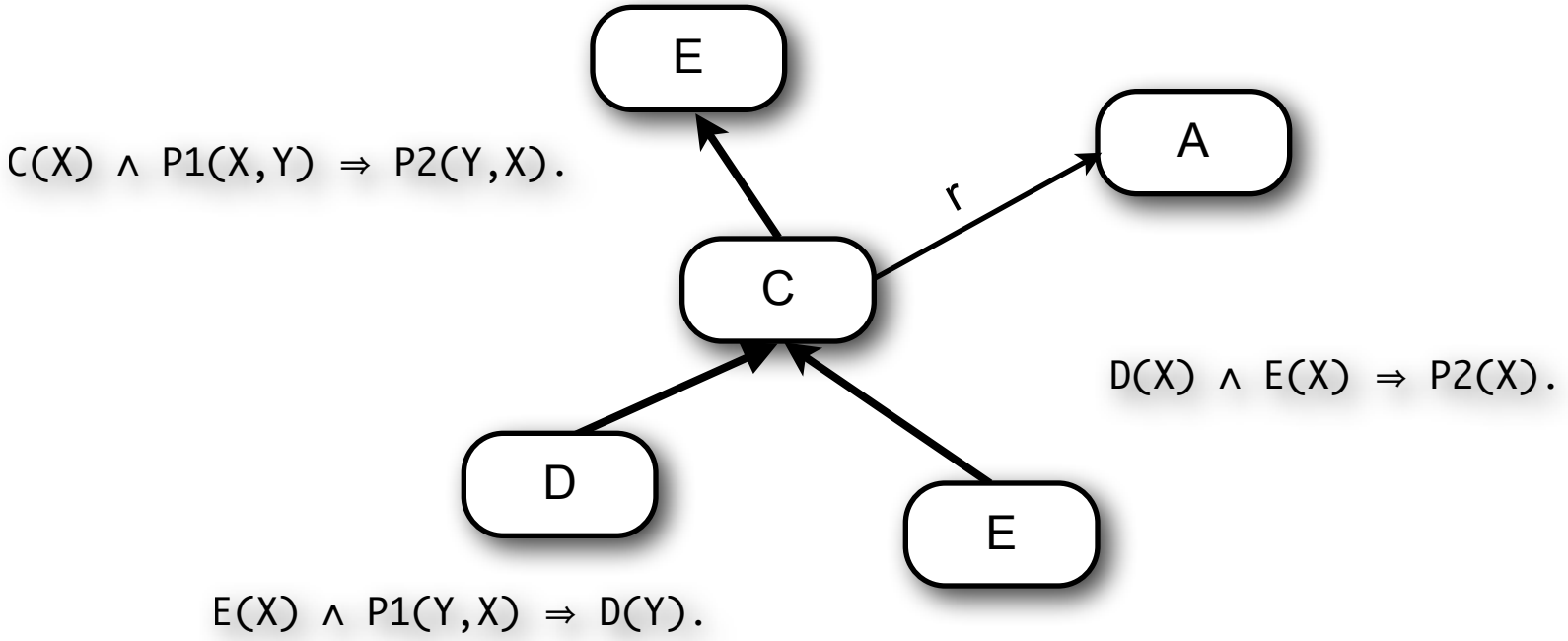
Verification of Ontologies with Rules

Seminar aus maschinellem Lernen WS 2011/12

Dr. Heiko Paulheim, Frederik Janssen

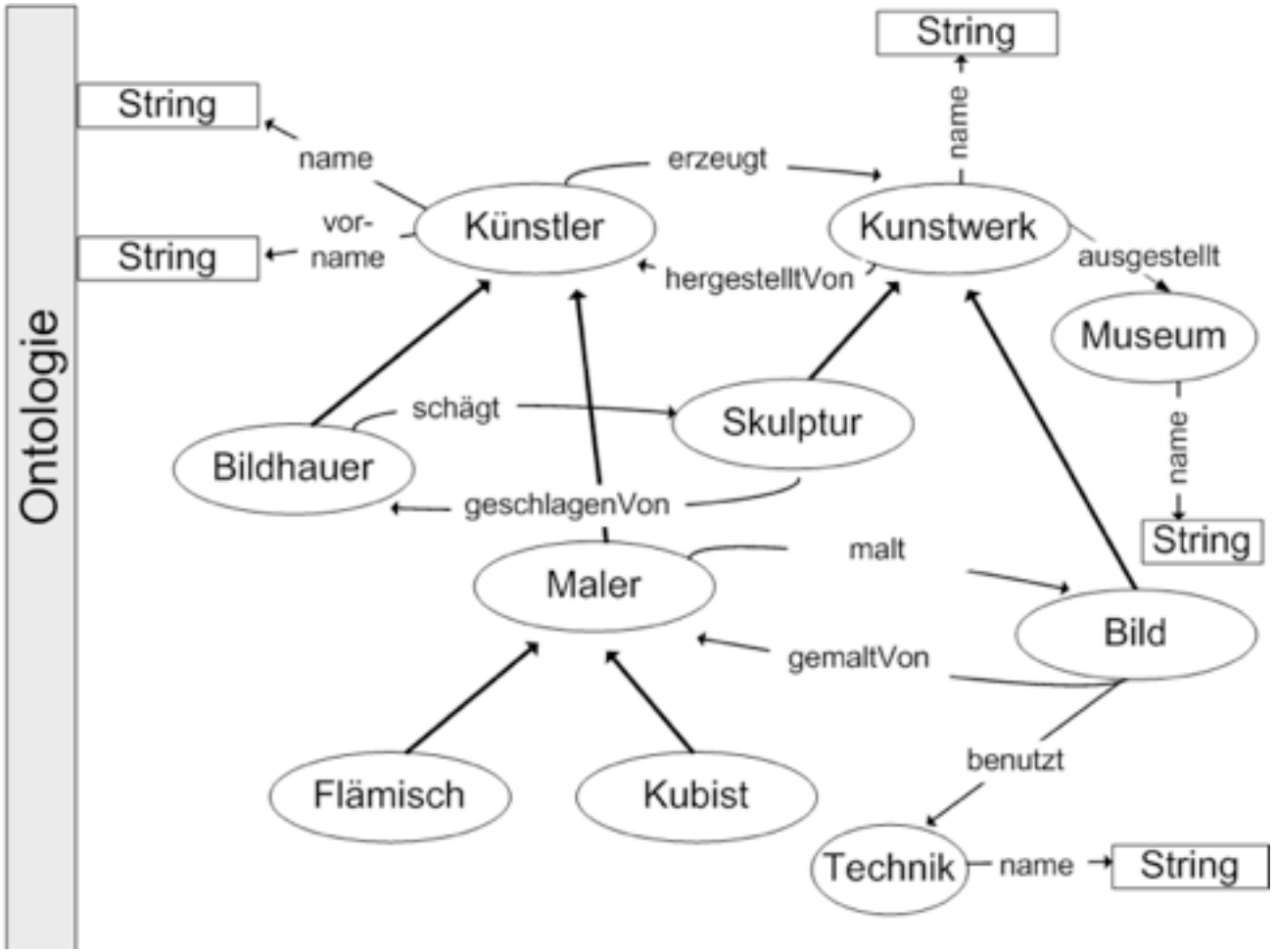


Ontologien...

