

**SOL Manual**

**Jean-Paul A Barthès**

Division Informatique  
BP 349  
COMPIEGNE  
Tel +33 3 44 23 44 66  
Fax \*33 3 44 23 46 33

Email: [jean-paul.barthes@utc.fr](mailto:jean-paul.barthes@utc.fr)

---

**Memo UTC/GI/DI/N 199**  
June 2006

---

### **Warning**

This document is intended to explain how to install and use the SOL tools.

Keywords: SOL Compiler, SOL tools

**Revisions**

---

<b>Version</b>	<b>Date</b>	<b>Author</b>	<b>Remarks</b>
0	June 06	Barthès	Draft

## **Related Documents**

---

The following document compares OWL and MOSS for specifying an ontology

UTC/GI/N190 - MOSS vs. OWL

The following document describes the SOL to OWL compiler design choices.

UTC/GI/N191 - SOL

The following document describes the SOL ndex internal mechanism.

UTC/GI/N192 - SOL Technical Reference Index Mechanism

The following document describes the SOL rule mechanism.

UTC/GI/N198 - SOL Rules

## Contents

---

<b>1</b>	<b>INTRODUCTION.....</b>	<b>6</b>
<b>2</b>	<b>INSTALLING THE SOL SUITE.....</b>	<b>6</b>
2.1	WINDOWS .....	6
2.2	MAC OS X .....	8
<b>3</b>	<b>SOL CONTROL PANEL.....</b>	<b>8</b>
3.1	TITLE.....	8
3.2	SOL FILE .....	8
3.3	SOL TRACE MODE.....	8
3.4	SOL COMPILER: OWL OUTPUT .....	8
3.5	SOL COMPILER: HTML AND TEXT OUTPUTS.....	8
3.6	SOL COMPILER: RULE EXTRACTION.....	8
3.7	COMPILE BUTTON.....	8
<b>4</b>	<b>A FIRST TEST .....</b>	<b>9</b>
4.1	THE TEST FILE .....	9
4.2	CHECKING SYNTAX : NO ERROR .....	10
4.3	BASIC SYNTAX ERRORS .....	10
4.4	BASIC ONTOLOGY FORMAT ERROR.....	11
4.5	PRODUCING OUTPUT FILES.....	12
<b>5</b>	<b>A SECOND TEST: SEVERAL CONCEPTS.....</b>	<b>15</b>
5.1	UNKNOWN SUPERCLASS.....	15
5.2	UNDEFINED RANGE CLASS.....	15
<b>6</b>	<b>THIRD TEST: INDIVIDUAL CONCEPTS/INSTANCES.....</b>	<b>16</b>
6.1	NON EXISTING CONCEPT .....	17
6.2	UNKNOWN INDIVIDUAL .....	17
6.3	UNKNOWN PROPERTY.....	18
<b>7</b>	<b>FOURTH TEST: ONTOLOGY STRUCTURE .....</b>	<b>18</b>
<b>8</b>	<b>FIFTH TEST: RULES.....</b>	<b>19</b>
<b>9</b>	<b>SIXTH TEST: FULL EXAMPLE.....</b>	<b>20</b>
<b>10</b>	<b>INITIALIZATION FILE.....</b>	<b>20</b>
<b>11</b>	<b>POSSIBLE PROBLEMS.....</b>	<b>21</b>
11.1	THE SOL COMMAND PANEL DOES NOT APPEAR.....	21
11.2	UNJUSTIFIED PARENTHESIS OR QUOTE ERROR.....	21

---

**1****Introduction**

---

The main objective of this document is to describe how to install and to use the SOL tools.

SOL is intended to develop and maintain multilingual ontologies, and to provide a translation into an OWL format, as well as HTML files or files to display the ontologies using a graphical interface. The suite also extracts rules from the ontology into different files with different format according to the reasoner being used.

The current manual is intended for the *ontology specialist*.

---

**2****Installing the SOL Suite**

---

The installation depends on the platform. Installation is described for Windows and for Mac OS X.

**2.1 Windows**

The installation is described for the Windows-32 environment, and has been tested on XP.

