

Combining Ontology and Rule Languages on the Semantic Web

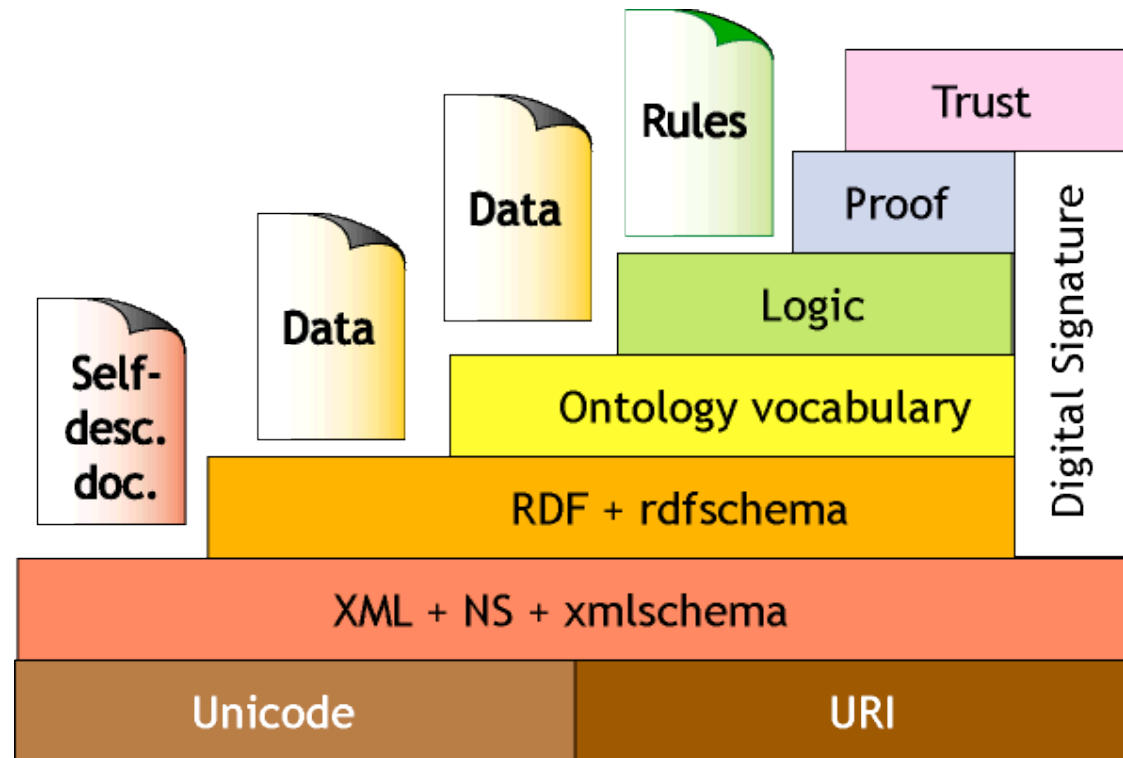
Case study: “A Policy language for a Pervasive Computing Environment”



Agenda

- Motivation
- Requirements and Use cases for a Semantic Web Rule Language
- Approaches for combining ontology and rule languages
- Case Study: REI - A Policy Language

Motivation





Motivation (Continuation)

What are rules?

- Statements that express a business/domain policy.

They specify:

