SPARQL Template Transformation Language

-

Un langage de transformation de graphe RDF

olivier.corby@inria.fr

Wimmics, Inria, UCA, I3S, CNRS

http://team.inria.fr/wimmics

STTL

STTL : transformation language for RDF

XSLT : transformation language for XML

- Input RDF graph
- Output Text format
- SPARQL based
- Declarative transformation rules

XSLT - STTL

```
<xsl:template  match="person">
        <xsl:apply-templates select="knows"/>
</xsl:template>
```

```
template { st:apply-templates(?y) }
where { ?in a foaf:Person ; foaf:knows ?y }
```

XSLT -STTL

	XSLT	STTL
Input	XML	RDF
Output	XML	Text
Syntax	XML	SPARQL extension
Template	xsl:template	template where
Named Template	xsl:template name="test"	template ex:test
Apply templates	xsl:apply-templates	st:apply-templates
Apply named template	xsl:call-template	st:call-template
Parameters	xsl:with-param	(?x, ?y)
Numbering	xsl:number	st:number
Sorting	xsl:sort	order by
Grouping	xsl:for-each-group	group by
Condition	xsl:if	if (exp, then, else)

STTL motivating use cases

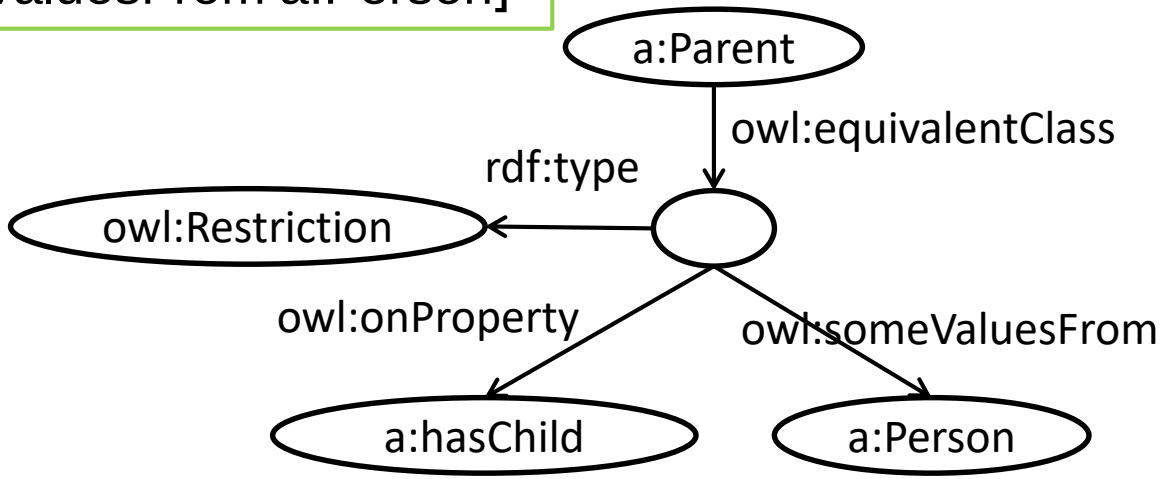
1. Transformation of RDF data from one RDF syntax to another:
   - Turtle
   - RDF/XML
   - JSON LD
2. Presentation of RDF data:
   - RDF to HTML
   - RDF to Latex
   - RDF to Natural Language
   - RDF to graphic format (GML)
3. Transformation of statements in a given language from RDF to another syntax:
   - OWL/RDF to OWL functional syntax
   - SPARQL/RDF (SPIN) to SPARQL syntax
   - AST of *L* in RDF to concrete syntax of *L*
4. Constraint checking
   - OWL Profile: OWL RL
   - SHACL

Example use case: OWL/RDF to OWL/FS

```
a:Parent owl:equivalentClass [ a owl:Restriction ;
                               owl:onProperty a:hasChild;
                               owl:someValuesFrom a:Person]
```

OWL/RDF (Turtle)

STTL transformation




```
EquivalentClasses (a:Parent ObjectSomeValuesFrom(a:hasChild, a:Person) )
```

OWL/Functional Syntax

SPARQL

Query forms

```
SELECT WHERE { GP }
CONSTRUCT { GP } WHERE { GP }
ASK { GP }
DESCRIBE  WHERE { GP }
```

SPARQL Template

Query forms

```
SELECT WHERE { GP }
CONSTRUCT { GP } WHERE { GP }
ASK { GP }
DESCRIBE  WHERE { GP }
```

```
TEMPLATE { Text Pattern } WHERE { GP }
```

SPARQL Template

An additional SPARQL query form:

```
TEMPLATE { Text Pattern } WHERE { GP }
```

with Text Pattern = (VARIABLE | EXP | TEXT)*

RDF to Turtle transformation

```
TEMPLATE { ?x  " "  rdfs:label  " "  ?name  "." }
WHERE { ?x a foaf:Person ; foaf:name ?name }
```

```
ns:olivier a foaf:Person ; foaf:name "Olivier".
ns:catherine a foaf:Person ; foaf:name "Catherine".
```

```
ns:olivier rdfs:label "Olivier".
ns:catherine rdfs:label "Catherine".
```

RDF to HTML transformation

```
TEMPLATE { format {"<a href='%s'>%s</a>"  str(?x) str(?name) } }
WHERE { ?x a foaf:Person ; foaf:name ?name }
```

```
ns:olivier a foaf:Person ; foaf:name "Olivier".
ns:catherine a foaf:Person ; foaf:name "Catherine".
```

```
<a href='http://ns.inria.fr/olivier'>Olivier</a>
<a href='http://ns.inria.fr/catherine'>Catherine</a>
```

