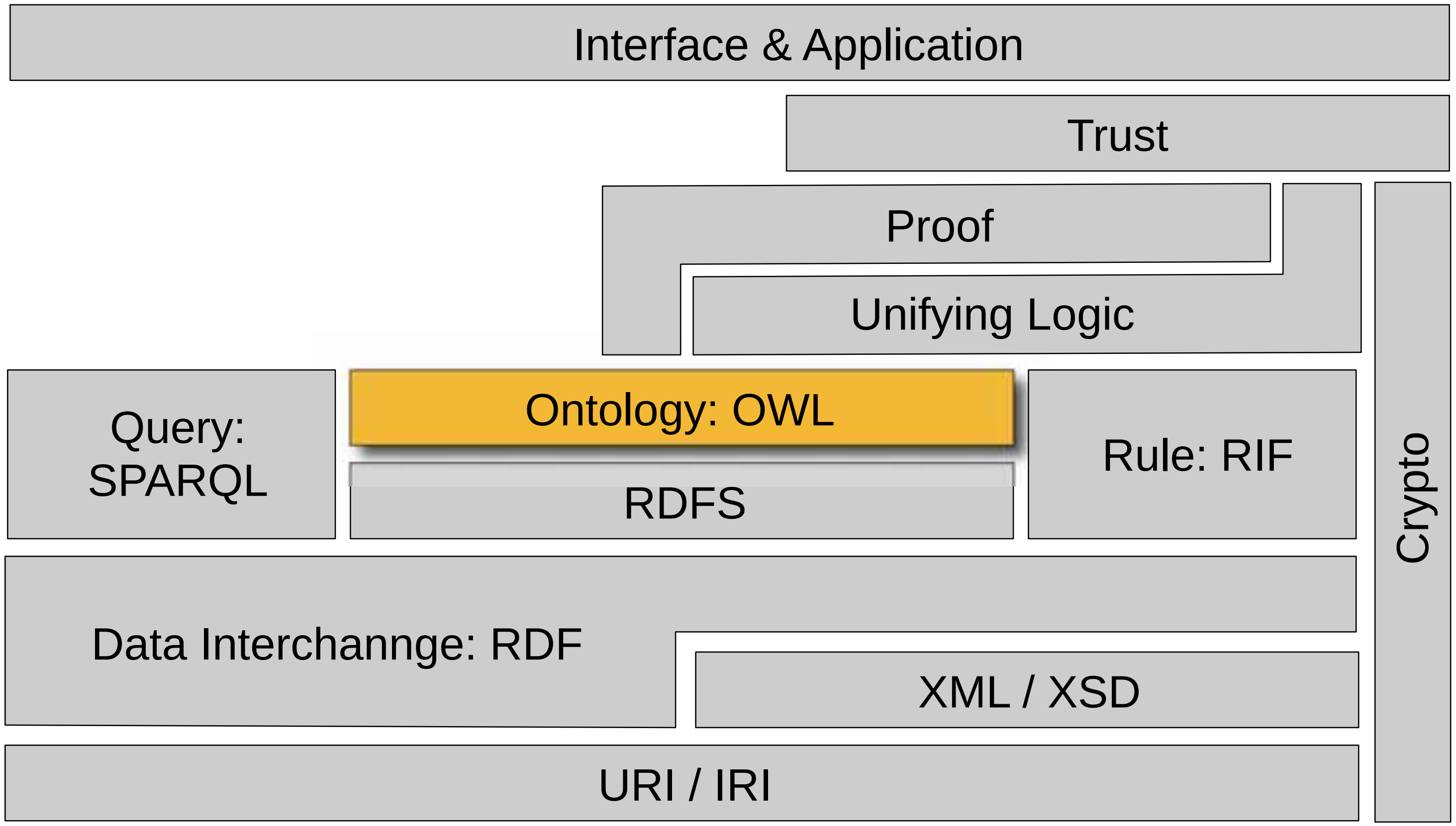
- Die Auswahl von (∃R.C)(a) im Tableauzweig A ist blockiert, falls es ein Individuum b gibt, so dass $\{C \mid C(a) \in A\} \subseteq \{C \mid C(b) \in A\}$ ist.

- Zwei Möglichkeiten der Terminierung:
  1. Abschluss des Tableaus.
     Dann Wissensbasis unerfüllbar.
  2. Keine ungeblockte Auswahl führt zu Erweiterung.
     Dann Wissensbasis erfüllbar.