TECHNISCHE UNIVERSITÄT DARMSTADT

?

Ontologien...


Ontologie
String
name
String
vor-
name
Künstler
erzeugt
String
name
Kunstwerk
hergestelltVon
ausgestellt
Museum
name
String
schägt
Skulptur
Bildhauer
geschlagenVon
malt
Maler
Bild
gemaltVon
benutzt
Flämisch
Kubist
Technik
name
String

Wieso Ontologien mit Regeln?

TECHNISCHE UNIVERSITÄT DARMSTADT

- vieles Wissen lässt sich deskriptiv kodieren, z.B.
  - Maler *is-a* Künstler
  - Künstler *erzeugt* Kunstwerk
  - Kunstwerk *hergestelltVon* mind. 1 Künstler (Property *hergestelltVon* hat minimale Kardinalität von 1)
  - ...
- formal: Description Logic (DL)
  - Vortrag nächste Woche: Concept Learning in Description Logics
- solches Wissen lässt sich in OWL (Web Ontology Language) beschreiben

Wieso Ontologien mit Regeln?

TECHNISCHE UNIVERSITÄT DARMSTADT

- **aber:** ein Großteil an Wissen lässt sich so schlecht auffassen, z.B.:
  - jede Schwester von einem meiner Elternteile ist meine Tante
  - dagegen leicht als Regel definierbar:
    - `sister(x,Y), parent(Y) => aunt(x)`

- -> Regeln müssen her

Regeln...

TECHNISCHE UNIVERSITÄT DARMSTADT

- gebe ich hier nur abstrakt wieder
- müssen aber immer aus einer konkreten Sprache stammen
- verschiedene Standards (SWRL, RuleML, ...)
- RuleML zum Beispiel unterscheidet verschiedene Untersprachen
- unterteilt nach Mächtigkeit
  - Datalog RuleML
  - Horn-Logic RuleML
  - FOL (First-Order Logic) RuleML
  - ...
- Mächtigkeit kommt immer zum Preis von Berechenbarkeit/Performanz

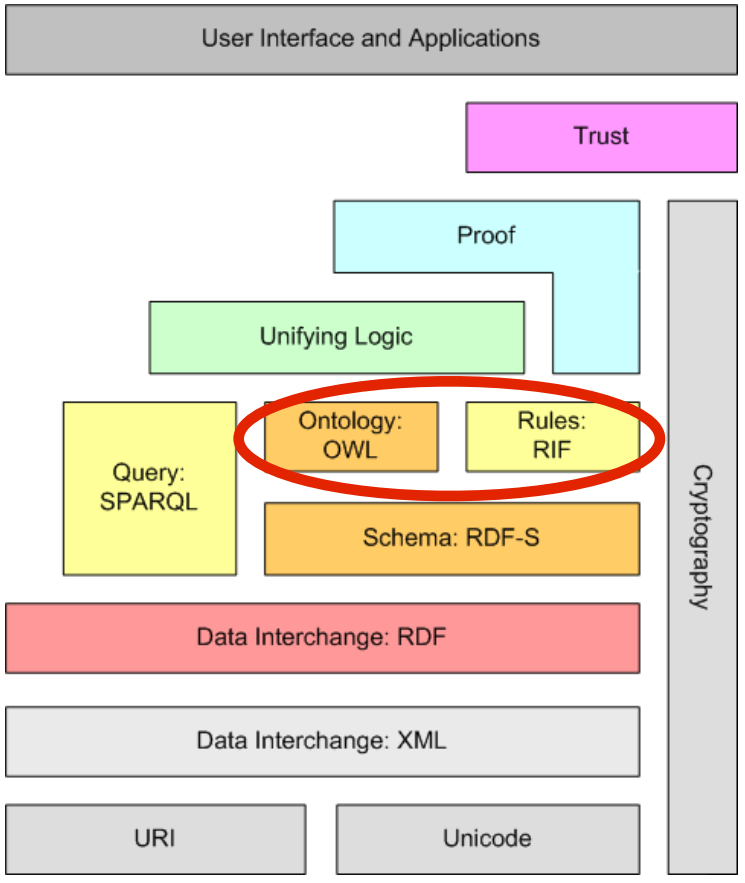
siehe z.B. http://ruleml.org/modularization/

Der Semantic Web Stack

here be dragons...

Semantic-Web-Technologie (Fokus der Vorlesung)

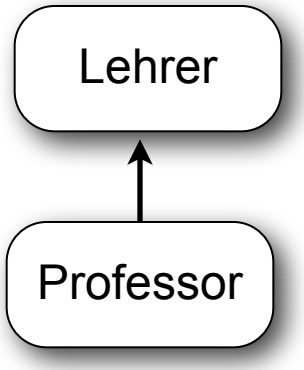
Technische Grundlagen



Berners-Lee (2009): *Semantic Web and Linked Data*
http://www.w3.org/2009/Talks/0120-campus-party-tbl/

Schön. Ist jetzt alles besser?

- Beobachtung:
- viele Fakten lassen sich nun deskriptiv (mit OWL) **und** mit Regeln kodieren:



OWL:

```
<owl:Class rdf:ID="Professor">
  <rdfs:subClassOf rdf:resource="#Lehrer" />
</owl:Class>
```

$$Professor(x) \Rightarrow Lehrer(x)$$

am Beispiel Prolog:

```
Lehrer(x) :- Professor(x)
```

Das Problem...

TECHNISCHE UNIVERSITÄT DARMSTADT

- Beobachtung:
- viele Fakten lassen sich nun deskriptiv (mit OWL) **und** mit Regeln kodieren

=> Redundanz

=> Inkonsistenz

Das Problem...