STTL: Transformation

A set of templates

```
TEMPLATE { "EquivalentClasses (" ?in " " ?c ")" }
WHERE { ?in owl:equivalentClass ?c }

TEMPLATE { "SubClassOf (" ?in " " ?c ")" }
WHERE { ?in rdfs:subClassOf ?c }

TEMPLATE { "ObjectSomeValuesFrom (" ?p " " ?c ")" }
WHERE { ?in a owl:Restriction ;
        owl:onProperty ?p ;
        owl:someValuesFrom ?c }
```

Template recursive call

```
TEMPLATE { "EquivalentClasses ("
    ?in " " ?c ")" }
WHERE { ?in owl:equivalentClass ?c . }
```

Template recursive call

```
TEMPLATE { "EquivalentClasses ("
    ?in " " ?c ")" }
WHERE { ?in owl:equivalentClass ?c . }
```

Template recursive call

```
TEMPLATE { "EquivalentClasses ("
    st:apply-templates(?in) " " ?c ")" }
WHERE { ?in owl:equivalentClass ?c . }
```

Template recursive call

```
TEMPLATE { "EquivalentClasses ("
    st:apply-templates(?in) " " st:apply-templates(?c) ")" }
WHERE { ?in owl:equivalentClass ?c . }
```

STTL

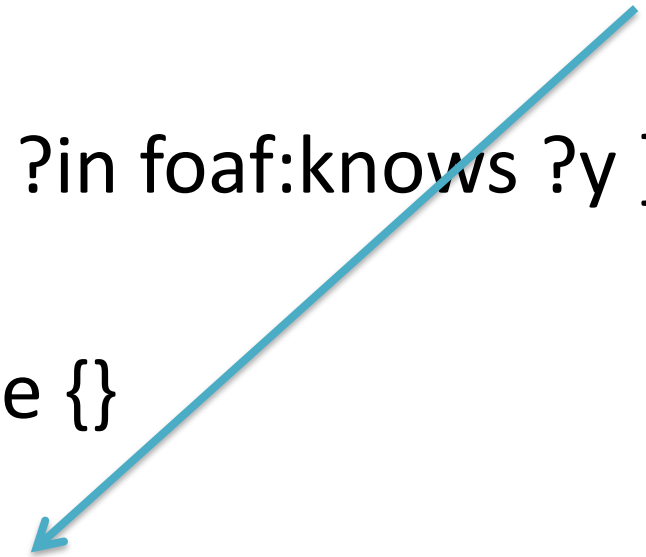
1. SPARQL Template Query form
2. Transformation: a set of templates
3. Extension functions: st:apply-templates, st:call-template

Focus Node

```
template {
        st:apply-templates(?y)
}
where { ?in foaf:knows ?y }
```

Focus Node ?in

```
template {
        st:apply-templates(?y)
}
where { ?in foaf:knows ?y }

template {}
where {
        ?in a foaf:Person
}
```

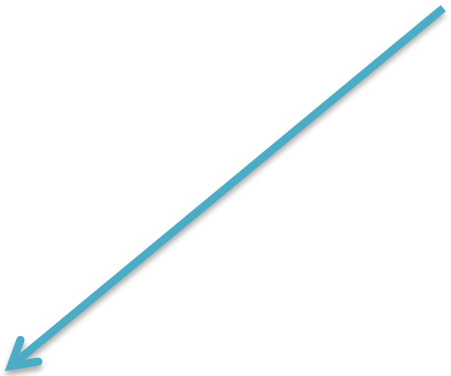


Named Template

```
template {
        st:call-template(st:title)
}
where {}
```

Named Template

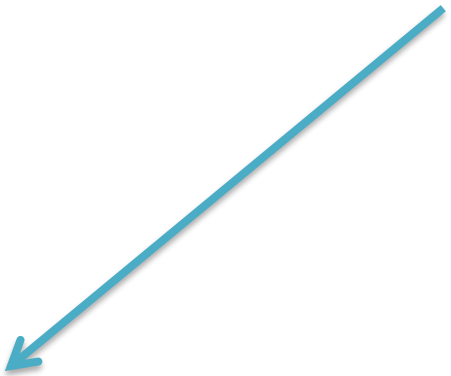
```
template {
        st:call-template(st:title)
}
where {}
```



```
template st:title {}
where {}
```

Named Template

```
template {
        st:call-template(st:title, ?y)
}
where {}
```



```
template st:title (?x) {}
where {}
```

STTL Features

STTL Extension Functions

prefix st: <http://ns.inria.fr/sparql-template/>

st:apply-templates(term)
st:apply-templates-with(transform-uri, term)

st:call-template(template-uri, term)
st:call-template-with(transform-uri, template-uri, term)

st:turtle(term)

Start template

```
template st:start {
        st:apply-templates(?x)
}
where {
        ?x a foaf:Person
}
```

Profile template: Declare Functions

```
template st:profile {}
where {}

function st:display(?x) {
   if (isBlank(?x),
         concat("bnode: " , ?x),
         st:turtle(?x))
}
```

Profile template: Import Functions

```
@import <function/definition.rq>

template st:profile {}
where {}
```

Variable Processing

```
template { ?y }
where { ?in ?p ?y }
```

Compiled into:

```
template { st:process(?y) }
where { ?in ?p ?y }
```

st:process(?y) = st:turtle(?y)

Overloading Variable Processing

```
function st:process(?x) {
   if (isBlank(?x),
         st:apply-templates(?x),
         st:turtle(?x))
}
```

Priority

template { }
where { }
pragma { st:template st:priority 5 }

default priority = 100

Template Statements

- Separator
- Format
- Group
- Box
- Numbering

Separator

```
template {
        ?y
        ;  separator = ", "
}
where {
        ?in foaf:knows ?y
}
```

