Reasoning on Web Data Semantics

Oui. Peut-on préciser l'heure et le lieu ?

Marie-Christine Rousset

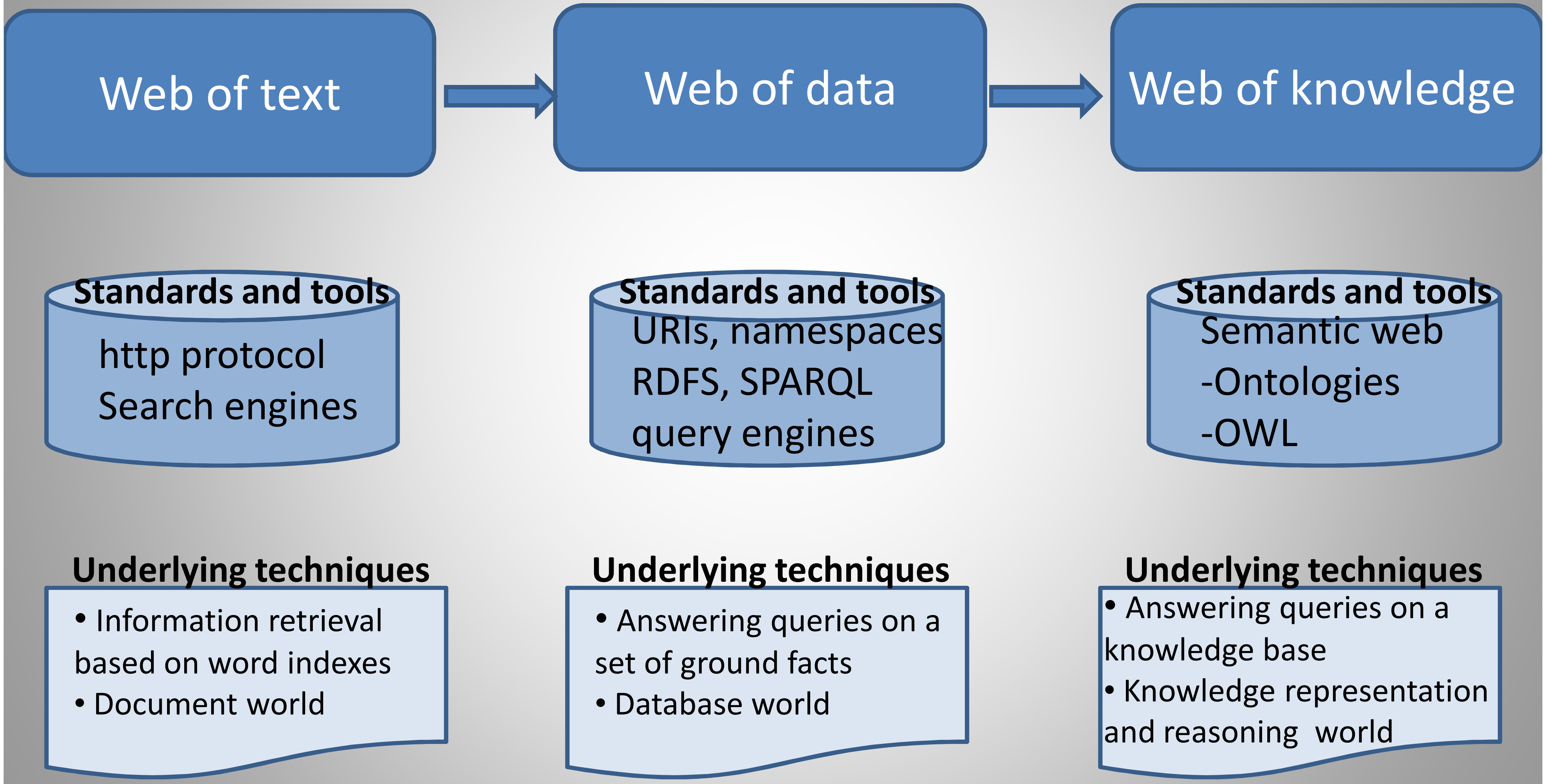
Merci

Université de Grenoble (UJF) et Institut Universitaire de France

Amicalement

Marie-Christine

Evolution of the Web



Web of text → Web of data → Web of knowledge

Standards and tools

http protocol

Search engines

Underlying techniques

- Information retrieval based on word indexes
- Document world

Standards and tools

URIs, namespaces

RDFS, SPARQL

query engines

Underlying techniques

- Answering queries on a set of ground facts
- Database world

Standards and tools

Semantic web

-Ontologies

-OWL

Underlying techniques

- Answering queries on a knowledge base
- Knowledge representation and reasoning world