The SOL suite comes as a zipped file containing all the necessary software. Unzip the file and install the corresponding SOL folder in your directory. We recommend installing it at the top level of your main disk.

The folder should contain the following important subfolders:

- clisp-2.37
- ltk-0.881
- ontologies
- sol-tools
- tcl-SOL

The other ones can be safely ignored.

The SOL folder contains a batch file:

- sol.bat

SOL is started by double clicking the batch file. A command window appears with a series of messages (Figure 1), followed by the SOL control panel as shown Figure 2.

```

c:\WINDOWS\system32\cmd.exe
;; Compiling file C:\SOL\sol-tools\sol-control.lisp ...
;; Wrote file C:\SOL\sol-tools\sol-control.fas
The following functions were used but not defined:
SOL-OWL::COMPILE-SOL SOL-HTML::COMPILE-SOL
0 errors, 0 warnings
;; Compiling file C:\SOL\sol-tools\sol2owl-v1.6.lisp ...
WARNING in CHECK-SYNTAX in lines 669..753 :
variable DUMMY is not used.
Misspelled or missing IGNORE declaration?
;; Wrote file C:\SOL\sol-tools\sol2owl-v1.6.fas
0 errors, 1 warning
*** Sol2owl loaded ***
;; Compiling file C:\SOL\sol-tools\sol2html-v1.2.lisp ...
;; Wrote file C:\SOL\sol-tools\sol2html-v1.2.fas
0 errors, 0 warnings
;; Compiling file C:\SOL\sol-tools\sol2rules-v0.1.lisp ...
;; Wrote file C:\SOL\sol-tools\sol2rules-v0.1.fas
The following functions were used but are deprecated:
GENTEMP - This function creates symbols that cannot be garbage-collected. Use G
ENSYM instead.
0 errors, 0 warnings
*** Sol2rules loaded ***
=====> "c:/sol/tcl-sol/bin/wish.exe"
WINDOW.w2ISNATOP-LEVELWINDOW

```

Figure 1 - SOL Command Window

DO NOT CLOSE THE COMMAND WINDOW as it is needed for control messages. However you can minimize it.

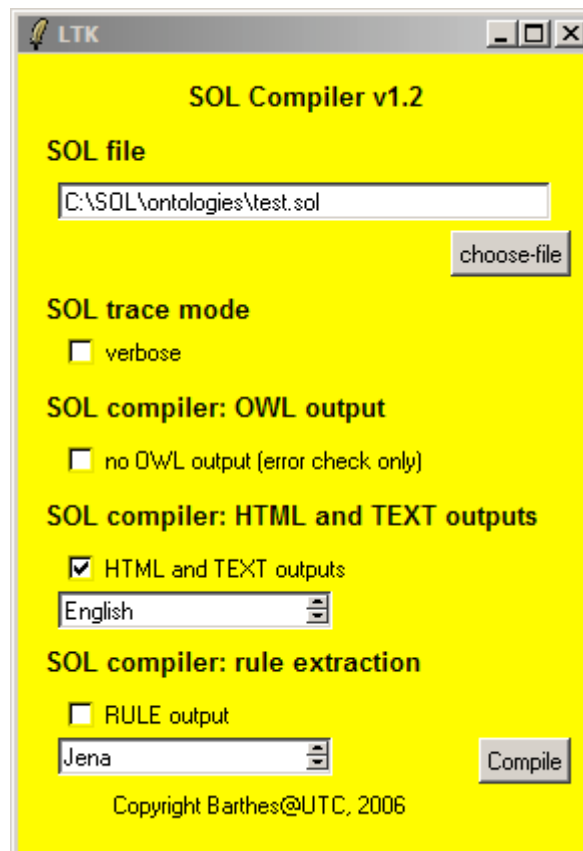


Figure 2 - SOL Control Panel

If the SOL control panel does not appear please refer to the Problems Section.

## 2.2 Mac OS X

Documentation not available yet.

# 3

## SOL Control Panel

---

The SOL control panel is shown Figure 2. The meaning of the different areas and buttons is given in the following paragraphs.

### 3.1 Title

The title indicates the current version of the software.