Format

```
template {
  format  {
        "<h2>%1$s</h2><p>%2$s</p>"

        st:apply-templates(?x)
        st:apply-templates(?y)
  }
}
where {
}
```

External Format

```
template {
  format  {
        <http://example.org/format/test.html>

        st:apply-templates(?x)
        st:apply-templates(?y)
  }
}
where {
}
```

Format Function

st:format(format, exp+)

Group

group { E1 .. En }

::=

group_concat(concat(E1, .. En))

Group

```
template {
        ?in " : " group { ?y }
}
where {
        ?in foaf:knows ?y
}
```

Group

group { E1 .. En ; separator = "--" }

Box Indentation

box { E1 .. En }

::=

concat(E1, .. En)

+

st:nl()

box | sbox | ibox

Box

box: nl(+1) exp nl(-1)

sbox: nl(+1) exp indent(-1)

ibox: indent(+1) exp indent(-1)

Numbering

```
template {
        st:number() " " st:apply-templates(?x)
}
where {
        ?in foaf:knows ?y
}
order by ?x
```

Hints

Store Partial Result

```
template {
        ?head " " ?body
}
where {
        bind (st:call-template(st:body) as ?body)
        bind (st:call-template(st:head) as ?head)
}
```

Split Complex Transformation

```
template {
        st:apply-templates-with(st:anotherone, ?x)
}
where {
}
```

LDScript Function

```
template { us:fun(?x) }
where { }

function us:fun(x) {
        # nice pretty print for x
}
```

LDScript Global Variable

```
function us:start() {
        set (amap = xt:map())
}

function us:map() {
        amap
}
```

LDScript Global Variable

```
template st:start {
}
where {
        bind (us:start() as ?tmp)
}
```

LDScript Global Variable

```
template {}
where {
        ?uri foaf:name ?value
        bind (xt:set(us:map(), ?uri, ?value) as ?tmp)
}
```

LDScript Global Variable

```
template {
        key " " val
}
where {
        values (key val) { unnest ( us:map() )}
}
```

Compiling STTL into SPARQL

template { E1 .. En }
where {}

compiled as :

select (concat(cp(E1), .. cp(En)) as ?out)
where {}
+
aggregate(Ω, group_concat, ?out)

STTL Compilation

cp(Var(x)) = st:process(x)

Default:

st:process(?x) = st:turtle(?x)

Overloaded:

```
function st:process(?x) {
  st:apply-templates(?x)
}
```

STTL Transformations

1.	RDF to Turtle	st:turtle
2.	RDF to RDF/XML	st:rdfxml
3.	RDF to JSON-LD	st:jsonld
4.	OWL to Functional Syntax	st:owl
5.	SPIN to SPARQL	st:spin
6.	SPARQL Query Result	st:sparql
7.	SPARQL Tutorial	st:web
8.	DBpedia Navigator	st:navlab
9.	Wikipedia Edit History Navigator	st:dbedit
10.	Calendar	st:calendar
11.	History Timeline	
12.	Sudoku (1 template)	
13.	OWL Profile check	st:owlrl
14.	*SHACL Validation*	*st:dsmain*

Usage

Create a directory e.g. sttl

Write templates in separate files in directory, extension .rq

```
template {
st:apply-templates-with("/home/myself/sttl/")
}
where {}
```

Define Function

Create a directory, e.g. "fun"

Write functions in a file with ".rq" extension

```
@import <fun/file.rq>

template st:profile {}
where {}
```

Usage Java

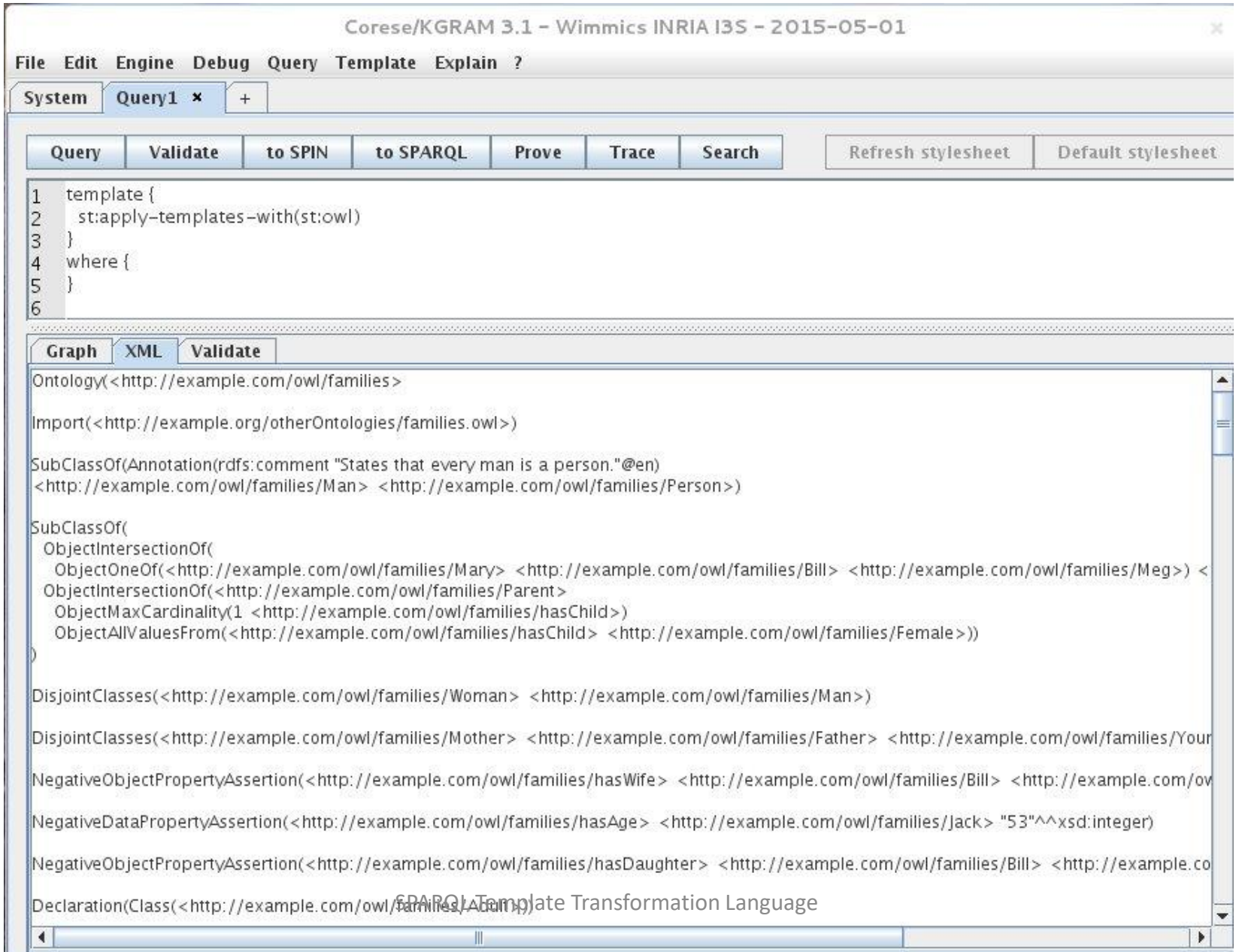
```
Transformer t = Transformer.create(g, "/home/myself/sttl/");
String str = t.transform();
```

