Plan

1. RDF : Resource Description Framework
2. SPARQL : RDF Query Language

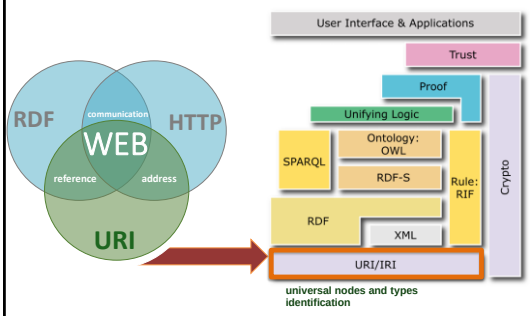
Matériel

1. Support de cours
   - http://wimmics.inria.fr/lectures
2. Logiciel Corese :
   - http://wimmics.inria.fr/doc/tutorial/corese-3.2.2.jar
3. Document RDF
   - http://wimmics.inria.fr/doc/tutorial/human.ttl
4. Documents SPARQL
   - http://wimmics.inria.fr/doc/tutorial/sparql-query.txt
   - http://wimmics.inria.fr/doc/tutorial/td-sparql.pdf

Plan

1. **RDF : Resource Description Framework**
2. SPARQL : RDF Query Language

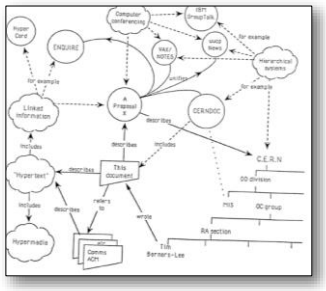
Web sémantique



RDF

- Resource Description Framework
- Formalisme de représentation et d'échange de données et de connaissances sur le Web
- Recommandation W3C

RDF: Graphe orienté étiqueté



Exemple : DBpedia

- Base de connaissances RDF
- Extraction automatique depuis Wikipedia
- Ressources décrites avec des URI

http://fr.dbpedia.org/resource/Université_Nice-Sophia-Antipolis
dbpedia-owl:city
http://fr.dbpedia.org/resource/Nice

Triplet RDF

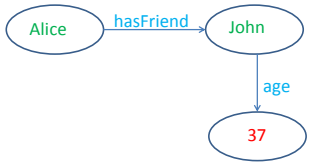
resource property value

subject predicate object

Triplet RDF

- Alice, à pour ami, John
- John, à pour age, 37
- sujet, prédicat, objet

Graphe étiqueté orienté



Etiquettes

1. URI : http://www.inria.fr/human/data#Alice h:age
2. Blank Node :
   - _:b23
   - [ ]
3. Literal :
   - "Alice"
   - 37
   - 3.14
   - true
   - "1930-01-29"^^xsd:date

Turtle Syntax

```
@prefix h: <http://www.inria.fr/human#> .
@prefix i: <http://www.inria.fr/human/data#> .

i:Alice h:hasFriend i:John .
i:John h:age 37.
```

13

RDF/XML Syntax

```
<rdf:RDF
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:h="http://www.inria.fr/human#">

<rdf:Description rdf:about="http://www.inria.fr/human/data#Alice">
  <h:hasFriend rdf:resource="http://www.inria.fr/human/data#John"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inria.fr/human/data#John">
 <h:age rdf:datatype="http://www.w3.org/2001/XMLSchema#int">37</h:age>
</rdf:Description>

</rdf:RDF>
```

14

Triplet RDF

Sujet : URI, Blank Node, ~~Literal~~
Propriété : URI, ~~Blank Node, Literal~~
Valeur : URI, Blank Node, Literal

15

Literal

Typés avec XML Schema Datatype

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
```

- rdf:langString (literal avec language tag)
- xsd:string
- xsd:integer, xsd:double, xsd:decimal
- xsd:boolean
- xsd:date
- etc.

16

Literal

```
i:John h:age  "37"^^xsd:integer .

i:John h:age  37 .
```

17

Blank Node

```
i:John h:own [ a h:Car ; h:color "Blue" ] .
```

18

Blank Node

```
i:John h:own [ a h:Car ; h:color "Blue" ] .

i:John h:own _:b1 .
_:b1 a h:Car ; h:color "Blue"  .
```

19

RDF namespace

```
@prefix rdf:
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
```

20

Typer les ressources