Evolution of the Web

Current web

Web of text

Billions of web pages

Is emerging

Web of data

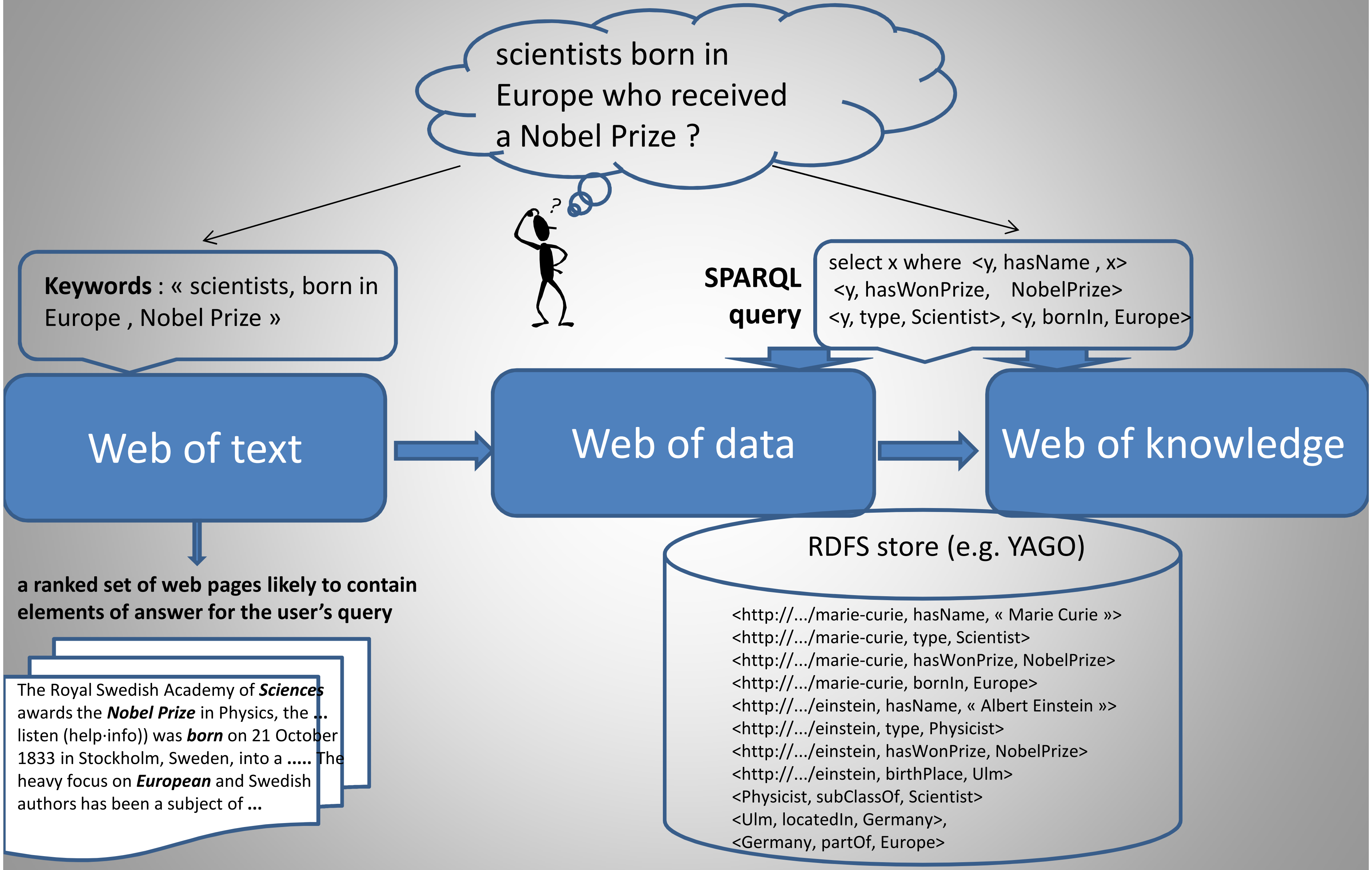
YAGO, DBPedia, Linked Data
Billions of RDF triples

Very next step

Web of knowledge

a flourishing of ontologies
and of algorithms for
ontology-based data access

Main differences illustrated by example



scientists born in
Europe who received
a Nobel Prize ?

Keywords : « scientists, born in Europe , Nobel Prize »

SPARQL query

select x where <y, hasName , x>
<y, hasWonPrize, NobelPrize>
<y, type, Scientist>, <y, bornIn, Europe>

Web of text

Web of data

Web of knowledge

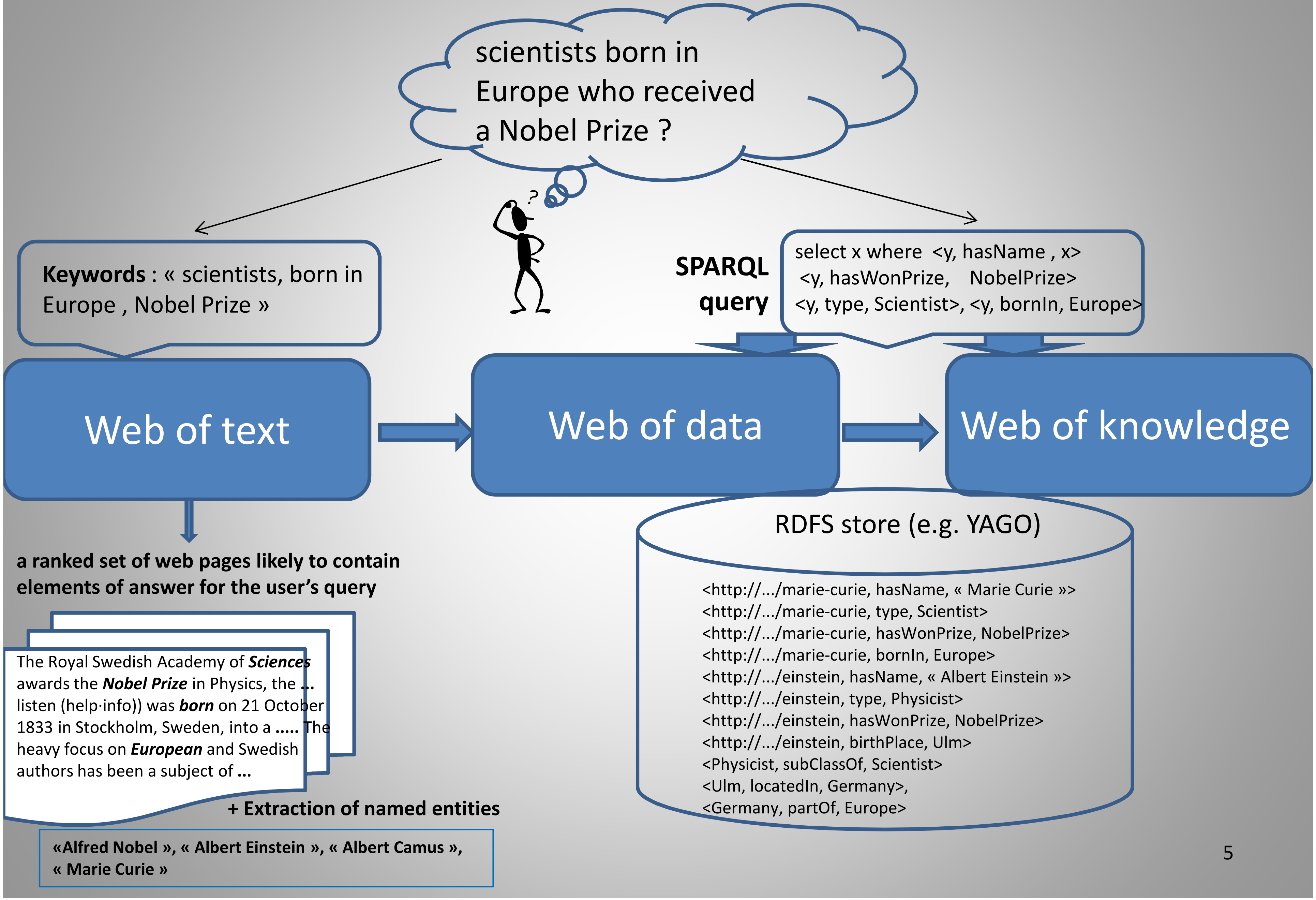
a ranked set of web pages likely to contain elements of answer for the user's query

The Royal Swedish Academy of ***Sciences*** awards the ***Nobel Prize*** in Physics, the **...** listen (help·info)) was ***born*** on 21 October 1833 in Stockholm, Sweden, into a **.....** The heavy focus on ***European*** and Swedish authors has been a subject of **...**

RDFS store (e.g. YAGO)

<http://.../marie-curie, hasName, « Marie Curie »>
<http://.../marie-curie, type, Scientist>
<http://.../marie-curie, hasWonPrize, NobelPrize>
<http://.../marie-curie, bornIn, Europe>
<http://.../einstein, hasName, « Albert Einstein »>
<http://.../einstein, type, Physicist>
<http://.../einstein, hasWonPrize, NobelPrize>
<http://.../einstein, birthPlace, Ulm>
<Physicist, subClassOf, Scientist>
<Ulm, locatedIn, Germany>,
<Germany, partOf, Europe>

Main differences illustrated by example



scientists born in Europe who received a Nobel Prize ?