Compile Transformation (optional)

GUI:

- File/Compile Transformation
- Select Transformation Directory
- File/Save Result
- Save in a file with ".rul" extension

STTL development environment



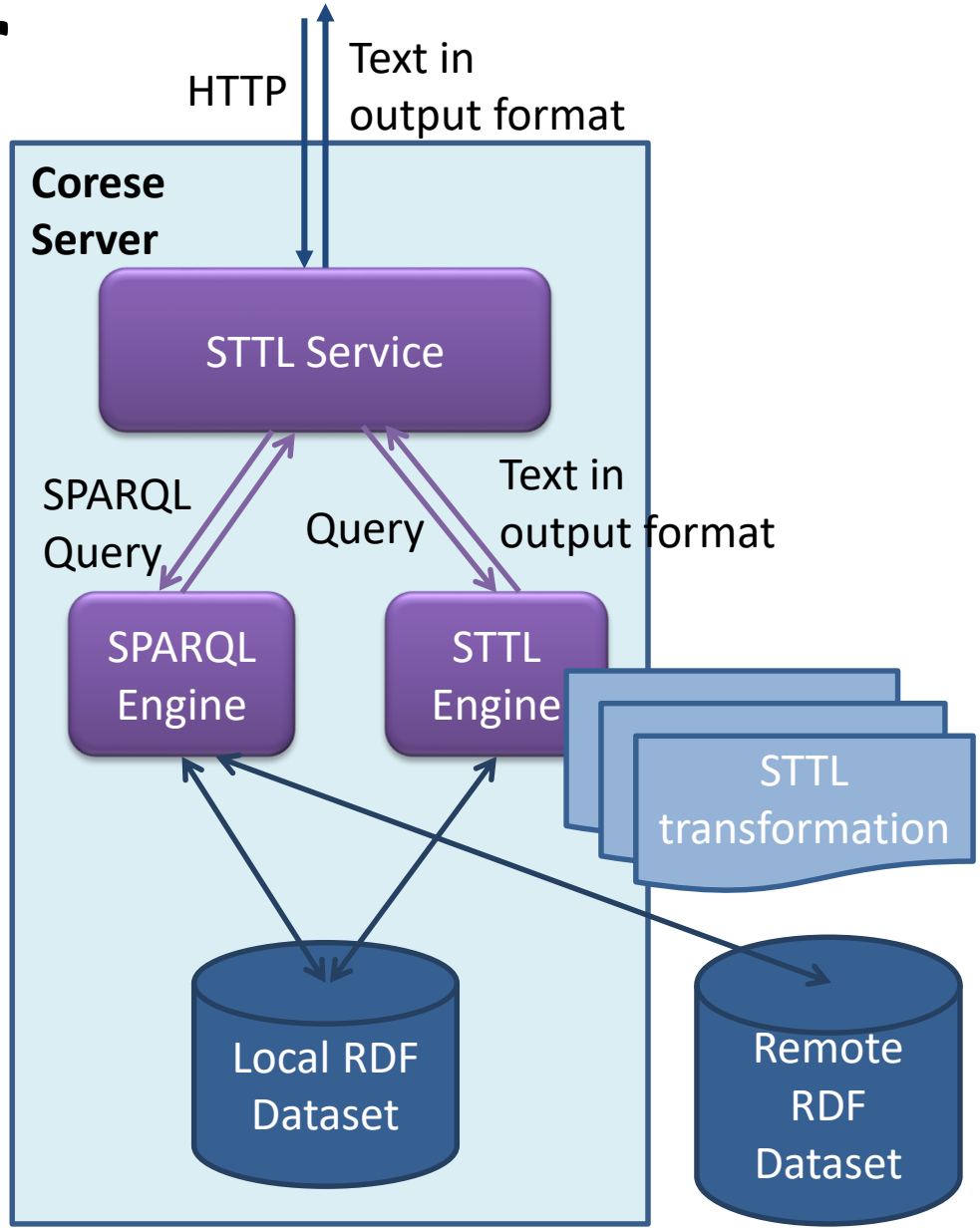
Application

STTL engine

available in the Corese Semantic Web Factory

- Free download: http://project.inria.fr/corese
  - SPARQL engine
  - STTL engine
  - Standalone environment to develop transformation
  - SPARQL endpoint
  - STTL server
- Web Server