Tableau - Verfahren für OWL DL

- Tableaux Verfahren in PL und FOL
- Wissensbasis in Negationsnormalform (NNF)
- Tableaux-Erweiterungsregeln für OWL DL
- Tableaux Terminierungsproblem und Blocking

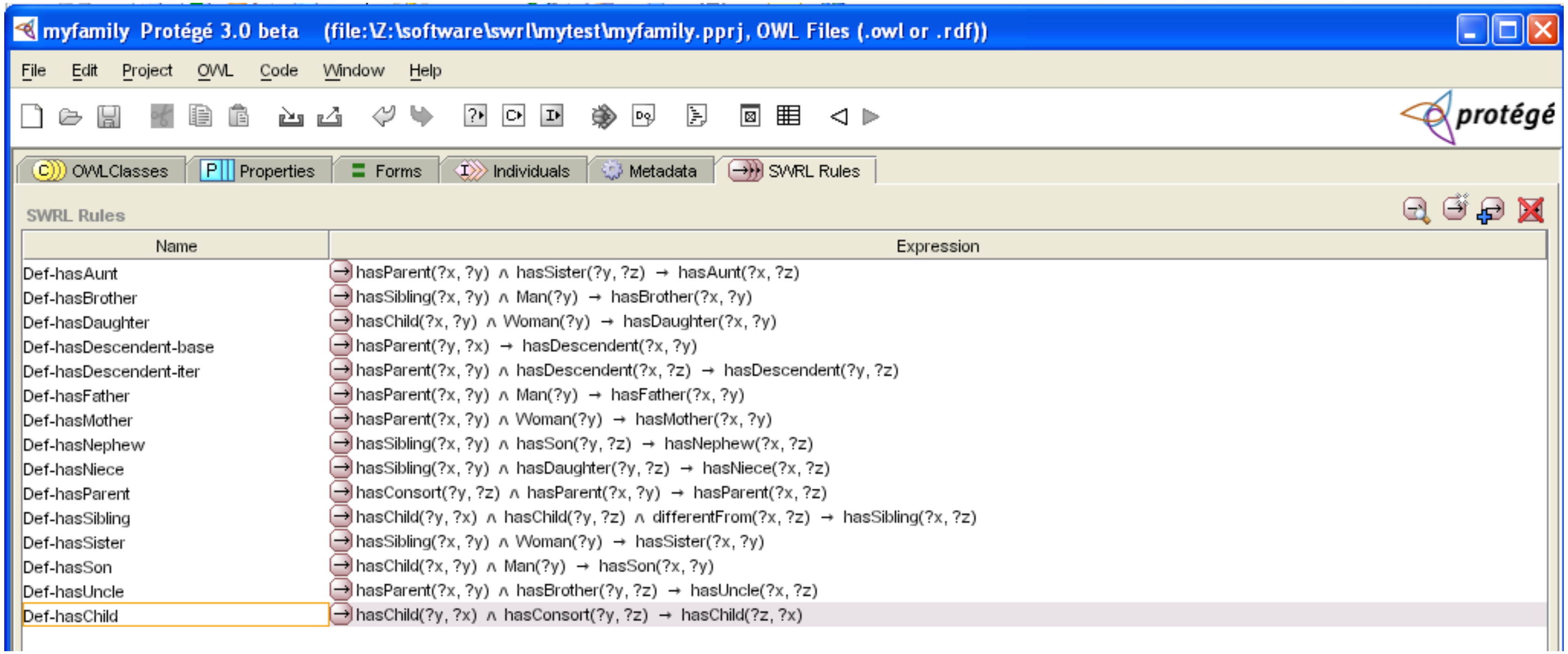
Semantic Web Architektur



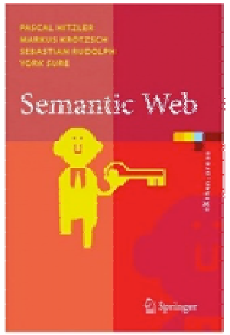
3.7 Web Ontology Language (OWL)

Nächste Vorlesung:

Regeln und Semantic Web Rule Language



Literatur



» P. Hitzler, M. Krötzsch, S. Rudolph, Y. Sure
Semantic Web Grundlagen, Springer, 2008.

Literatur

- Blog
  http://sw0809.blogspot.com/
- Materialien-Webseite
  http://www.hpi.uni-potsdam.de/meinel/teaching/semantic_web_ws08090.html

- bibsonomy - Bookmarks
  http://www.bibsonomy.org/user/lysander07/sw0809-10