Keywords : « scientists, born in Europe , Nobel Prize »

SPARQL query

select x where <y, hasName , x>
<y, hasWonPrize, NobelPrize>
<y, type, Scientist>, <y, bornIn, Europe>

Web of text

Web of data

Web of knowledge

a ranked set of web pages likely to contain elements of answer for the user's query

The Royal Swedish Academy of ***Sciences*** awards the ***Nobel Prize*** in Physics, the **...** listen (help·info)) was ***born*** on 21 October 1833 in Stockholm, Sweden, into a **.....** The heavy focus on ***European*** and Swedish authors has been a subject of **...**

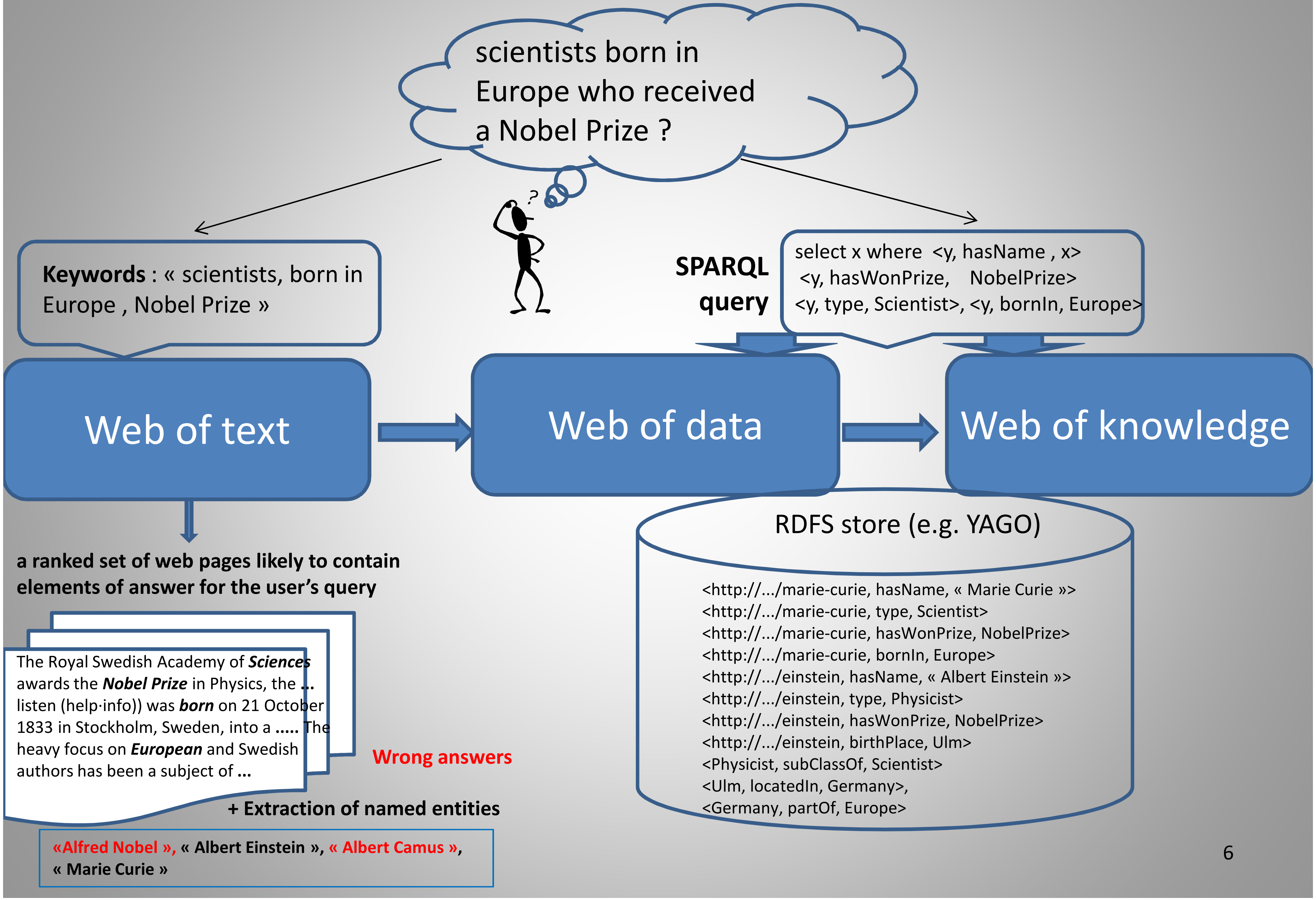
+ Extraction of named entities

«Alfred Nobel », « Albert Einstein », « Albert Camus », « Marie Curie »

RDFS store (e.g. YAGO)

<http://.../marie-curie, hasName, « Marie Curie »>
<http://.../marie-curie, type, Scientist>
<http://.../marie-curie, hasWonPrize, NobelPrize>
<http://.../marie-curie, bornIn, Europe>
<http://.../einstein, hasName, « Albert Einstein »>
<http://.../einstein, type, Physicist>
<http://.../einstein, hasWonPrize, NobelPrize>
<http://.../einstein, birthPlace, Ulm>
<Physicist, subClassOf, Scientist>
<Ulm, locatedIn, Germany>,
<Germany, partOf, Europe>

Main differences illustrated by example



scientists born in Europe who received a Nobel Prize ?