### 3.2 SOL file

Displays the name of the file containing the source ontology. The name is a default file that can be used as an example and will be described in subsequent sections.

The **choose file** button allows to choose a different file.

### 3.3 SOL trace mode

A single **verbose** check box allows controlling the printed output. Default is terse (non verbose).

### 3.4 SOL compiler: OWL output

A check box entitled **no OWL output** allows controlling the production of the OWL file. The box can be checked if one wants to verify the original ontology without building the actual output file. This option is handy to check errors. Default is to produce the output file.

### 3.5 SOL compiler: HTML and TEXT outputs

A check box controls the production of the corresponding output files. A rotary switch allows specifying the target language. Three files can be produced, one for display in a browser, and 2 for producing graphs. Default is generating the files. Default language is English.

### 3.6 SOL compiler: rule extraction

A check box controls the production of a special file containing rules. A rotary switch allows selecting the rule external format. Currently only JENA format is supported. Default is no output. Default format is JENA.

### 3.7 Compile button

Launches the compilation. The trace is printed in a separate window (Figure 4).

The following sections describe the use of the SOL suite in details.



## 4

## A First Test

Ontology files are normally located in the ontology folder of the SOL folder. The ontology folder comes with several examples of ontology files, from very simple to very complex.

### 4.1 The Test File

The test0.sol file contains a minimal information, namely, the definition of a single concept (see UTC/GI/N191 - SOL for the SOL syntax) :

```
(defconcept
  (:name :en "Territory" :fr "Territoire" :it "territorio")
  (:att (:en "name" :fr "nom") (:type :name)(:unique))
  (:doc :en "A territory is a part of a geographical entity under the
    rule of nation or a part of it."
    :fr "Un territoire est une étendue de terre qu'offre un état,
    une province, une ville, une juridiction, etc.))

:EOF
```

The concept is one of territory, it has an attribute (a name) and a documentation in English and in French.

In order to produce an OWL file, one must first select the file by clicking the choose file button. A selection window appears allowing to select the file.