STTL Server


HTTP
Text in
output format
Corese
Server
STTL Service
SPARQL
Query
Query
Text in
output format
SPARQL
Engine
STTL
Engine
STTL
transformation
Local RDF
Dataset
Remote
RDF
Dataset

Corese Web Server Demo

localhost:8080

Corese | SPARQL Tutorial | SPARQL-SPIN Converter | OWL | Others | Sudoku Solver

CORESE Web Server

Corese is a Semantic Web Factory implementing RDF, RDFS, SPARQL and Inference Rules. This site presents demos of Semantic Web servers and Linked Data Navigators designed with SPARQL Template Transformation Language.

Linked data browsers

Louis XIV de France
(1638 - 1715)

Auguste (dbpeida fr)

Auguste (dbpedia)

Emmanuel-Philibert de Savoie
(1528-1580)

Places
(Nice)

History
(XIVe Siècle)

Online services

SPARQL Query

Server

```
select * where {
  ?x ?p ?y
}
```

Query

DBpedia Query

STD

```
select * where {
  service <http://fr.dbpedia.org/sparql> {
    <http://fr.dbpedia.org/resource/Antibes> ?p ?y
  }
}
limit 10
offset 10
```

Query

Self Service

RDF graph URI:

Format:

st:turtle

Transform

Contact: Olivier Corby

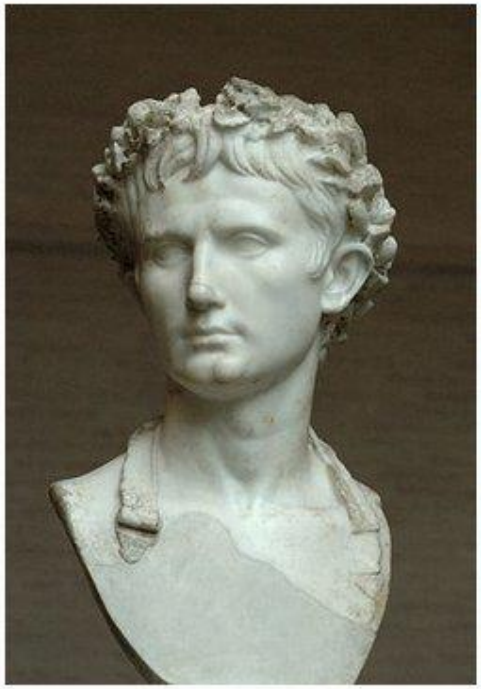
Designed by Fuqi Song using Simplex css style

Corese Web Server

corese.inria.fr/srv/template?uri=http://fr.dbpedia.org/resource/Auguste&profile=st:dbpedia

Corese | SPARQL Tutorial | SPARQL-SPIN Converter | OWL | Others | Sudoku Solver

Auguste



Naissance	-63-09-23+02:00
Décès	14-08-19+02:00
Prédécesseur	Jules César
Successeur	Tibère
Père	Gaius Octavius
Mère	Atia Balba Caesonia
Conjoints	Scribonia (épouse d'Octavien) Clodia Pulchra Livie
Enfants	Julia Caesaris filia
Résumé	Auguste, né sous le nom de Caius Octavius le 23 septembre 63 av. J.-C. à Rome, d'abord appelé Octave puis Octavien, porte le nom de Imperator Caesar Divi Filius Augustus à sa mort le 19 août 14 ap. J.-C. à Nola. Il est le premier empereur romain, du 16 janvier 27 av. J.-C. au 19 août 14 ap. J.-C.Issu d'une ancienne et riche famille de rang équestre appartenant à la gens plébéienne des Octavii, il devient fils adoptif posthume de son grand-oncle maternel Jules César en 44 av.
Wikipedia	http://fr.wikipedia.org/wiki/Auguste
DBpedia	http://fr.dbpedia.org/resource/Auguste

Contact: Olivier Corby

Designed by Fuqi Song using Simplex css style

Corese Web Server Demo – Mozilla Firefox

File Edit View History Bookmarks ScrapBook Tools Help

Corese Web Server Demo

corese.inria.fr/srv/template?uri=http://fr.dbpedia.org/res

Google

Nord	Colomars Falicon Saint-André-de-la-Roche
Nord Est	La Trinité (Alpes-Maritimes)
Est	Villefranche-sur-Mer
Sud Est	
Sud	
Sud Ouest	
Ouest	Saint-Jeannet (Alpes-Maritimes) La Gaude
Nord Ouest	Gattières
Latitude	43.6959
Longitude	7.27141
Wikipedia	http://fr.wikipedia.org/wiki/Nice
DBpedia	http://fr.dbpedia.org/resource/Nice

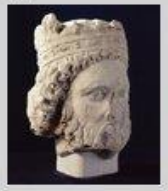
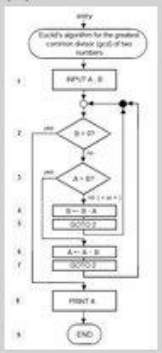
Corese Web Server