TECHNISCHE UNIVERSITÄT DARMSTADT

- Außerdem: Ontologien mit Regeln sind komplexer
- -> mehr menschliche Fehler

Verifikation!

TECHNISCHE UNIVERSITÄT DARMSTADT

- viele Fehler, Redundanzen, etc. (Anomalien) können automatisch erkannt werden
- hier kommen die 3 Paper ins Spiel:
  - Klassifikation möglicher Anomalien
  - wie sie (autom.) erkannt werden können
  - wie man sie beheben kann

Verifikation! - Die Papers

TECHNISCHE UNIVERSITÄT DARMSTADT

- „Verification and Refactoring of Ontologies With Rules“
  - Joachim Baumeister, Dietmar Seipel
  - 2006
- „Towards the Verification of Ontologies with Rules“
  - Joachim Baumeister, Dietmar Seipel, Thomas Kleemann
  - 2007
- „Anomalies in Ontologies with Rules“
  - Joachim Baumeister, Dietmar Seipel
  - 2010

Verifikation!

TECHNISCHE UNIVERSITÄT DARMSTADT

- erkennen von Anomalien umgesetzt in Mischung aus Prolog und Datalog
- -> eigene Sprache „Datalog*“
- mehr dazu später

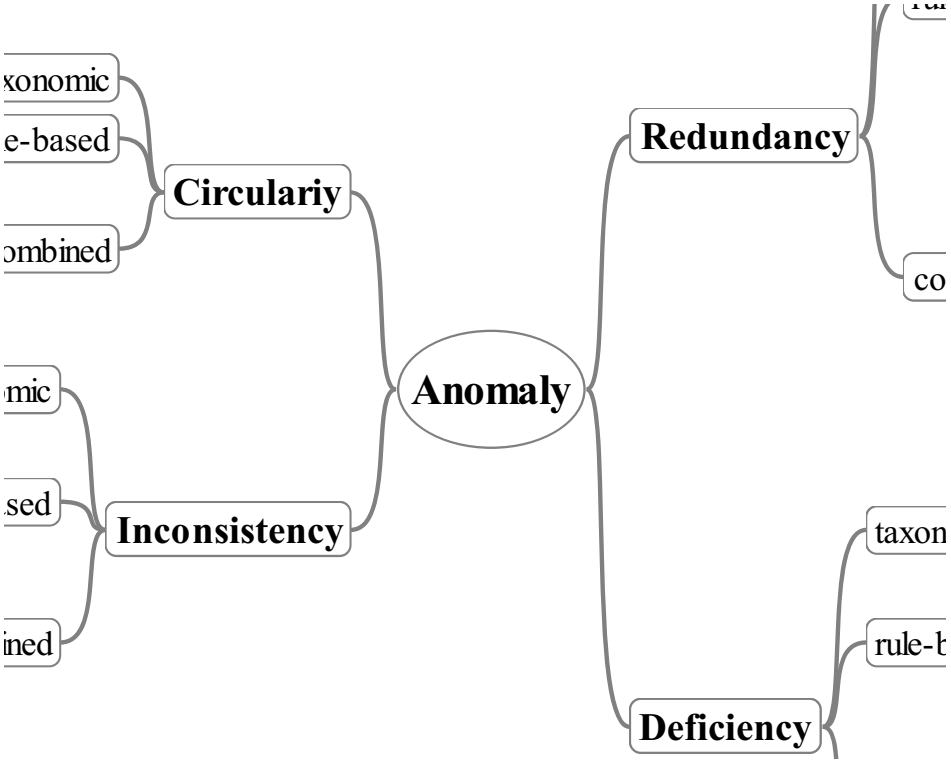
Vereinfachungen

TECHNISCHE UNIVERSITÄT DARMSTADT

- alle Anomalien zu finden ist ein unentscheidbares Problem
- daher Vereinfachungen
- nur Untermenge von OWL DL wird betrachtet:
  - Klasseneigenschaften:
    - Unterklassen (owl:subClassOf)
    - Komplement (owl:complementOf)
    - Disjunktheit (owl:disjointWith)
  - Property-Eigenschaften *(neu in Paper 3 in kursiv)*:
    - Transitivität (owl:transitiveProperty)
    - *Symmetrie (owl:symmetricProperty)*
    - *Definitions- und Wertebereich (rdfs:domain bzw. rdfs:range)*
    - *Kardinalitäteinschränkungen (owl:minCardinality, owl:maxCa..)*

Arten von Anomalien

- Zirkularität (circularity)
- Inkonsistenz
- Redundanz
- Unzulänglichkeit (deficiency)

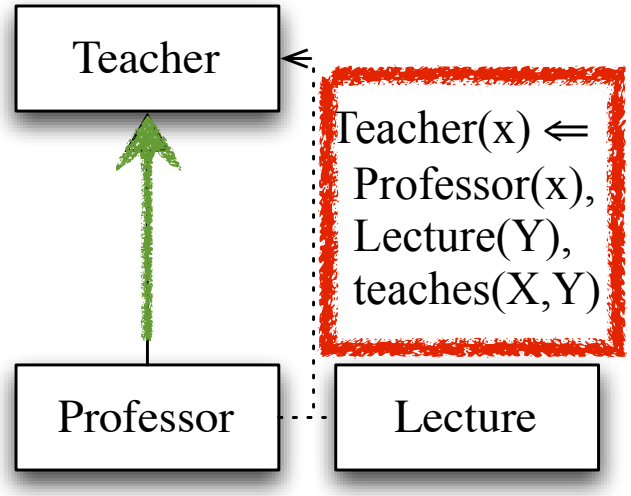


Beispiel 1: Redundanz von Regeln

TECHNISCHE UNIVERSITÄT DARMSTADT

subClassOf(Professor, Teacher)