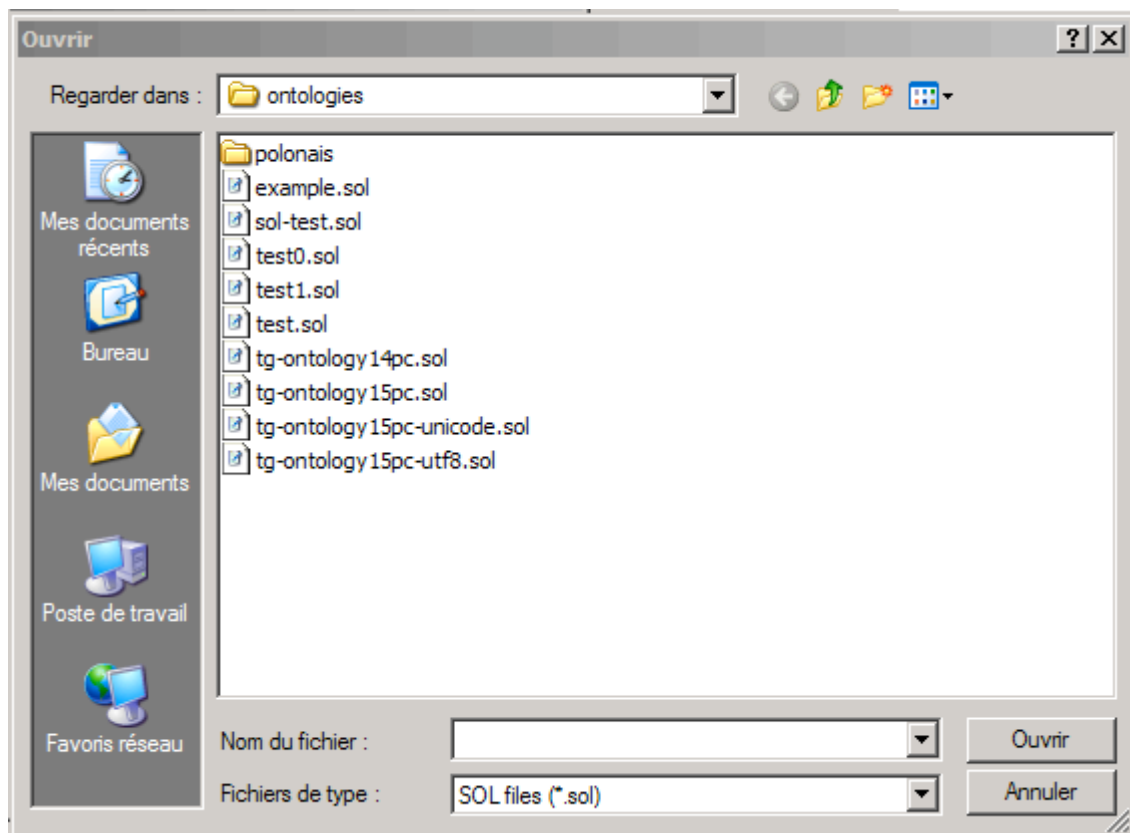
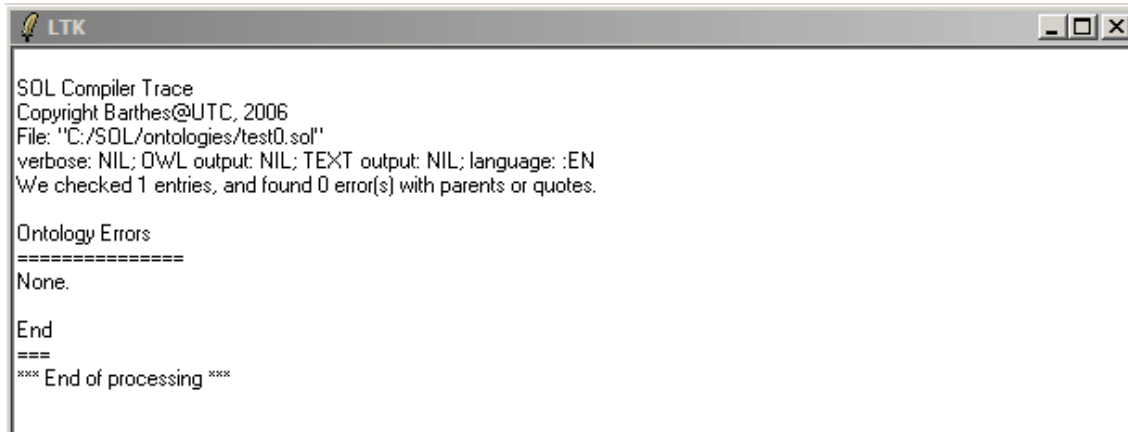


Figure 3 - Selection window

## 4.2 Checking Syntax : no error

Once this is done, a first check for errors can be done by selecting **no OWL output**, unselecting the HTML and TEXT files and clicking the **compile** button.

A new window appears as shown Figure 4.



```

LTK
SQL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SOL/ontologies/test0.sol"
verbose: NIL; OWL output: NIL; TEXT output: NIL; language: :EN
We checked 1 entries, and found 0 error(s) with parents or quotes.

Ontology Errors
=====
None.

End
===
**** End of processing ****