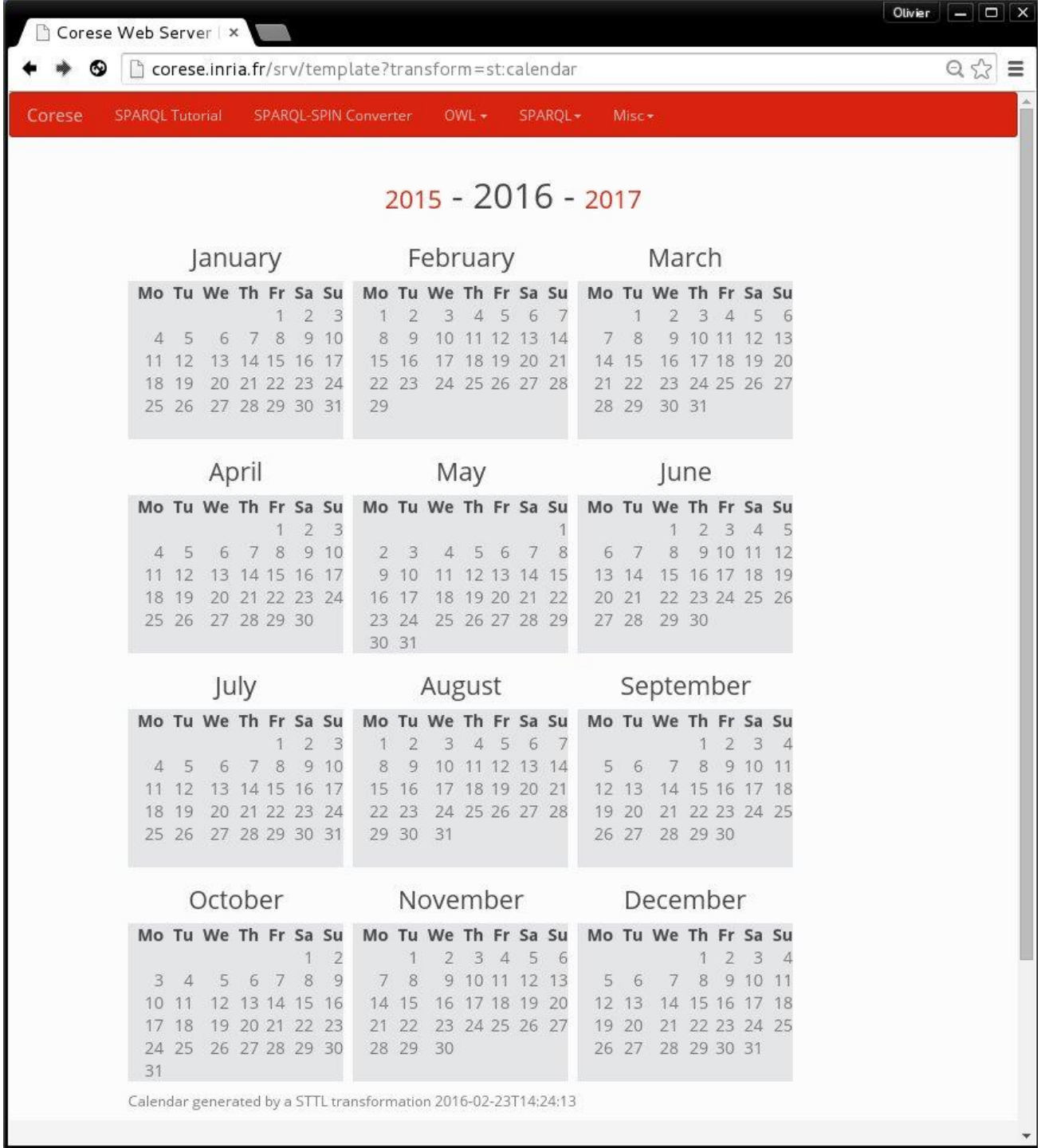
corese.inria.fr/srv/template?uri=http://ns.inria.fr/sparql-template/date/01/2002&profile=st:d

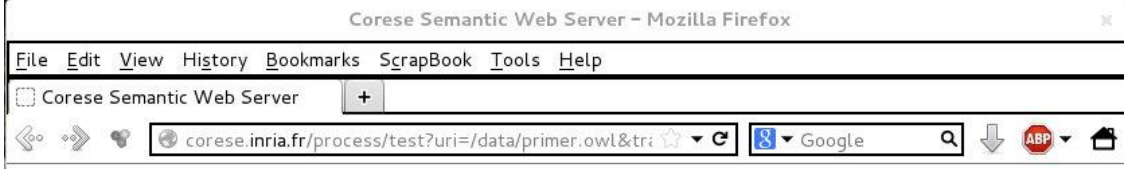
Corese | SPARQL Tutorial | SPARQL-SPIN Converter | OWL | SPARQL | Misc

DBpedia History 01/2002

01/2001 << 12/2001 << 01/2002 >> 02/2002 >> 01/2003

1	Clotaire Ier (2) 	John McCarthy (2) 	Algorithmique (1) 	Carl Sagan (1) 	Dagobert Ier (1) 	
2	Espéranto (1) 	GNU (1) 	Iron Maiden (1) 	Lisp (1) 	Linus Torvalds (1) 	
3	Blanc d'œuf (1) 	Modèle standard (physique des particules) (1) 	Tourisme (1) 			


Corese Web Server
Olivier
corese.inria.fr/srv/template?transform=st:calendar
Corese
SPARQL Tutorial
SPARQL-SPIN Converter
OWL
SPARQL
Misc
2015 - 2016 - 2017
January
February
March
April
May
June
July
August
September
October
November
December
Mo Tu We Th Fr Sa Su
Calendar generated by a STTL transformation 2016-02-23T14:24:13