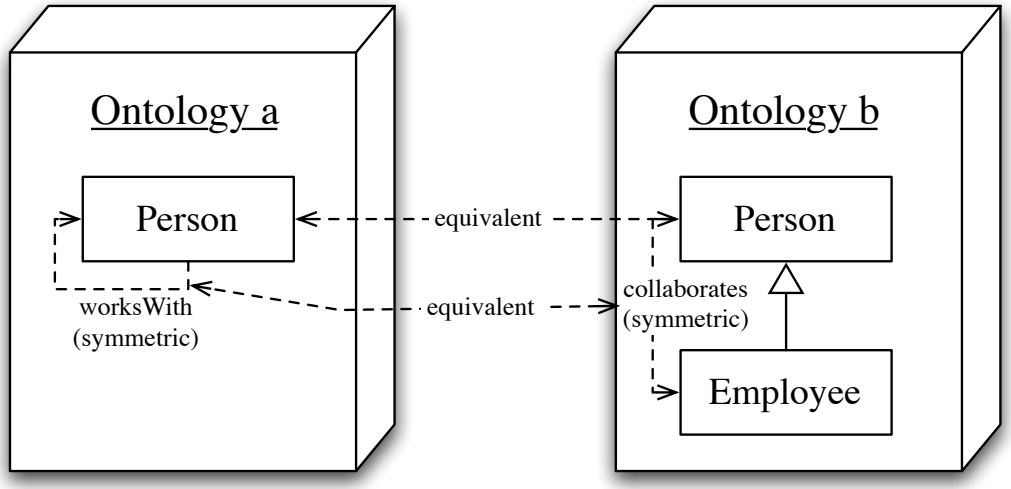
Professor(x) ∧ Lecture(Y) ∧ teaches(X,Y)
⇒ Teacher(x)



Beispiel 2: Redundanz in Regeln

TECHNISCHE UNIVERSITÄT DARMSTADT

a:worksWith(a:Person, a:Person)
b:collaborates(b:Person, b:Employee)
equivalentClasses(a:Person, b:Person)
equivalentProperties(a:worksWith, b:collaborates)
subClassOf(b:Employee, b:Person)

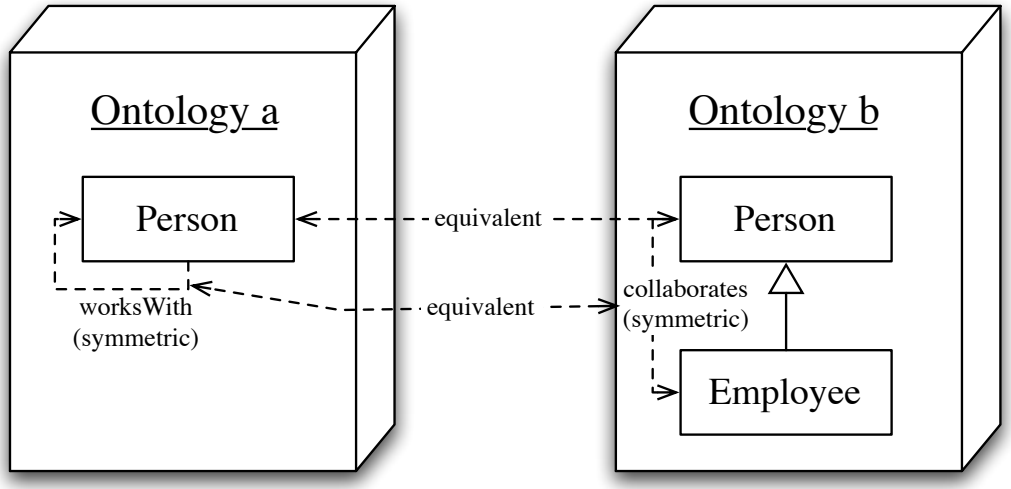


$a:P(X) \wedge a:worksWith(X,Y) \wedge b:collaborates(Y,X) \Rightarrow b:E(Y)$

Beispiel 2: Redundanz in Regeln

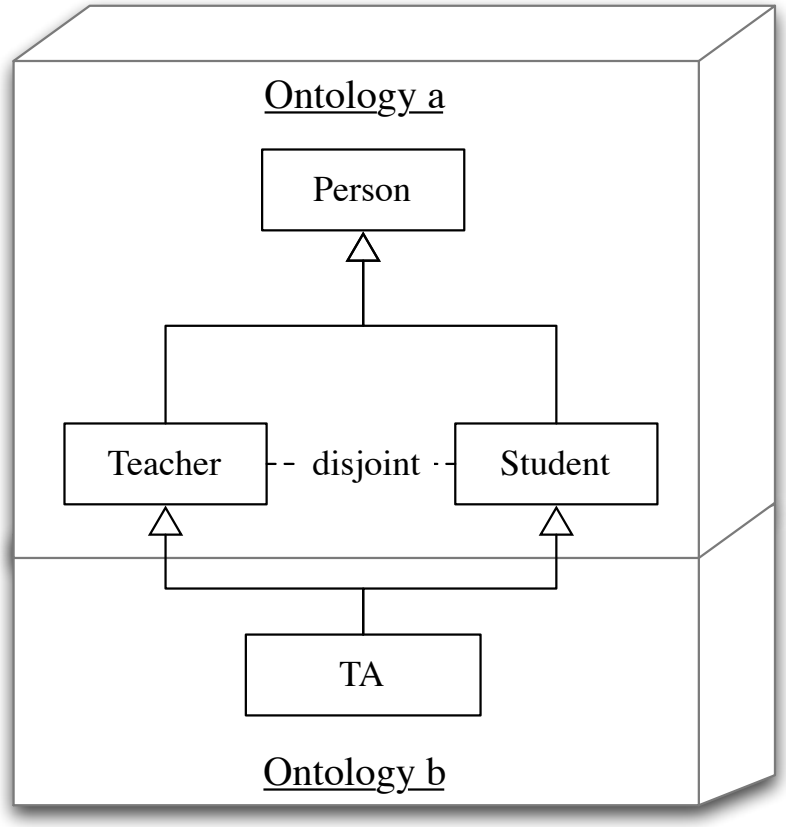
TECHNISCHE UNIVERSITÄT DARMSTADT

```
a:worksWith(a:Person, a:Person)
b:collaborates(b:Person, b:Employee)
equivalentClasses(a:Person, b:Person)
equivalentProperties(a:worksWith, b:collaborates)
subClassOf(b:Employee, b:Person)
```