```
i:Alice rdf:type h:Woman .

i:Alice a h:Woman .
```

21

Multi instanciation

```
i:Alice a h:Woman , h:Researcher .
```

22

Liste

```
i:book  h:author  ("Catherine" "Fabien" "Olivier")
```

23

Liste

```
i:book  h:author  ("Catherine" "Fabien" "Olivier")

i:book h:author [
rdf:first "Catherine"      ; rdf:rest [
rdf:first "Fabien"         ; rdf:rest [
rdf:first "Olivier"        ; rdf:rest rdf:nil ]]]
```

24

Graphe par défaut

```
i:James a h:Lecturer ;
        h:name "James" .

i:James a h:Musician ;
        h:name "Jimmy" .
```

25

Graphes nommés

```
graph i:g1  {
      i:James a h:Lecturer ;
              h:name "James" .
}

graph i:g2  {
      i:James a h:Musician ;
              h:name "Jimmy" .
}
```

26

Graphes nommés

- Contextualiser les données
- Annoter, typer les URI des graphes nommés
- Versioning, Annotation temporelle
- Distinguer ontologie et données

27

RDFS : RDF Schema

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

- Hiérarchie de classes

```
h:Woman a rdfs:Class ;
   rdfs:subClassOf  h:Person, h:Female .

h:Man a rdfs:Class ;
   rdfs:subClassOf  h:Person, h:Male .
```

28

RDFS : RDF Schema

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

- Signature de propriétés

```
h:name a rdf:Property ;
  rdfs:domain h:Person ;
  rdfs:range xsd:string .
```

29

Exercice

Modélisez des énoncés en RDF, par exemple :

- Des personnes avec des relations et des liens de parenté

@prefix ex: <http://example.org/ns/> .

```
ex:John a ex:Person ;
  ex:name "John" ;
  ex:knows ex:Jim .
```

30

Exercice

Télécharger Corese :
http://wimmics.inria.fr/doc/tutorial/corese-3.2.2.jar

Valider votre document RDF en le chargeant dans Corese:

- File/Load RDF

Plan

1. RDF : Resource Description Framework
2. **SPARQL : RDF Query Language**

Matériel

1. Support de cours
   - http://wimmics.inria.fr/lectures
2. Logiciel Corese :
   - http://wimmics.inria.fr/doc/tutorial/corese-3.2.2.jar
3. Document RDF
   - http://wimmics.inria.fr/doc/tutorial/human.ttl
4. Documents SPARQL
   - http://wimmics.inria.fr/doc/tutorial/sparql-query.txt
   - http://wimmics.inria.fr/doc/tutorial/td-sparql.pdf

Triple Pattern

- Turtle triple syntax
- Variables

```
?x  a  h:Person

?x  h:name "John"

<http://www.inria.fr/human/data#John>  h:name  ?name

<http://www.inria.fr/human/data#John>  ?p  ?v
```


Select Query

```
prefix h: <http://www.inria.fr/human#> .
select *
where {
        ?x a h:Person ;
          h:name "David", ?name .
        ?x h:hasFriend ?y
}
```


Blank Node

- *Anonymous* variable
- Value of Blank Node is not returned in result

```
select *
where {
        [ ] a h:Person ;
          h:name ?name
}
```

Filter

```
prefix h: <http://www.inria.fr/human#> .
select * where {
  ?x h:age ?age
  filter (?age >= 18)
}
```


Filter Language

• URI, Literal, Variable	i:John, 3.14, ?x
• < <= >= >	?age >= 18, ?n < "John"
• = !=	?x != ?y, ?y = "James"
• ( )	
• + - * /	(?n * (?n + 1))/2
• && || !	! (?x < 0 && ?y < 0)
• Function	datatype(?x)

Conditional Statement

- if then else

```
filter if (lang(?name) = "fr",
           ?age >= 18,
           ?age >= 21)
```


Function

- isBlank(?x)
- isURI(?x)
- isLiteral(?x)
- bound(?x)

Function

- datatype(?x)
- lang(?l)
- langMatches(lang(?l), "fr")
- str(<http://example.org>)
- uri("http://example.org")
- xsd:integer("123")
- xsd:string(12)
- strdt(str, datatype)
- strlang(str, lang)

Function

- contains(str_1, str_2)
- strstarts(str_1, str_2)
- strends(str_1, str_2)
- concat(str_1, str_2)
- substr(str, n)
- strlen(str)
- regex(str, ".*cnrs")
- …

Query Form

1. Select
2. Ask
3. Construct
4. Describe

Select

- Return variable bindings