Keywords : « scientists, born in Europe , Nobel Prize »

SPARQL query

select x where <y, hasName , x>
<y, hasWonPrize, NobelPrize>
<y, type, Scientist>, <y, bornIn, Europe>

Web of text

Web of data

Web of knowledge

a ranked set of web pages likely to contain elements of answer for the user's query

The Royal Swedish Academy of ***Sciences*** awards the ***Nobel Prize*** in Physics, the **...** listen (help·info)) was ***born*** on 21 October 1833 in Stockholm, Sweden, into a **.....** The heavy focus on ***European*** and Swedish authors has been a subject of **...**

Wrong answers

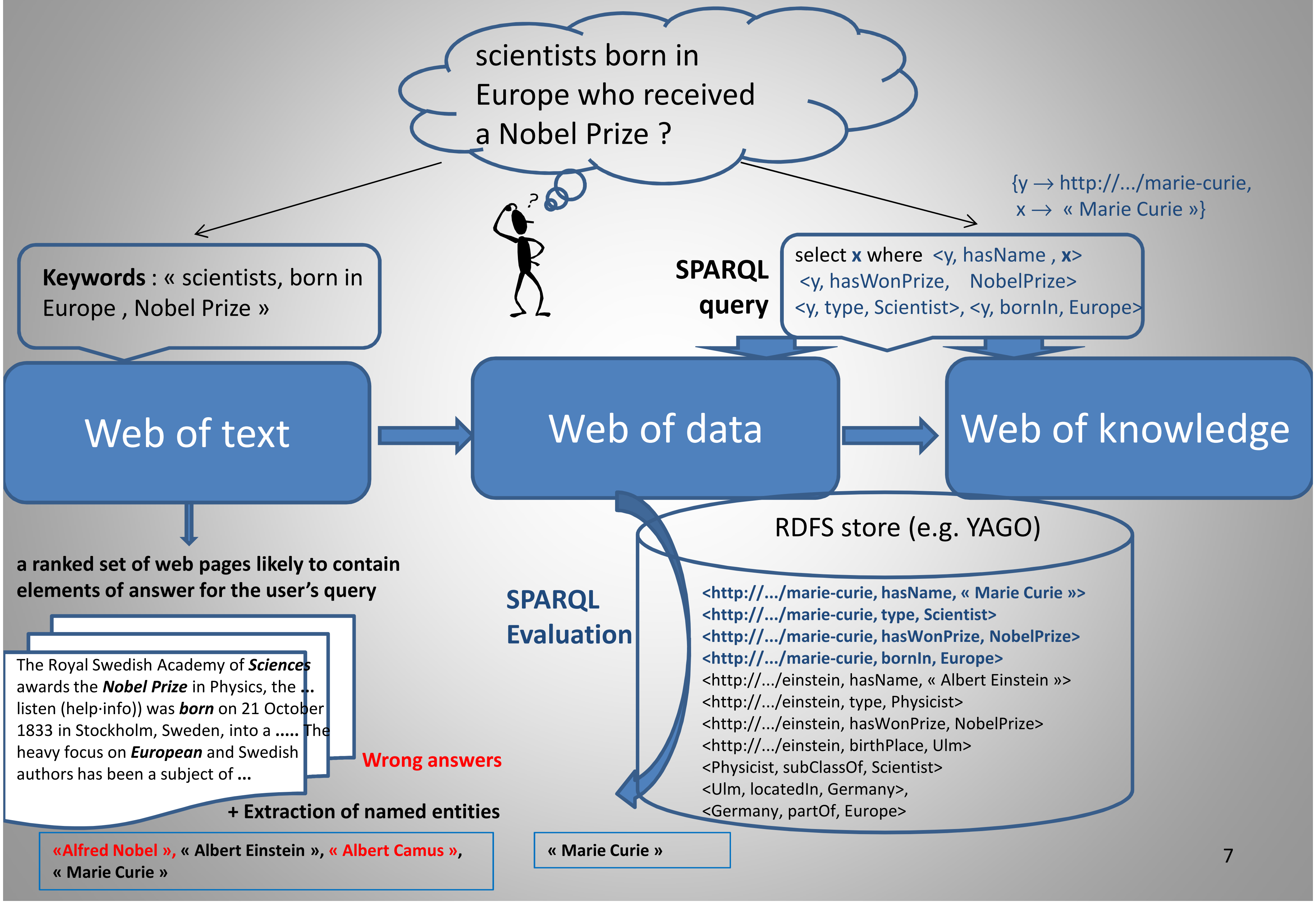
+ Extraction of named entities

«Alfred Nobel », « Albert Einstein », « Albert Camus », « Marie Curie »

RDFS store (e.g. YAGO)

<http://.../marie-curie, hasName, « Marie Curie »>
<http://.../marie-curie, type, Scientist>
<http://.../marie-curie, hasWonPrize, NobelPrize>
<http://.../marie-curie, bornIn, Europe>
<http://.../einstein, hasName, « Albert Einstein »>
<http://.../einstein, type, Physicist>
<http://.../einstein, hasWonPrize, NobelPrize>
<http://.../einstein, birthPlace, Ulm>
<Physicist, subClassOf, Scientist>
<Ulm, locatedIn, Germany>,
<Germany, partOf, Europe>

Main differences illustrated by example



scientists born in Europe who received a Nobel Prize ?

Keywords : « scientists, born in Europe , Nobel Prize »

SPARQL query

select **x** where <y, hasName , **x**>
<y, hasWonPrize, NobelPrize>
<y, type, Scientist>, <y, bornIn, Europe>

{y → http://.../marie-curie,
x → « Marie Curie »}

Web of text

Web of data

Web of knowledge

a ranked set of web pages likely to contain elements of answer for the user's query

The Royal Swedish Academy of ***Sciences*** awards the ***Nobel Prize*** in Physics, the **...** listen (help·info)) was ***born*** on 21 October 1833 in Stockholm, Sweden, into a **.....** The heavy focus on ***European*** and Swedish authors has been a subject of **...**

Wrong answers

+ Extraction of named entities

«Alfred Nobel », « Albert Einstein », « Albert Camus », « Marie Curie »