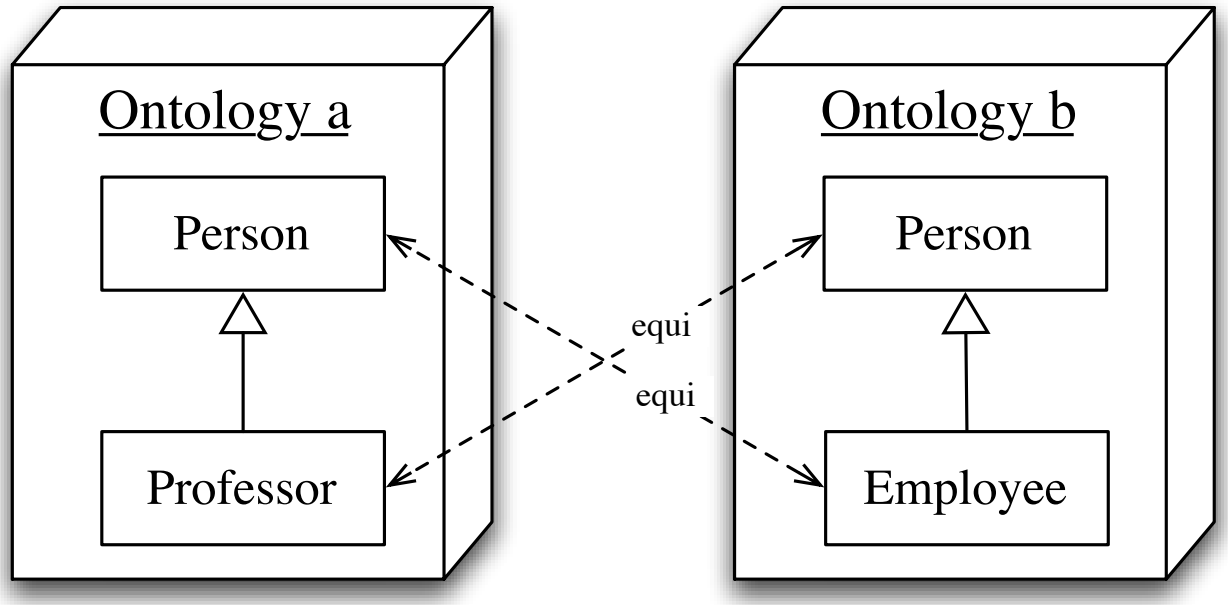


$a:P(X) \wedge a:worksWith(X,Y) \wedge b:collaborates(Y,X) \Rightarrow b:E(Y)$

Beispiel 3: Inkonsistenz

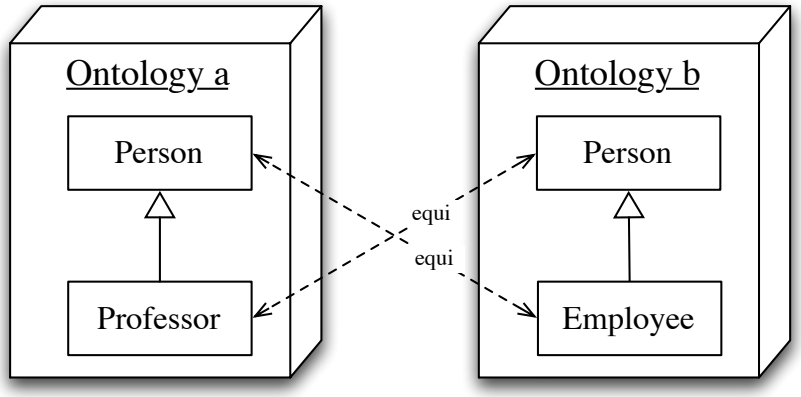


Beispiel 4: Zirkularität



Beispiel 4: Zirkularität

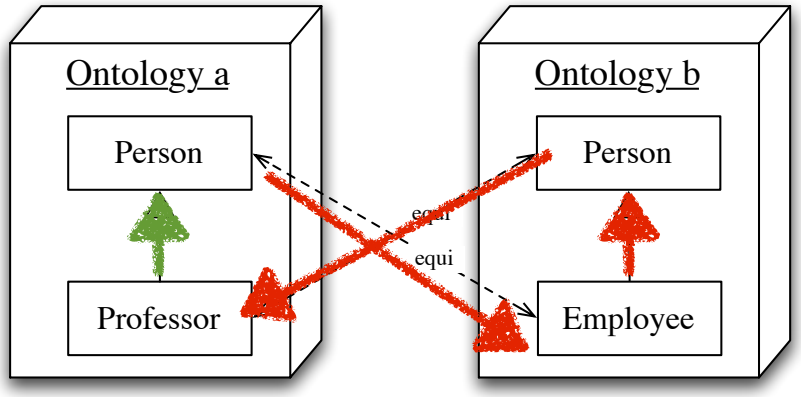
```
anomaly(exact_circularity, [E, F]) :-
   derives(E, F),
   E \= F,
   tc_derives(F, E).
```



Beispiel 4: Zirkularität

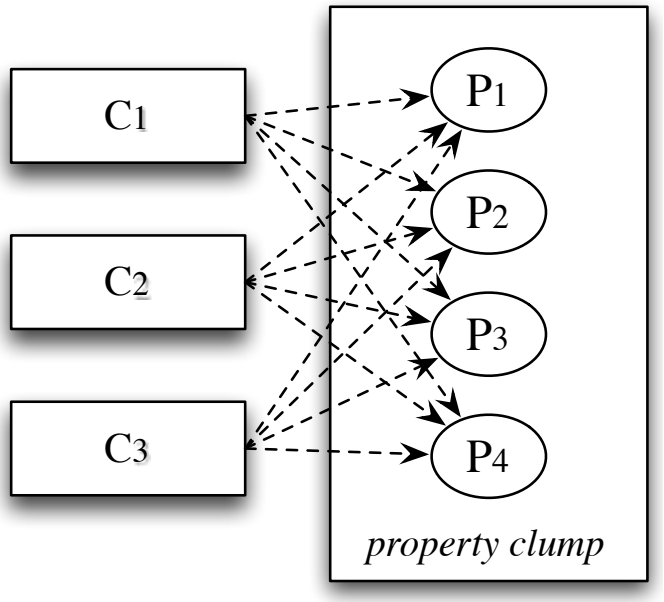
```
anomaly(exact_circularity, [E, F]) :-
   derives(E, F),
   E \= F,
   tc_derives(F, E).

derives(Professor, Person)
Professor \= Person
tc_derives(Person, Professor)
```