- Derivations

A gold customer is a customer with over 1M in deposits

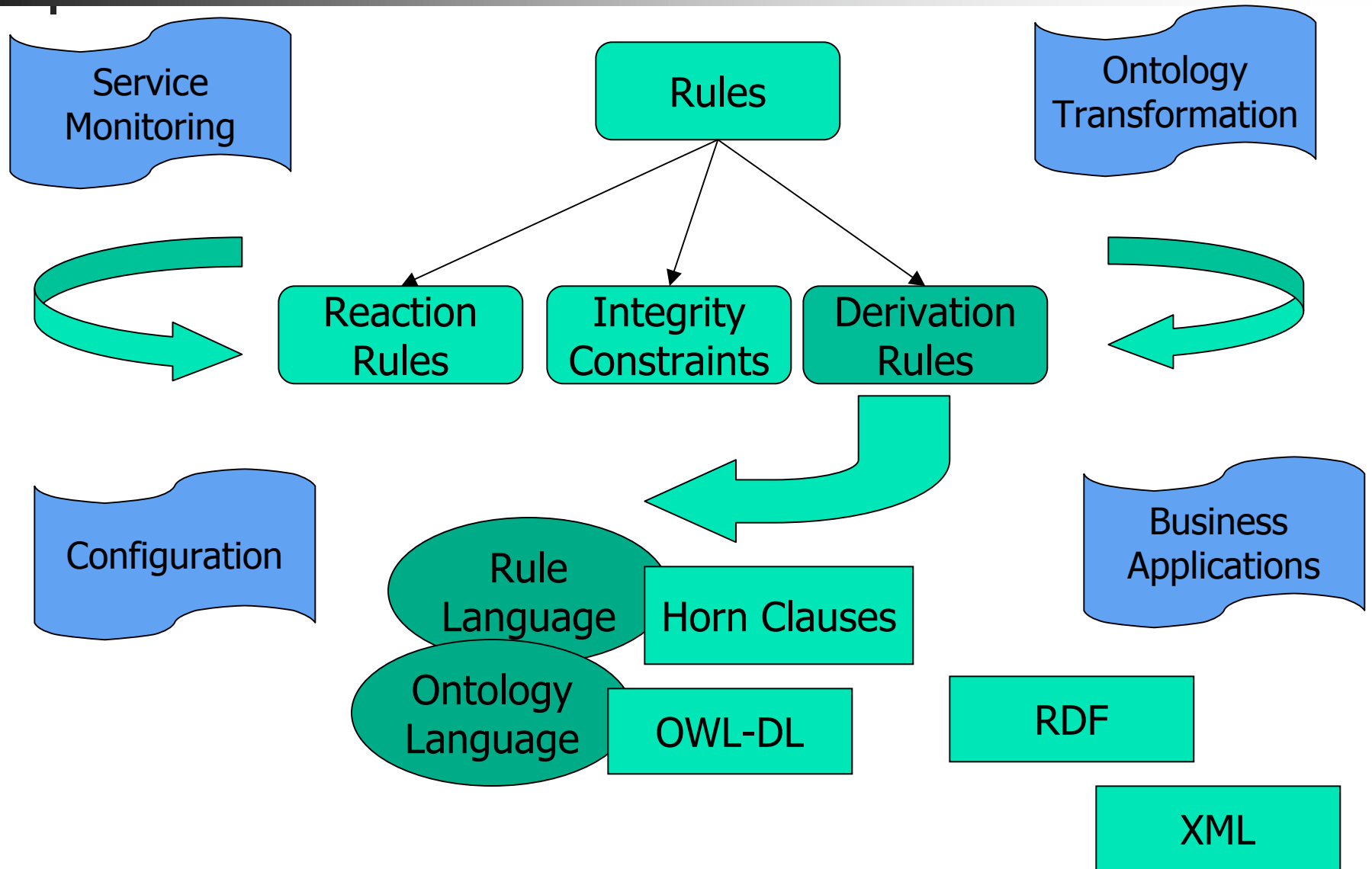
- Static and dynamic integrity constraints