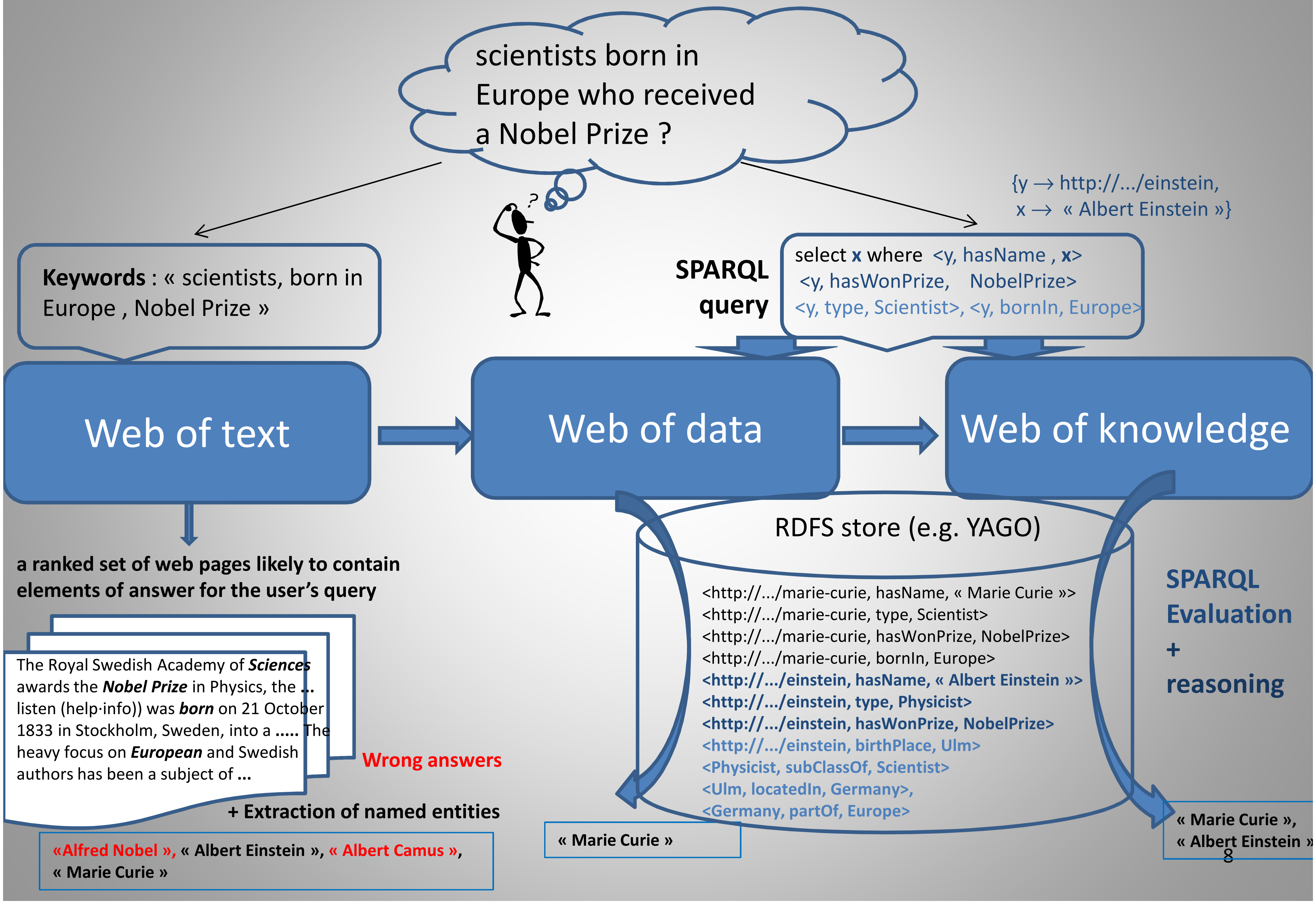
SPARQL Evaluation

RDFS store (e.g. YAGO)

<http://.../marie-curie, hasName, « Marie Curie »>
<http://.../marie-curie, type, Scientist>
<http://.../marie-curie, hasWonPrize, NobelPrize>
<http://.../marie-curie, bornIn, Europe>
<http://.../einstein, hasName, « Albert Einstein »>
<http://.../einstein, type, Physicist>
<http://.../einstein, hasWonPrize, NobelPrize>
<http://.../einstein, birthPlace, Ulm>
<Physicist, subClassOf, Scientist>
<Ulm, locatedIn, Germany>,
<Germany, partOf, Europe>

« Marie Curie »

Main differences illustrated by example



scientists born in Europe who received a Nobel Prize ?

Keywords : « scientists, born in Europe , Nobel Prize »

SPARQL query

select **x** where <y, hasName , **x**>
<y, hasWonPrize, NobelPrize>
<y, type, Scientist>, <y, bornIn, Europe>

{y → http://.../einstein,
x → « Albert Einstein »}

Web of text

Web of data

Web of knowledge

a ranked set of web pages likely to contain elements of answer for the user's query

The Royal Swedish Academy of ***Sciences*** awards the ***Nobel Prize*** in Physics, the **...** listen (help·info)) was ***born*** on 21 October 1833 in Stockholm, Sweden, into a **.....** The heavy focus on ***European*** and Swedish authors has been a subject of **...**

Wrong answers

+ Extraction of named entities

«Alfred Nobel », « Albert Einstein », « Albert Camus », « Marie Curie »

RDFS store (e.g. YAGO)

<http://.../marie-curie, hasName, « Marie Curie »>
<http://.../marie-curie, type, Scientist>
<http://.../marie-curie, hasWonPrize, NobelPrize>
<http://.../marie-curie, bornIn, Europe>
<http://.../einstein, hasName, « Albert Einstein »>
<http://.../einstein, type, Physicist>
<http://.../einstein, hasWonPrize, NobelPrize>
<http://.../einstein, birthPlace, Ulm>
<Physicist, subClassOf, Scientist>
<Ulm, locatedIn, Germany>,
<Germany, partOf, Europe>