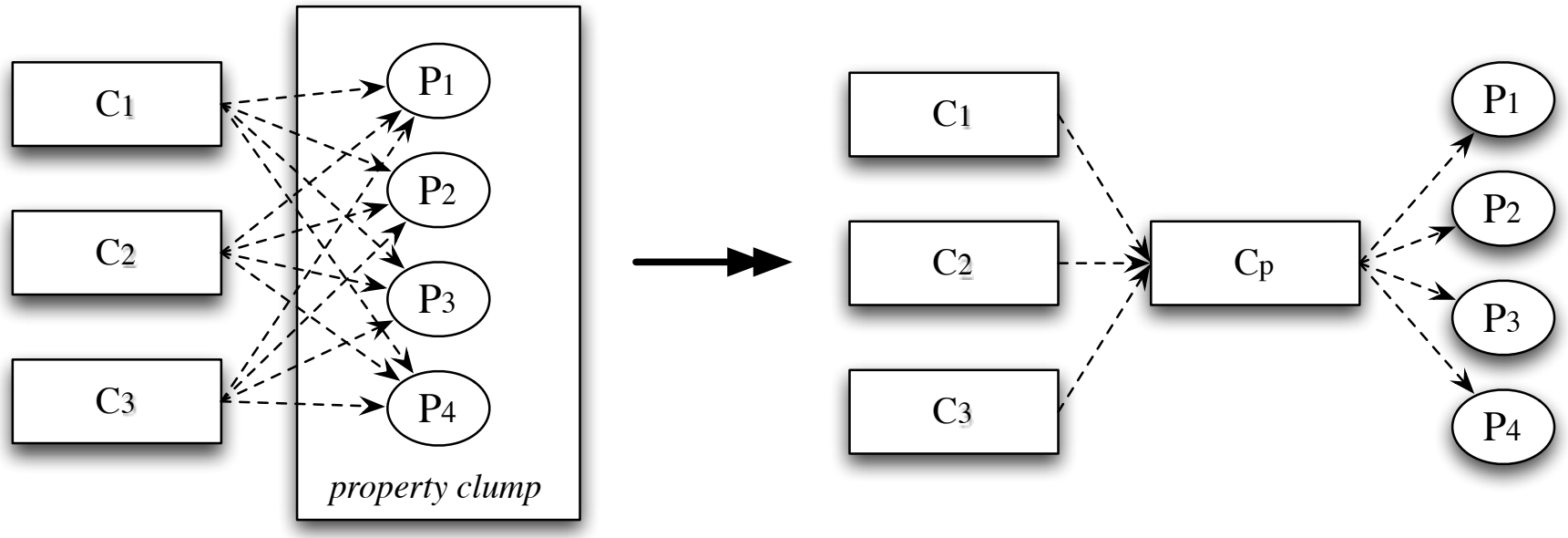
Beispiel 4: Unzulänglichkeit (Deficiency)

- mehrere Klassen teilen eine Menge von Properties
- sog. „Property Clump“



Beispiel 4: Unzulänglichkeit (Deficiency)

- Lösung ist einfach
- Zusammenfassen des Property Clumps zu einer neuen Klasse:



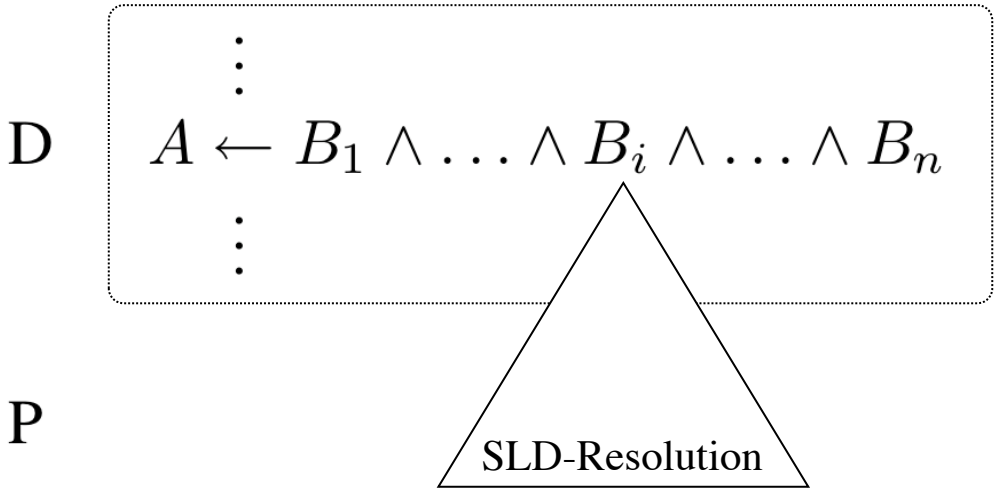
Implementierung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Mix aus Datalog und Prolog
- wieso?
  - Prolog ist zwar mächtiger als Datalog, aber nicht deterministisch
  - Datalog dagegen nicht ausreichend (nur Atome in Prädikaten, keine Funktionssymbole)
  - determin. Charakteristika aus Datalog werden mit Ausdrucksstärke von Datalog kombiniert
  - -> Kompromisslösung „Datalog*“

Implementierung

- größter Unterschied:
  - Literale einer Datalog*-Regel können auf zwei Arten hergeleitet werden:
  - mittels **Forward-Chaining** (wie in **Datalog**)
  - mittels **Backward-Chaining** (SLD-Resolution, wie in **Prolog**)



Stratifizierung

- Konzept aus Datalog
- Regeln werden in Schichten (Strata) aufgeteilt
- zuerst werden Regeln aus Schicht 1 ausgewertet, dann Regeln aus Schicht 2, etc.
- erlaubt Verwendung negierter Literale trotz Fixpunkt-Semantik

2

```
anomaly(subsumed_rule, [R1, R2]) :-
      rule(R1), ...,
      not(rule_subsumes_check(R2, R1)).
```

```
anomaly(lazy_element, E) :-
      element(E), ...,
      not(rule_predicate(E)).
```

etc.

1

```
rule_subsumes_check(R1, R2) :-
      ...
```

```
clause_subsumes(Cs1, Cs2) :-
      checklist( implies, Cs1, Cs2 ).
```

etc.

```
rule_predicate(E) :-
      rule(Rule),
      ( head_predicate(Rule, E)
      ; body_predicate(Rule, E) ).
```

Bemerkungen u. Kritik

TECHNISCHE UNIVERSITÄT DARMSTADT

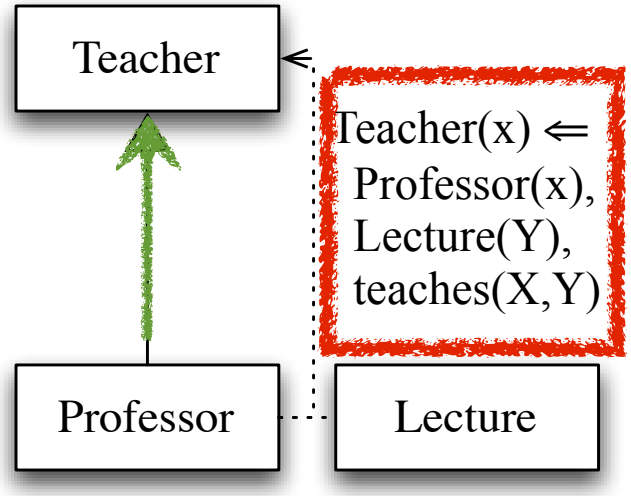
- Typisierung der Anomalien eindeutig?