The driver of a rental car must be 25 years or older

- Reactions

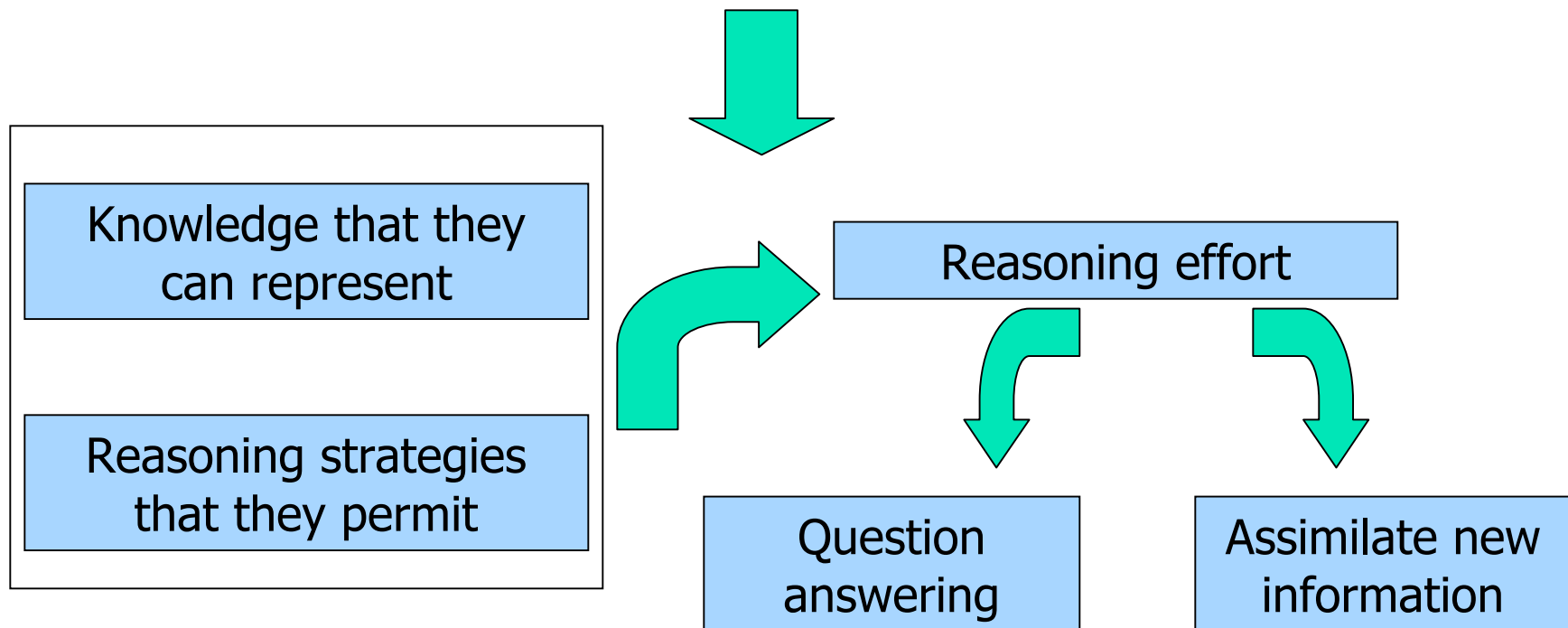
When a share price drops by more than 5%, then sell it

Requirements for a Semantic Web Rule Language



Approaches - Expressiveness and Tractability

Tradeoff between expressiveness of the representational language and its tractability





Approaches - Expressiveness and Tractability

Two pseudo-solutions:

Techniques for optimizing performance
Return an answer after a certain amount of time

Another pseudo-solution

Controlling the incompleteness of the knowledge represented
“Representation language is not used to capture complex details of the domain but to avoid representing details that may not be known”



Restricting the logical form of the KB
(Database, Logic Program, Description Logics, Frame-description or combinations)



Approaches for combining Ontology and Rule Languages

1. Rules in OWL-DL: Expressing rules in OWL-DL (Could do with some syntactic sugar)

- Restricted to tree-like rules
- Provides both existentially and universally qualified variables

2. Axiomatic: Extending the axioms of OWL-DL to include Horn clauses.

(SWRL - W3C Proposal for a Semantic Web Rule Language)



Approaches for combining Ontology and Rule Languages

3. Two components: OWL-DL Knowledge Base and Rule Knowledge Base.

3.1 Restricting rules to *DL-safe* ones. (Motik, Sattler, Studer. "Query Answering for OWL-DL with Rules")

3.2 AL-Log. (Donini, Lenzerini, Nardi. "AL-Log: Integrating Datalog and Description Logics")

3.3 CARIN. (Levy, Rousset). "CARIN: A representation Language Combining Horn rules and Description Logics".

4. Intersection: Expressive intersection between Logic Programs without function symbols and OWL-DL.

Description Logic Programs (DLP) and Description Horn Logic (DHL).

(Grosz, Volz, Horrocks, Decker. "Description Logic Programs: Combining Logic Programs with description Logic")



Approaches for combining Ontology and Rule Languages

2. **Axiomatic**: SWRL - Semantic Web Rule Language

- Extends the set of OWL axioms to include Horn-like rules.
- Antecedent and consequent atoms are concepts or roles of the form $C(x)$, $P(x,y)$, $\text{sameAs}(x,y)$, $\text{differentFrom}(x,y)$ where x and y are either variables, individuals or data values.
- Built-Ins for comparisons, mathstrings, date, time, Etc.
- Undecidable, simulates role value maps.