```
prefix h: <http://www.inria.fr/human#> .
select * where {
        ?x h:hasFriend ?y
}
```


Ask

- Return true/false

```
prefix h: <http://www.inria.fr/human#> .
ask { ?x  rdf:type  h:Man }
```


Construct

- Return new RDF graph

```
prefix h: <http://www.inria.fr/human#> .
construct {?x rdfs:seeAlso?z}
where { ?x h:hasFriend ?y . ?y h:hasFriend ?z }
```


Describe

- Return description of resource(s) as RDF graph

```
describe <http://www.inria.fr/human/data#Alice>

prefix h: <http://www.inria.fr/human#> .
describe ?x
where {  ?x h:name "Alice"  }
```


Statement

```
1.  {?x a h:Man} union {?x a h:Woman}
2.  {?x a h:Man} optional {?x h:name ?n}
3.  {?x a h:Woman} minus {?x h:hasFriend [ h:name "John"]}
4.  filter exists {?x h:name ?n}
5.  filter not exists {?x h:name ?n}
6.  select from ex:g1 from named ex:g2
7.  graph ?g {?x a h:Man}   -- graph ex:g1 {?x a h:Man}
8.  Property Path   ?x rdf:type/rdfs:subClassOf* h:Woman
9.  select * where { {select * where {}} }
10. bind (2 * ?age  as  ?twice)
11. values  ?v  { "John"  "Alice" }
12. service <http://fr.dbpedia.org/sparql> { ?x rdfs:label "Auguste"@fr}
```

Property Path Language

?x exp ?y

exp ::=

uri :	property
!uri :	negation
exp/exp :	sequence
exp|exp :	alternative
exp? :	optional
exp* :	zero or several
exp+ :	one or several
^exp :	reverse

?list rdf:rest*/rdf:first ?elem

Result & Modifier

1. select distinct ?x ?y
2. select (datatype(?x) as ?d)
3. count, sum, avg, min, max, group_concat, sample
4. group by ?author
5. having (count(?x) > 10)
6. order by ?date
7. limit 10
8. offset 20

SPARQL Update

Manage (modify) content of RDF Dataset

Load RDF document

```
load <http://wimmics.inria.fr/doc/tutorial/human.ttl>

load <http://wimmics.inria.fr/doc/tutorial/human.ttl>
into graph h:g1
```


Insert Data

```
prefix h: <http://www.inria.fr/human#> .
prefix i: <http://www.inria.fr/human/data#> .
insert data {
 i:Jim a h:Man ; h:age 18 .
 i:Jill a h:Woman ; h:age 81 .
}
```


Delete Data

```
prefix h: <http://www.inria.fr/human#> .
prefix i: <http://www.inria.fr/human/data#> .
delete data {
 i:Jim a h:Man ; h:age 18 .
 i:Jill a h:Woman ; h:age 81 .
}
```

Delete Where

```
prefix h: <http://www.inria.fr/human#> .
delete {
  ?x h:age ?age
}
where {
  ?x a h:Man ; h:age ?age
  filter (?age > 70)
}
```


Insert Where

```
prefix h: <http://www.inria.fr/human#> .
insert {
  ?x h:mail ?mail
}
where {
  ?x h:name ?n
  bind (concat (?n, "@acme.com") as ?mail)
}
```


Update: Delete Insert Where

```
prefix h: <http://www.inria.fr/human#> .
delete { ?x h:age ?age }
insert  { ?x h:age ?new }
where {
  ?x h:age ?age
  filter (?age >= 57)
  bind (?age / 2  as  ?new)
}
```


Matériel

1. Support de cours
   - http://wimmics.inria.fr/lectures
2. Logiciel Corese :
   - http://wimmics.inria.fr/doc/tutorial/corese-3.2.2.jar
3. Document RDF
   - http://wimmics.inria.fr/doc/tutorial/human.ttl
4. Documents SPARQL
   - http://wimmics.inria.fr/doc/tutorial/sparql-query.txt
   - http://wimmics.inria.fr/doc/tutorial/td-sparql.pdf