Beispiel 1: Redundanz von Regeln (Wdh.)

TECHNISCHE UNIVERSITÄT DARMSTADT

subClassOf(Professor, Teacher)

Professor(x) ∧ Lecture(Y) ∧ teaches(X,Y)
⇒ Teacher(x)

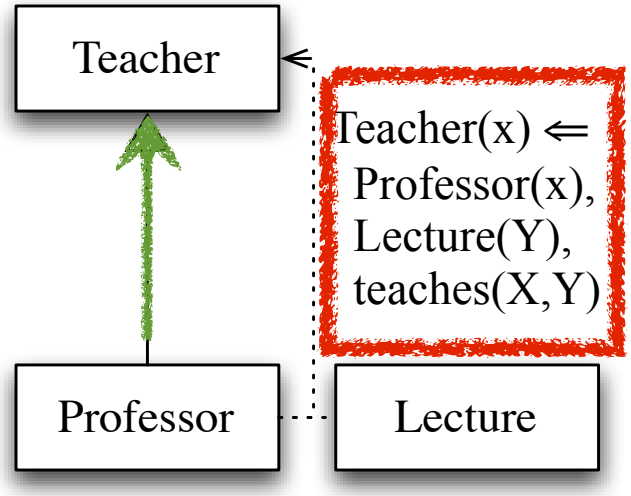


Beispiel 1: Redundanz von Regeln (Wdh.)

TECHNISCHE UNIVERSITÄT DARMSTADT

```
subClassOf(Professor, Teacher)

Professor(x) ∧ Lecture(Y) ∧ teaches(X,Y)
   ⇒ Teacher(x)
```



⇒ Inkonsistenz?

Bemerkungen u. Kritik

TECHNISCHE UNIVERSITÄT DARMSTADT

- Hinweis darauf, wie konkrete Anomalien „hergeleitet“ werden, fehlt
- woher weiß man, ob eine sinnvolle Anzahl an Anomalien abgedeckt ist?

Bemerkungen u. Kritik

TECHNISCHE UNIVERSITÄT DARMSTADT

- Real-World-Tests wären interessant gewesen
- Problem hier aber: bisher existieren keine größeren Ontologien mit Regeln
- nur wenige „Toy Examples“

Bemerkungen u. Kritik

TECHNISCHE UNIVERSITÄT DARMSTADT

- weitere Hinweise zur Implementierung von Datalog* fehlen
- z.B.: wann wird ein Literal einer Datalog*-Regel mittels Forward-, wann mittels Backward-Chaining hergeleitet

Bemerkungen u. Kritik

TECHNISCHE UNIVERSITÄT DARMSTADT

- Anomalien in aufeinanderfolgenden Papers mal neu eingeführt, mal wiederverwendet

Implementierte Anomalien: Redundanz

TECHNISCHE UNIVERSITÄT DARMSTADT

Paper 1

rule_subsumption
implication_of_superclasses
redundant_transitivity
redundancy_in_antecedent
unsatisfiable_condition

Paper 2

implication_of_superclasses
redundancy_in_antecedent
redundant_transitivity
subsumed_rule
redundant_range
redundant_domain
redundant_mincardinality_0
maxcardinality_0
mincardinality_1

Paper 3

unsatisfiable_condition
subsumed_rule
redundant_mincardinality_0
redundant_transitivity_hb
redundant_symmetry_hb
redundant_derivation
redundant_transitivity_b
redundant_symmetry_b
unsupported_condition
subsumed_maxcardinality_1

Diskussion

TECHNISCHE UNIVERSITÄT DARMSTADT

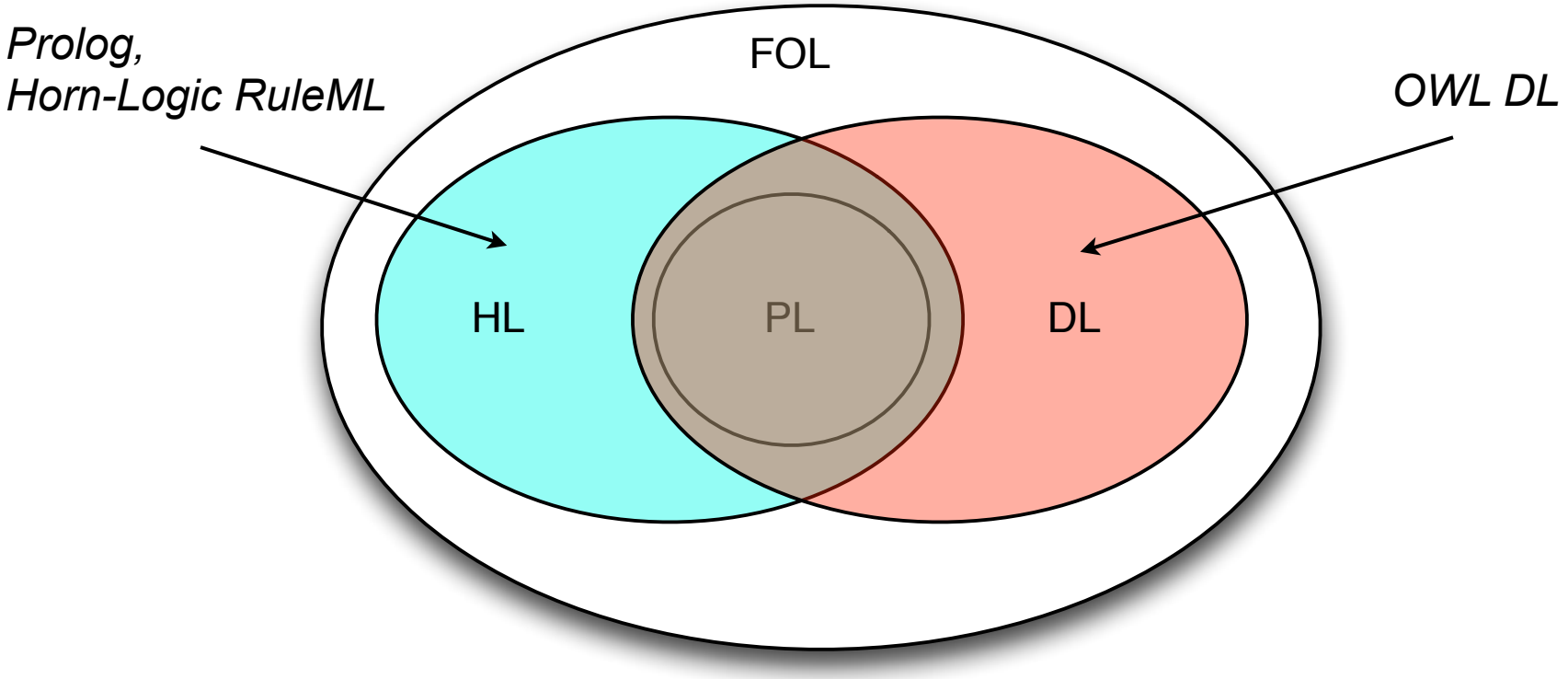
Fragen?