Examples:

▪ `hasParent(?x1,?x2) ^ hasBrother(?x2,?x3) =>hasUncle(?x1,?x3)`

`Implies (Antecedent (hasParent (I-variable (x1) I-variable (x2))
hasBrother (I-variable (x2) I-variable (x3)))`

`Consequent (hasUncle (I-variable (x1) I-variable (x3))))`

▪ `artist(?x) ^ (<=1 artistStyle)(?x) ^ creator(?z,?x)
=>(<=1 stylePeriod)(?z)`



Approaches for combining Ontology and Rule Languages

3.1 Two Components: Rules are *DL-safe*

- Concepts and roles are allowed to occur as atoms in head and body of rules
- Each variable in a rule is required to occur in a non-DL atom.
 $\mathcal{SHIQ}(\mathcal{D})$ infinite tree models can be “wound” up to be finite, but with individuals which are “known”.
- Algorithm for query answering in $\mathcal{SHOIN}(\mathcal{D})$ without nominals ($\mathcal{SHIQ}(\mathcal{D})$) reduces DL KB to Disjunctive Datalog Programs, then adds Rule KB.

Examples

$\text{BadChild}(\mathbf{x}) \leftarrow \text{GrandChild}(\mathbf{x}), \text{parent}(\mathbf{x}, \mathbf{y}), \text{parent}(\mathbf{z}, \mathbf{y}), \text{hates}(\mathbf{x}, \mathbf{z}),$
 $\text{O}(\mathbf{x}), \text{O}(\mathbf{y}), \text{O}(\mathbf{z}).$

$\text{O}(\alpha)$ for each explicitly named individual α (in the A-Box)

$\text{father} \sqsubseteq \text{parent} \quad \text{father}(\text{Cain}, \text{Adam}), \text{father}(\text{Abel}, \text{Adam})$



Approaches for combining Ontology and Rule Languages

3.2 Two Components: AL-Log

- Knowledge base is a pair (Σ, Π) . Σ is an *ALC* knowledge base. Π is a constrained Datalog program.
- A constrained Datalog clause is of the form $\gamma \ \& \ \beta_1, \dots, \beta_m$ where γ is a Datalog clause and β_i is a constraint of the form $s:C$. Every variable appearing in a DL atom appears in the Datalog clause
- Query answering is based on constrained SLD-derivation and tableau algorithm. A derivation terminates with the last query of the form $\& \ \beta_1, \dots, \beta_m$ which is a constrained empty clause
- Query answering is decidable

Examples

```
curr(X,Z) ← exam(X,Y), subject(Y,Z) & X:St, Y:Co, Z:Tp.  
exam(paul, ai) .
```

$AC \cup BC = Co$

$FP \subseteq FM \text{ NFP} = FP \cap \neg \exists TC.Co \text{ John:FP johnTCai}$



Approaches for combining Ontology and Rule Languages

3.3 Two Components: CARIN

- Concepts and Roles appear in the antecedent of rules
- Underlying Description Logics is $ALCNR$.
- Decidable for non-recursive Horn rules. For recursive rules, decidability can be regained by disallowing certain combinations of constructors and by requiring the rules to be *role-safe*.
- Query answering using constrained resolution and tableau algorithm.

Examples

`made-by(X,Y) ∧ no-fellow-company(Y) ⇒ price(X,usa,high)`

`made-by(X,Y) ∧ associate(Y,Z) ∧ american(Z) ∧ monopoly(Y,X,usa)
⇒ price(X,usa,high)`

`american_associate ≡ ∃ associate.american`



Approaches for combining Ontology and Rule Languages

4. Intersection: DLP and DLH

- Finds expressive intersection between the two KR's (DLH).
- Translation from the DLP fragment of DL to LP. Enables the rule KR to have access to DL ontological definitions in order to build rules on top of ontologies. (Some DL constructors cannot be mapped).
- Some fragment of DL inferencing is performed by LP engines.