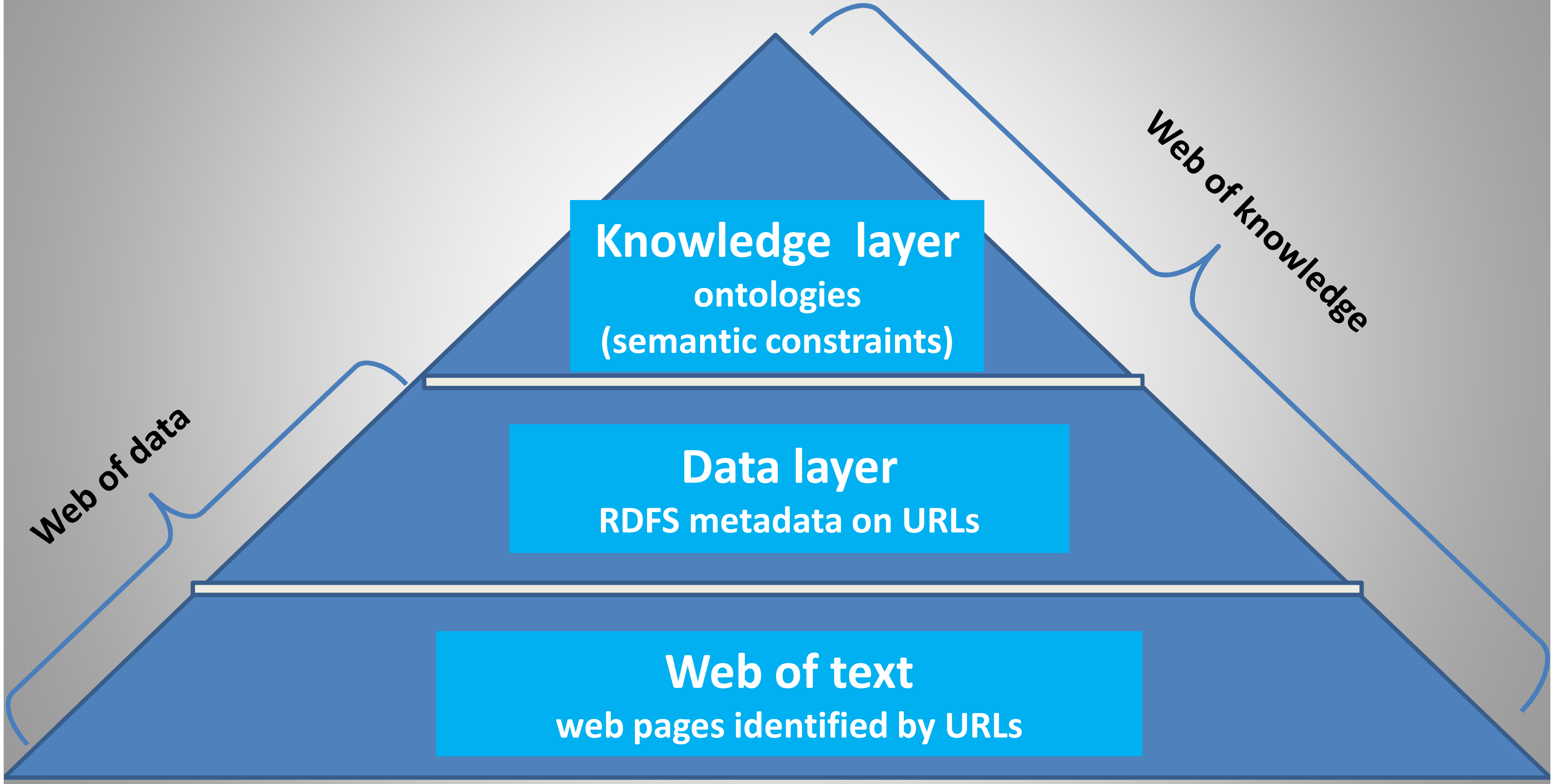
« Marie Curie »

SPARQL Evaluation + reasoning

« Marie Curie »,
« Albert Einstein »

The real picture

The web is evolving from a web of text to a web of knowledge in a coherent and a smooth way



Knowledge layer
ontologies
(semantic constraints)

Data layer
RDFS metadata on URLs

Web of text
web pages identified by URLs

Lesson learnt from the example

- Answering queries over the web of knowledge requires **reasoning**
  - Ontological statements can be used to infer new facts and deduce answers that could not be obtained otherwise
  - They are constraints used as deductive rules that infer new facts
  - Subtlety: some inferred facts can be partially known

From the constraint "a professor teaches at least one master course"

$$\forall x\ (\mathrm{Professor}(x) \Rightarrow \exists\ y\ \mathrm{Teaches}(x,y), \mathrm{MasterCourse}(y))$$

and the fact:

Professor(dupond) (RDF syntax: **<dupond, type, Professor>**)

it can be inferred the two following incomplete "facts" :

Teaches(dupond, v) , MasterCourse(v)

i.e, in RDF notation, two RDF triples with blank nodes:

<dupond, Teaches, _v> , <_v, type, MasterCourse>

Finding inconsistent information on the Web

- **Reasoning**: a tool for checking consistency
  - Some ontological statements can be used as **integrity** constraints

    “a professor cannot be a lecturer” ; “a course must have a responsible”

    $\forall x$ (Professor(x) => ¬ Lecturer(x))

    $\forall x$ (Course(x) => $\exists$ y ResponsibleFor(y,x))

    “a master course is taught by a single teacher”

    “only professors can be responsible of the courses they have to teach”

    $\forall x\ \forall y$ (Course(x), ResponsibleFor(y,x) => Professor(y), Teaches(y,x))
  - Subtlety: showing data inconsistency may require **intricate reasoning** on different rules, constraints and facts

    The facts: **Lecturer (jim), Teaches(jim, ue431) , MasterCourse(ue431)**

    \+ the above integrity constraints

    \+ the rule **$\forall x$ (MasterCourse(x) => Course(x))** leads to an inconsistency

Automatic Reasoning

- Not a novel problem
  - Many decidability and complexity results coming from decades of research in the KR&R community
  - Several inference algorithms and implemented reasoners
- The key point
  - first-order-logic is appropriate for knowledge representation
  - but **full first-order-logic is not decidable**

    no general algorithm that, applied to two any FOL formula, determines whether the first one implies the second one

$\Rightarrow$ the game is to find restrictions to design:

  - decidable fragments of first-order-logic
  - expressive enough for modeling useful knowledge or constraints

Description Logics

- A family of class-based logical languages for which reasoning is decidable
  - Provides algorithms for reasoning on (possibly complex) logical constraints over unary and binary predicates
- This is exactly what is needed for handling ontologies
  - in fact, the OWL constructs come from Description Logics
- A fine-grained analysis of computational complexity with surprising complexity results
  - $\mathcal{ALC}$ is EXPTIME–complete

=>any sound and complete inference algorithm for reasoning on most of the subsets of constraints expressible in OWL may take an exponential time (in the worst-case)

“only professors or lecturers may teach to undergraduate students”

$\forall$x $\forall$y (TeachesTo(x,y), UndergraduateStudent(y) => Professor(x) $\vee$ Lecturer(x))

$$\exists TeachesTo.UndergraduateStudent \sqsubseteq Professor \sqcup Lecturer$$

The same game again…

- Find restrictions on the logical constructs and/or the allowed axioms in order to:
  - design sublanguages for which reasoning is in P

    EL, **DL-Lite**
  - expressive enough for modeling useful constraints over data
- **DL-Lite: a good trade-off**
  - captures the main constraints used in databases and in software engineering
  - extends **RDFS** (the formal basis of OWL2 QL profile)
  - specially designed for answering queries over ontologies to be **FOL-reducible**

FOL-reducibility

Query answering and data consistency checking can be performed in two separate steps:

- a **query reformulation step**
  - reasoning on the ontology (and the queries)
  - independent of the data