Unterschiede der Papers

TECHNISCHE UNIVERSITÄT DARMSTADT

- 1. Paper von 2006, 2. von 2007 und 3. von 2010
- Anomalienerkennung mittels Prolog in 1 und 2, Datalog* in 3
- 1. und 2. Paper sehr knapp, 3. Paper enthält dagegen viele anschauliche Beispiele

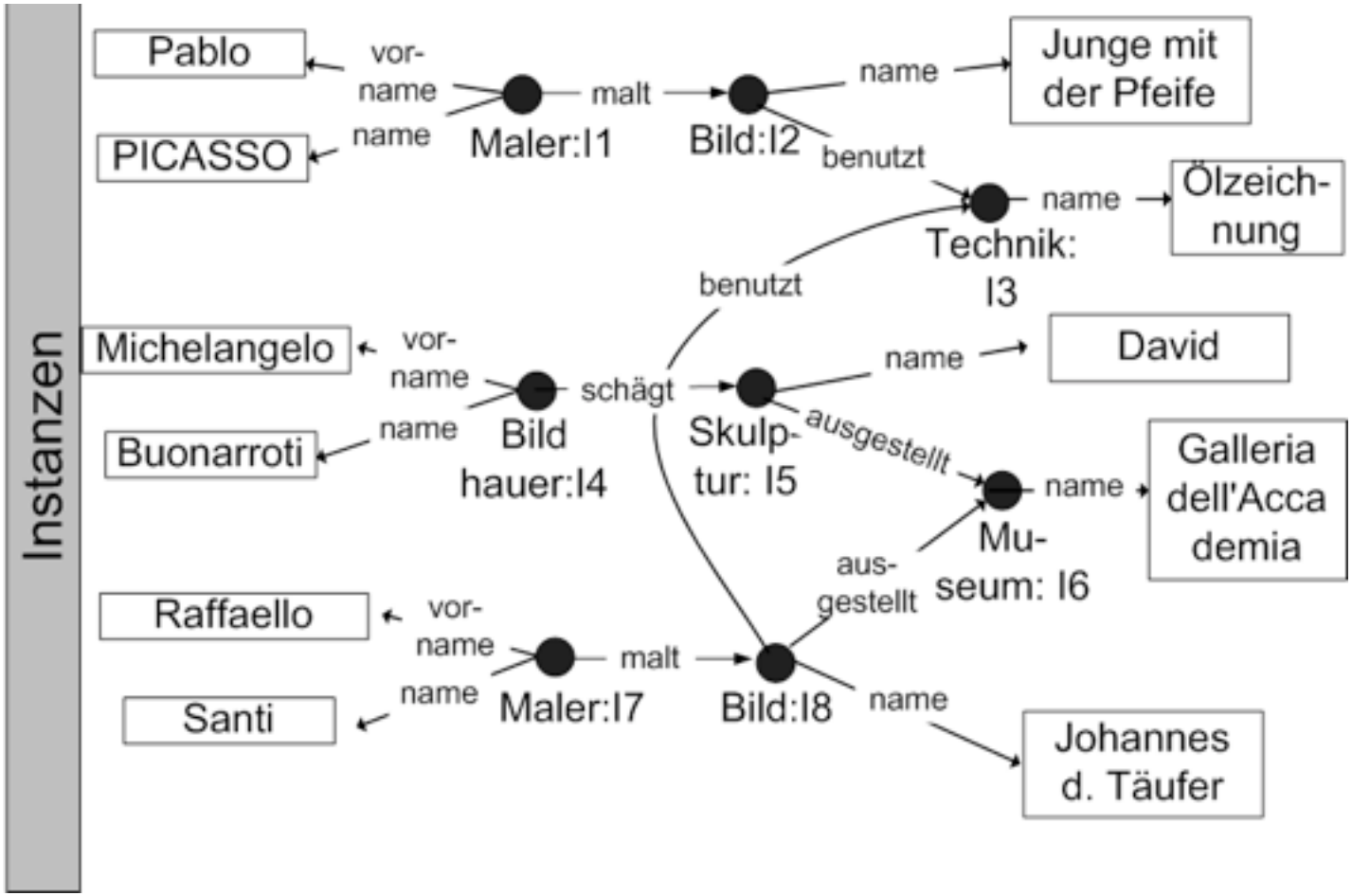
Die verschiedenen Logiken



PL - Propositional Logic (Aussagenlogik)
FOL - First Order Logic (Prädikatenlogik)

HL - Horn Logic
DL - Description Logic

Ontologien...


Instanzen
Pablo
vor-
name
PICASSO
name
Maler:I1
malt
Bild:I2
name
Junge mit
der Pfeife
benutzt
Technik:
I3
name
Ölzeich-
nung
benutzt
Michelangelo
vor-
name
Buonarroti
name
Bild
hauer:I4
schägt
Skulp-
tur: I5
name
David
ausgestellt
Mu-
seum: I6
name
Galleria
dell'Acca
demia
aus-
gestellt
Raffaello
vor-
name
Santi
name
Maler:I7
malt
Bild:I8
name
Johannes
d. Täufer