Example. $C_1 \cup C_2 \subseteq D$ is equivalent to $D(x) \leftarrow C_1(x). \quad D(x) \leftarrow C_2(x).$

- Enables efficient LP inferencing algorithms.
- DL queries (class/property instance membership, subsumption, hierarchy, satisfiability) are reduced to logic programming queries.
- Deductive databases: Datalog



Comparison of approaches

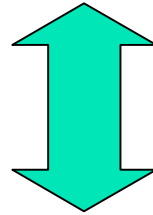
Approach	Expressivity	Decidability
OWL-DL	SHOIN(D) Tree-like rules	Decidable
Axiomatic (SWRL)	DL with role-value maps Unrestricted rules	Un-decidable
DL-safe rules	SHIQ(D) Concepts and roles in head and body of rules. DL-safe rules	Decidable
AL-LOG	ALC Concepts in body of rules	Decidable
CARIN	ALCNR Concepts and roles in body of rules. Role-safe rules	Decidable for non-recursive rules
Intersection	Some constructors of DL. Variants include if constructor occurs on left side or right side of rules. Unrestricted rules	Decidable



REI Policy language for pervasive computing environments

REI

Policy language that includes constructs for rights, prohibitions, obligations and dispensations, speech acts, meta-policies.



Expressive
Lightweight
Extensible

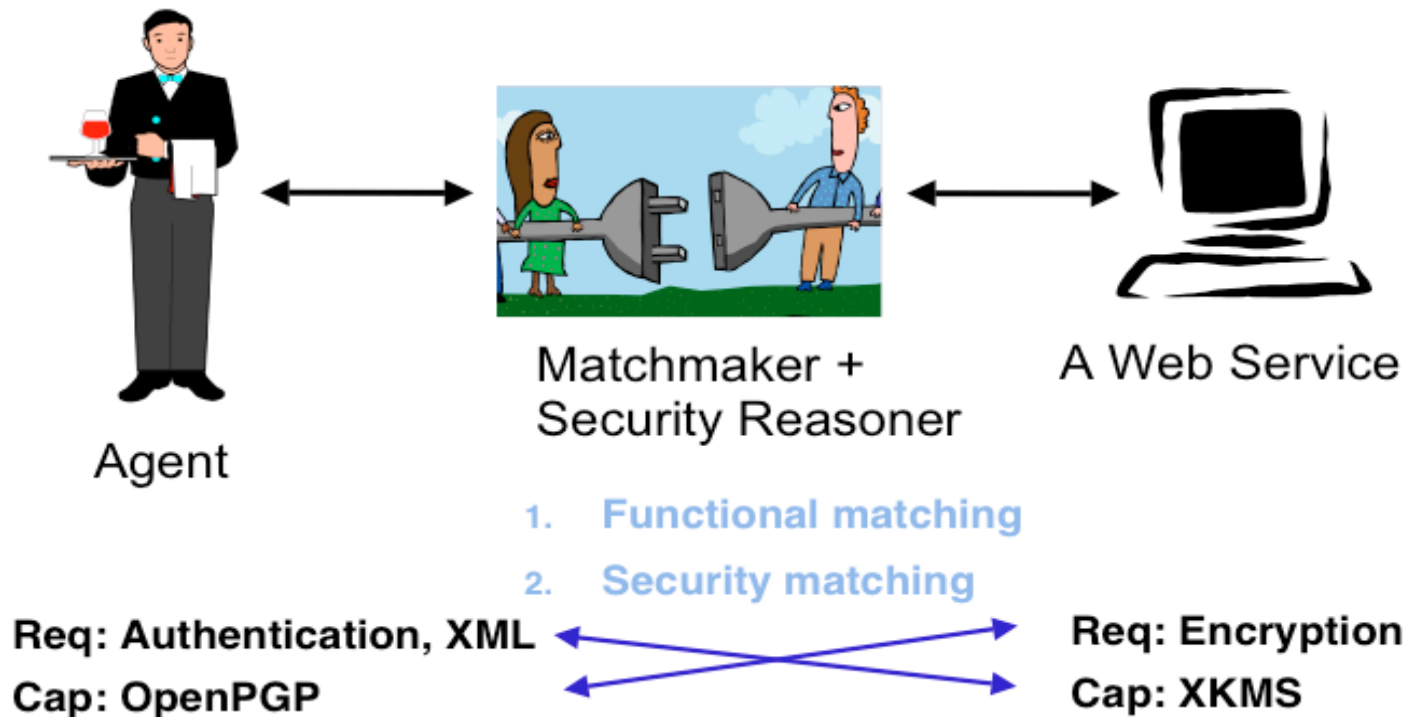
Pervasive computing environments

People and devices are mobile and use various wireless networking technologies to discover and access services and devices in their vicinity