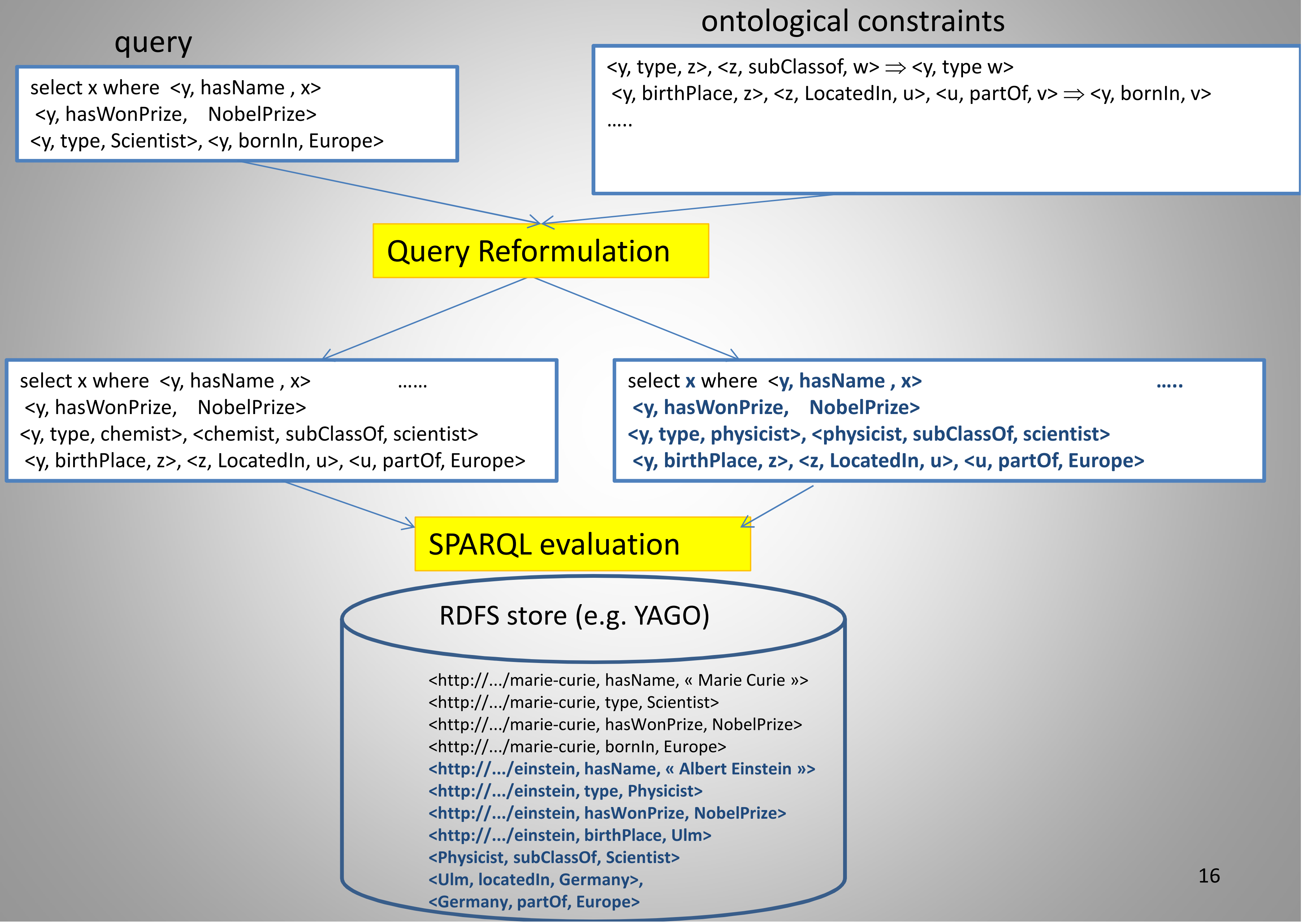
⇒ a set a queries: the reformulations of the input query

- an **evaluation step**
  - of the (SPARQL) query reformulations on the (RDF) data
  - independent of the ontology

⇒ Main advantage

  - makes possible to use an SQL or SPARQL engine
  - thus taking advantage of well-established query optimization strategies supported by standard relational DBMS

Illustration



query

select x where <y, hasName , x>
<y, hasWonPrize, NobelPrize>
<y, type, Scientist>, <y, bornIn, Europe>

ontological constraints

<y, type, z>, <z, subClassof, w> ⇒ <y, type w>
<y, birthPlace, z>, <z, LocatedIn, u>, <u, partOf, v> ⇒ <y, bornIn, v>
…..

Query Reformulation

select x where <y, hasName , x> ……
<y, hasWonPrize, NobelPrize>
<y, type, chemist>, <chemist, subClassOf, scientist>
<y, birthPlace, z>, <z, LocatedIn, u>, <u, partOf, Europe>

select **x** where **<y, hasName , x>** **…..**
<y, hasWonPrize, NobelPrize>
<y, type, physicist>, <physicist, subClassOf, scientist>
<y, birthPlace, z>, <z, LocatedIn, u>, <u, partOf, Europe>

SPARQL evaluation

RDFS store (e.g. YAGO)

<http://…/marie-curie, hasName, « Marie Curie »>
<http://…/marie-curie, type, Scientist>
<http://…/marie-curie, hasWonPrize, NobelPrize>
<http://…/marie-curie, bornIn, Europe>
<http://…/einstein, hasName, « Albert Einstein »>
<http://…/einstein, type, Physicist>
<http://…/einstein, hasWonPrize, NobelPrize>
<http://…/einstein, birthPlace, Ulm>
<Physicist, subClassOf, Scientist>
<Ulm, locatedIn, Germany>,
<Germany, partOf, Europe>

DL-Lite by example

Professor $\sqsubseteq \exists$ Teaches

$$\forall x\,(\mathrm{Professor}(x) \Rightarrow \exists y\,\mathrm{Teaches}(x,y))$$

$\exists$ Teaches^{-} $\sqsubseteq$ Course

$$\forall x \forall y\,(\,\mathrm{Teaches}(x,y) \;\; \Rightarrow \mathrm{Course}(y))$$

ResponsibleFor $\sqsubseteq$ Teaches

$$\forall x \forall y\,(\,\mathrm{ResponsibleFor}(x,y) \Rightarrow \mathrm{Teaches}(x,y))$$

(funct ResponsableFor^{-})

$$\forall x \forall y \forall z (\mathrm{ResponsibleFor}(y,x) \wedge \mathrm{ResponsibleFor}(z,x) \Rightarrow y=z)$$