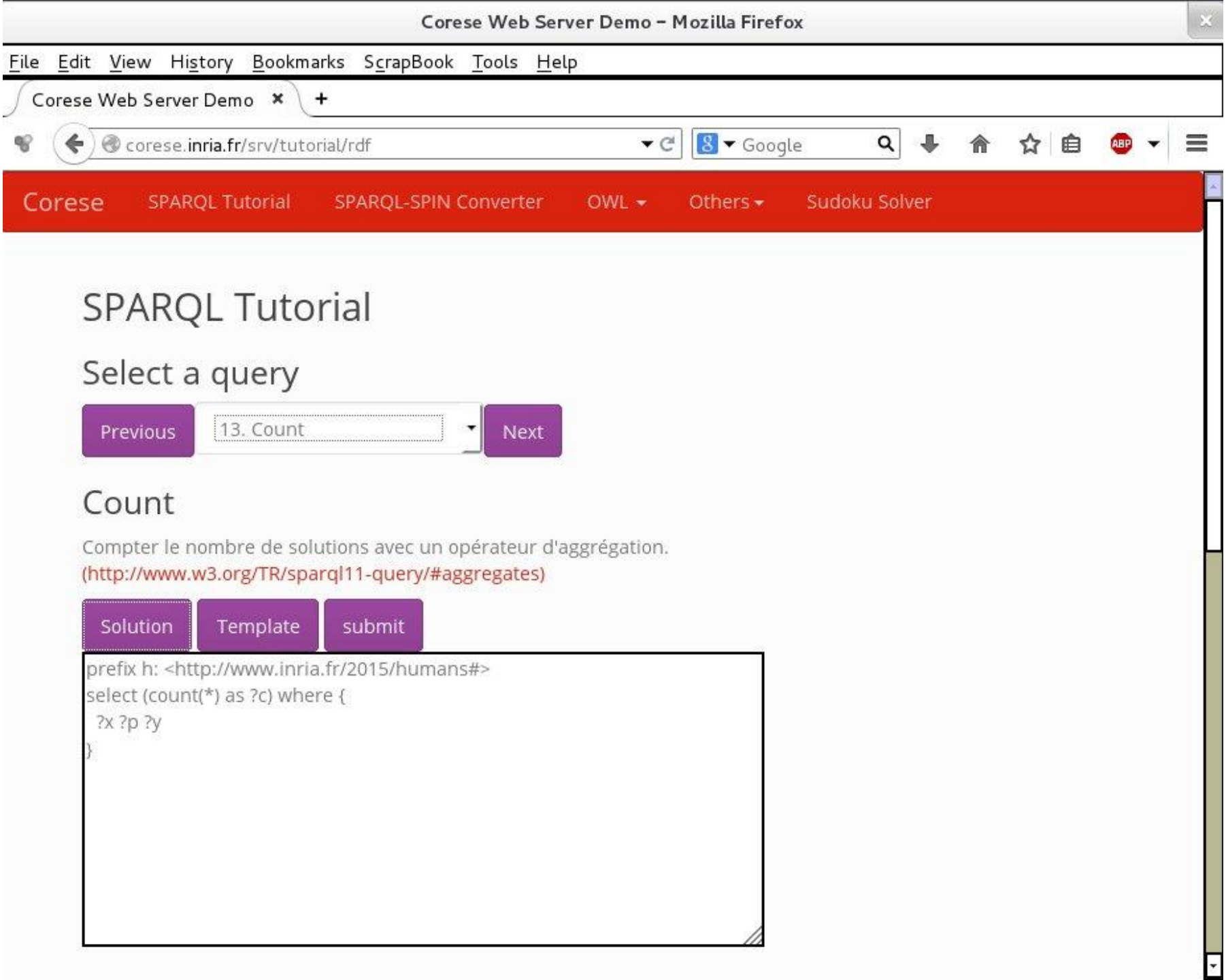


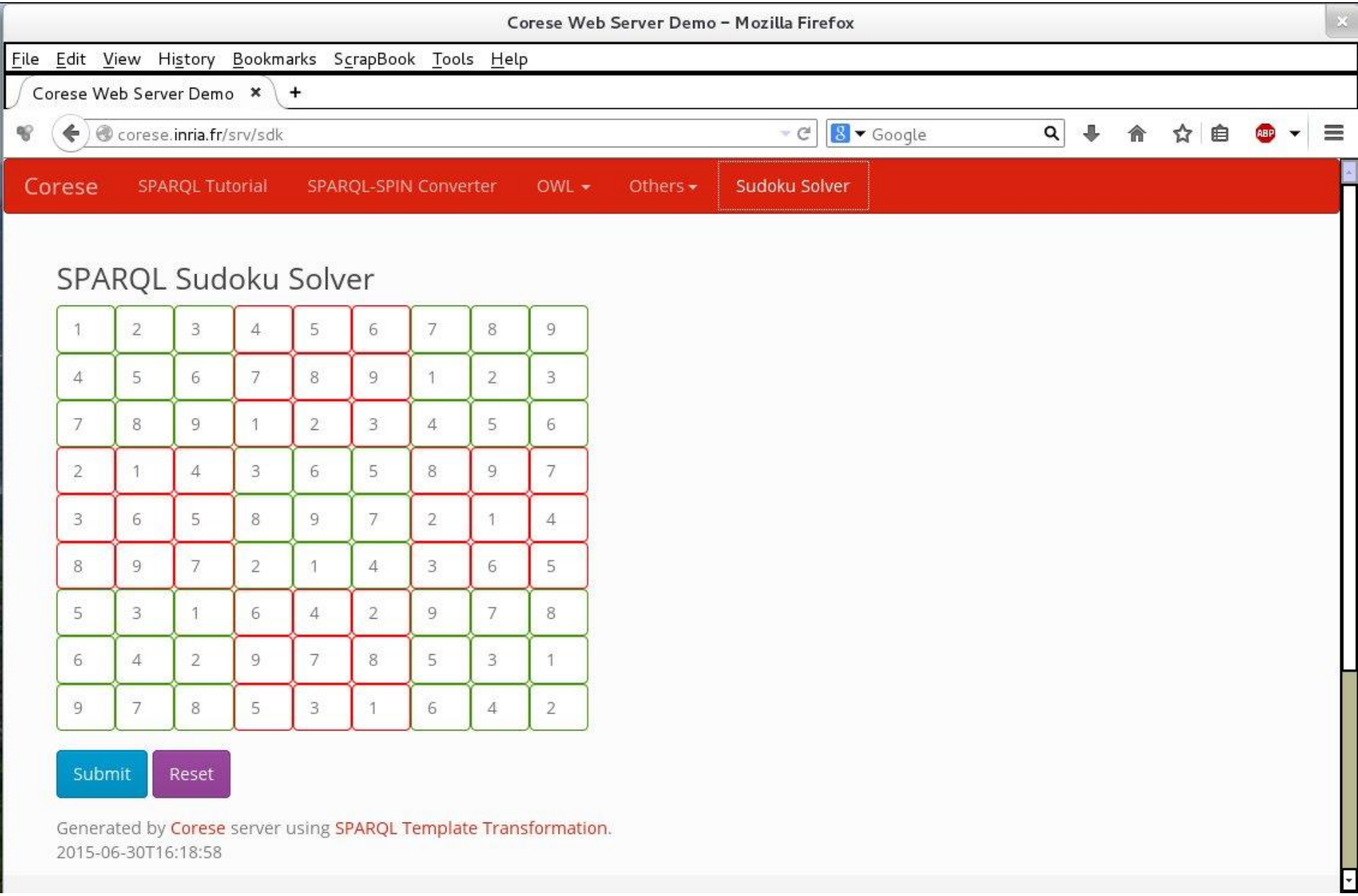
Load: /data/primer.owl
Transform: st:owl