REI Policy language for pervasive computing environments



Matching Security Annotations





Constructs REI Policy Language

1. Policy objects: Rights, Prohibitions, Obligations, Dispensations

policyObject(Action, Conditions)

has(Subject, PolicyObject)

Example:

```
has(Variable, prohibition(usefacultyPrinter,
student(Variable, 'UMD')))
```

2. Action Specifications: Action and its parameters

Action(ActionName, TargetObjects, Pre-Conditions, Effects)

Example:

```
Action(printOnePageHP, [printerHP],
(containsCartridge(printerHP), availablePaper(printerHP, X), X>1),
availablePaper(printerHP, X-1))
```

▪ **Action Operators:** Complex action descriptions. Operators are *Sequence*, *Non-deterministic*, *Repetition* and *Once*

Example:

```
Has(john, right(nond(seq(printUM, repetition(printColor)),
once(faxUM)), lab-member(john, 'AI')))
```



Constructs REI Policy Language

3. Speech Acts: Speech acts used to decentralize control: delegation, request, cancel and revocation. Nesting of speech acts and policy objects.

Example:

```
delegateSpeechAct(john, mark,  
                  right(printlabPrinter,  
                        (project(john, SomeProject),  
                               project(mark, SomeProject))))
```

```
Action(printLabPrinter, [laserPrinter123],  
        queue(laserPrinter123, 0),  
        queue(laserPrinter123, 1))
```

Request for an action causes an obligation on the receiver

Request for a right causes the receiver to delegate his right to the sender



Expressing REI in the different approaches

```
has(Variable, right(useStudentPrinter, student(Variable, 'UMD'))  
hasRight(X, useStudentPrinter) :- student(X) .
```

OWL-DL constructors and axioms
student \subseteq \forall right.useStudentPrinter

```
Action(printOnePageHP, [printerHP],  
(containsCartridge(printerHP), availablePaper(printerHP, X), X > 1),  
availablePaper(printerHP, X - 1))  
printOnePage(X) :- containsCartridge(X), availablePaper(X, Y), Y > 1.  
availablePaper(X, Y1) :- availablePaper(X, Y), printOnePage(X), Y1 is Y - 1.
```

OWL-DL constructors and axioms but difficult to express.
More “natural” in the other three approaches.

```
delegateSpeechAct(var1, var2,  
right(printlabPrinter,  
project(var1, SomeProject),  
project(var2, SomeProject)))  
right(X, Y) :- right(Z, delegationright), delegate(Z, X, Y) .  
right(X, printlabPrinter) :-  
delegate(_, X, printlabPrinter), project(X, S), project(Y, S) .
```

Non-tree rule. Cannot be expressed in OWL-DL. Other three approaches.



References

- Grosz, Horrocks, Volz, Decker. Description Logic Programs. Combining Logic Programs with Description Logics.
- Kagal, Finin, Joshi. A Policy Language for a pervasive Computing Environment.
- Franconi, Tessaris. Rules and Queries with Ontologies: A Unified Logical Framework.
- Pan, Franconi, Tessaris, Stamou, Tzouvaras, Serafini, Horrocks, Glimm. Specification of Coordination of Rule and Ontology Languages.
- Brachman, Levesque. Expressiveness and Tractability in Knowledge Representation and Reasoning.
- SWRL. A Semantic Web Rule Language. Combining OWL and RuleML.
- Donini, Lenzerini, Nardi. AL-Log: Integrating Datalog and Description Logics.
- Motik, Sattler, Studer. Query Answering for OWL-DL with Rules.
- Levy, Rousset. CARIN: A representation Language Combining Horn rules and Description Logics