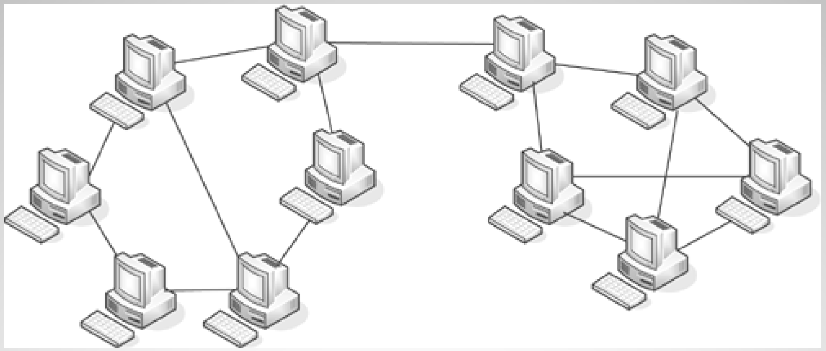
Lecturer $\sqsubseteq \neg$ ($\exists$ResponsibleFor)

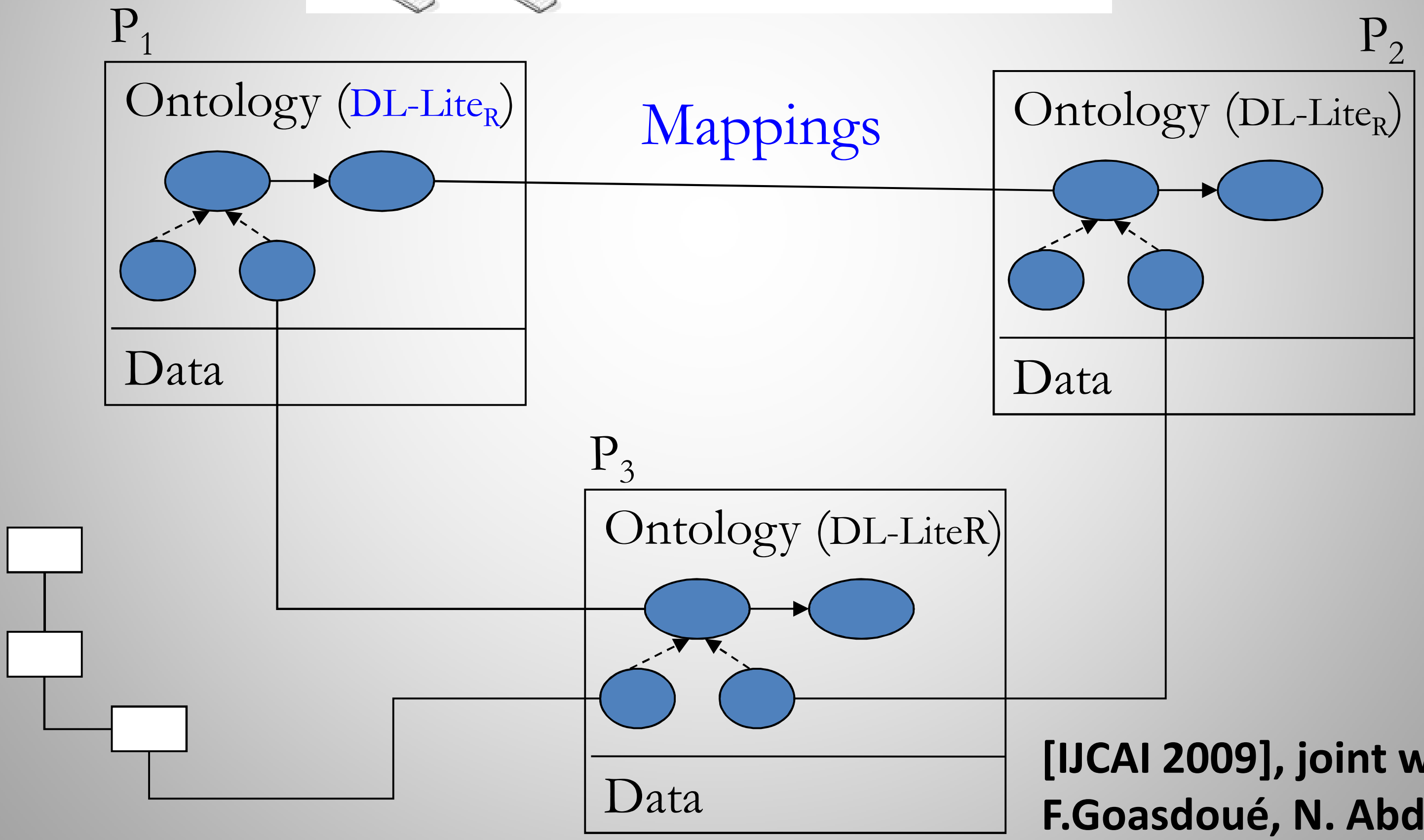
$$\forall x\ \forall y\,(\mathrm{Lecturer}(x) \wedge \mathrm{ResponsibleFor}(x,y) \Rightarrow \bot)$$

DL-Lite: a frontier for FOL reducibility

- The reasoning step is polynomial in the size of the ontology
- The evaluation step has the same data complexity as standard evaluation of conjunctive queries over relational databases
  - in AC0 (strictly contained in LogSpace and thus in P)
- The interaction between relation inclusion constraints and functionality constraints makes reasoning in DL-Lite P-complete in data complexity
  - **DL-Lite$_A$ is FOL-reducible**
  - **full DL-Lite is not FOL-reducible**
    - reformulating a query may require recursion (Datalog)

Decentralized ontology-based data access





P_1

Ontology (DL-Lite$_R$)

Data

Mappings

P_2

Ontology (DL-Lite$_R$)

Data

P_3

Ontology (DL-LiteR)

Data

[IJCAI 2009], joint work with F.Goasdoué, N. Abdallah

Conclusion

- The scalability of reasoning on Web data requires light-weight ontologies
- RDFS is not expressive enough to express useful constraints
- Forget about (most of fragments) of OWL

⇒extend RDFS with constraints expressible in a logic for which data management is FOL reducible

  - **DL-Lite$_A$** is an example of such a logic
  - (some fragments of) **Datalog^{+-}** too

References

Web data management,

Abiteboul, Manolescu, Rigaux, Rousset, Senellart

webdam.inria.fr/Jorge

Decentralized reasoning in DL-Lite

Abdallah, Goasdoué, Rousset

IJCAI 2009

Reasoning for reference reconciliation

Sais, Pernelle, Rousset

IJCAI 2007

Trust for resource finding in semantic P2P networks

Atencia, Euzenat, Pirro, Rousset

ISWC 2011

MERCI

Université
Joseph Fourier
GRENOBLE

UNIVERSITÉ DE GRENOBLE

IUF