Result

```
Ontology(<http://example.com/owl/families>

Import(<http://example.org/otherOntologies/families.owl>)

SubClassOf(Annotation(rdfs:comment "States that every man is a person.")
<http://example.com/owl/families/Man> <http://example.com/owl/families/Person>)

SubClassOf(
  ObjectIntersectionOf(
    ObjectOneOf(<http://example.com/owl/families/Mary> <http://example.com
/owl/families/Bill> <http://example.com/owl/families/Meg>) <http://example.com
/owl/families/Female>)
  ObjectIntersectionOf(<http://example.com/owl/families/Parent>
    ObjectMaxCardinality(1 <http://example.com/owl/families/hasChild>)
    ObjectAllValuesFrom(<http://example.com/owl/families/hasChild>
<http://example.com/owl/families/Female>))
)

DisjointClasses(<http://example.com/owl/families/Woman> <http://example.com
/owl/families/Man>)

DisjointClasses(<http://example.com/owl/families/Mother> <http://example.com
/owl/families/Father> <http://example.com/owl/families/YoungChild>)

NegativeObjectPropertyAssertion(<http://example.com/owl/families/hasWife>
<http://example.com/owl/families/Bill> <http://example.com/owl/families/Mary>)

NegativeDataPropertyAssertion(<http://example.com/owl/families/hasAge>
<http://example.com/owl/families/Jack> "53"^^xsd:integer)

NegativeObjectPropertyAssertion(<http://example.com/owl/families/hasDaughter>
<http://example.com/owl/families/Bill> <http://example.com/owl/families/Susan>)

Declaration(Class(<http://example.com/owl/families/Adult>))

EquivalentClasses(<http://example.com/owl/families/Adult> <http://example.org
/otherOntologies/families/Grownup>)

Declaration(Class(<http://example.com/owl/families/ChildlessPerson>))

EquivalentClasses(<http://example.com/owl/families/ChildlessPerson>
```

Generated by SPARQL Template Transformation using Corese 2015-05-11T10:48:00


Corese Web Server Demo – Mozilla Firefox
File Edit View History Bookmarks ScrapBook Tools Help
Corese Web Server Demo
corese.inria.fr/srv/tutorial/rdf
Google
Corese
SPARQL Tutorial
SPARQL-SPIN Converter
OWL
Others
Sudoku Solver
SPARQL Tutorial
Select a query
Previous
13. Count
Next
Count
Compter le nombre de solutions avec un opérateur d'aggrégation.
(http://www.w3.org/TR/sparql11-query/#aggregates)
Solution
Template
submit
prefix h: <http://www.inria.fr/2015/humans#>
select (count(*) as ?c) where {
?x ?p ?y
}


Corese Web Server Demo – Mozilla Firefox
File Edit View History Bookmarks ScrapBook Tools Help
Corese Web Server Demo
corese.inria.fr/srv/sdk
Google
Corese
SPARQL Tutorial
SPARQL-SPIN Converter
OWL
Others
Sudoku Solver
SPARQL Sudoku Solver
1 2 3 4 5 6 7 8 9
4 5 6 7 8 9 1 2 3
7 8 9 1 2 3 4 5 6
2 1 4 3 6 5 8 9 7
3 6 5 8 9 7 2 1 4
8 9 7 2 1 4 3 6 5
5 3 1 6 4 2 9 7 8
6 4 2 9 7 8 5 3 1
9 7 8 5 3 1 6 4 2
Submit
Reset
Generated by Corese server using SPARQL Template Transformation.
2015-06-30T16:18:58

Conclusion

- STTL Transformation Language for RDF
- Based on SPARQL
- XSLT like
- Transformation
- Presentation
- Constraint Checking

Exercice proposé

- Transformation pour le résultat de SHACL
- Java 11
- Corese: https://project.inria.fr/corese/download
- Lab: https://project.inria.fr/corese/tutorial

SHACL Validation Report

```
template {
  st:apply-templates-with-graph("/home/sttl", ?g)
}
where {
        bind (sh:shacl() as ?g)
}
```