```

**Figure 4 - Compiling trace**

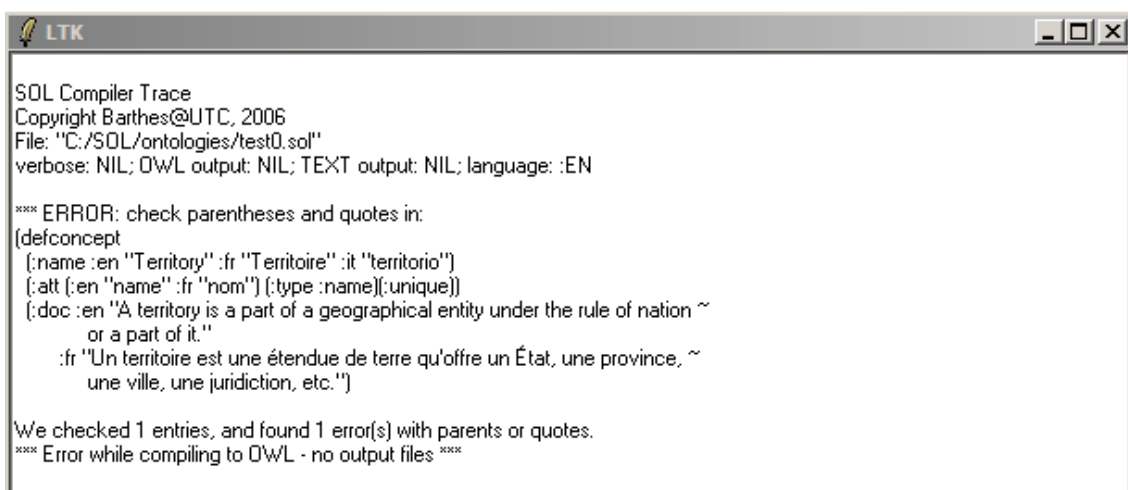
The trace says that the file to be compiled was « c:/SOL/ontologies/test0.sol » that the compiler was in the non verbose mode, and that no output files were requested, and the language was English (in the present case this setting is unimportant).

Then a first check indicates that the system did not detect any parenthesis or quote unbalance, and did not find any ontology error.

## 4.3 Basic Syntax Errors

Of course, most of the time there are some syntax errors.

First, let us see what happens if a parenthesis is missing. dit the test0.sol file and remove the last parenthesis. Close the trace window and recompile. The new result is shown



```

LTK
SQL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SOL/ontologies/test0.sol"
verbose: NIL; OWL output: NIL; TEXT output: NIL; language: :EN

**** ERROR: check parentheses and quotes in:
(defconcept
  (:name :en "Territory" :fr "Territoire" :it "territorio")
  (:att (:en "name" :fr "nom") (:type :name):unique))
  (:doc :en "A territory is a part of a geographical entity under the rule of nation ~
    or a part of it."
    :fr "Un territoire est une étendue de terre qu'offre un État, une province, ~
    une ville, une juridiction, etc.")

We checked 1 entries, and found 1 error(s) with parents or quotes.
**** Error while compiling to OWL - no output files ****

```

**Figure 5 - Syntax error**

The system tells you that there is a syntax error (mismatch of parentheses or quotes). It unfortunately cannot do much better. If you had asked for a file output, it tells you that it did not produce the output file.

You must correct the error before compiling again until you obtain the good result.

The same error message appears if you remove an opening parenthesis or a quote sign.

#### 4.4 Basic Ontology Format Error

Now assume that there are no more parenthesis or quote error, but that we make a mistake in some keyword. E.g., erase the column in front of the :en tag in the multilingual name attribute. Save and recompile. The result is shown on the next figure.

```

SQL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SQL/ontologies/test0.sol"
verbose: NIL; OWL output: T; TEXT output: NIL; language: :EN
We checked 1 entries, and found 0 error(s) with parents or quotes.

Ontology Errors
=====
,.***** Error in:(EN "name" :FR "nom"); bad name-list for making attribute-id: (EN "name" :FR "nom")

End
===
***** End of processing *****

```

**Figure 6 - Ontology Syntax Error**

In that case the diagnostic is more precise, however not as well located.

One can experiment with other types of error, e.g. removing the :fr language tag from the documentation property, which leads to the messages of the next figure.

```

SQL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SQL/ontologies/test0.sol"
verbose: NIL; OWL output: T; TEXT output: NIL; language: :EN
We checked 1 entries, and found 0 error(s) with parents or quotes.

Ontology Errors
=====
,.***** Error in:(EN "Territory" :FR "Territoire" :IT "territorio");
bad language tag
"Un territoire est une étendue de terre qu'offre un État, une province, ~
une ville, une juridiction, etc." in
(:EN
"A territory is a part of a geographical entity under the rule of nation ~
or a part of it."
"Un territoire est une étendue de terre qu'offre un État, une province, ~
une ville, une juridiction, etc.")

End
===
***** End of processing *****

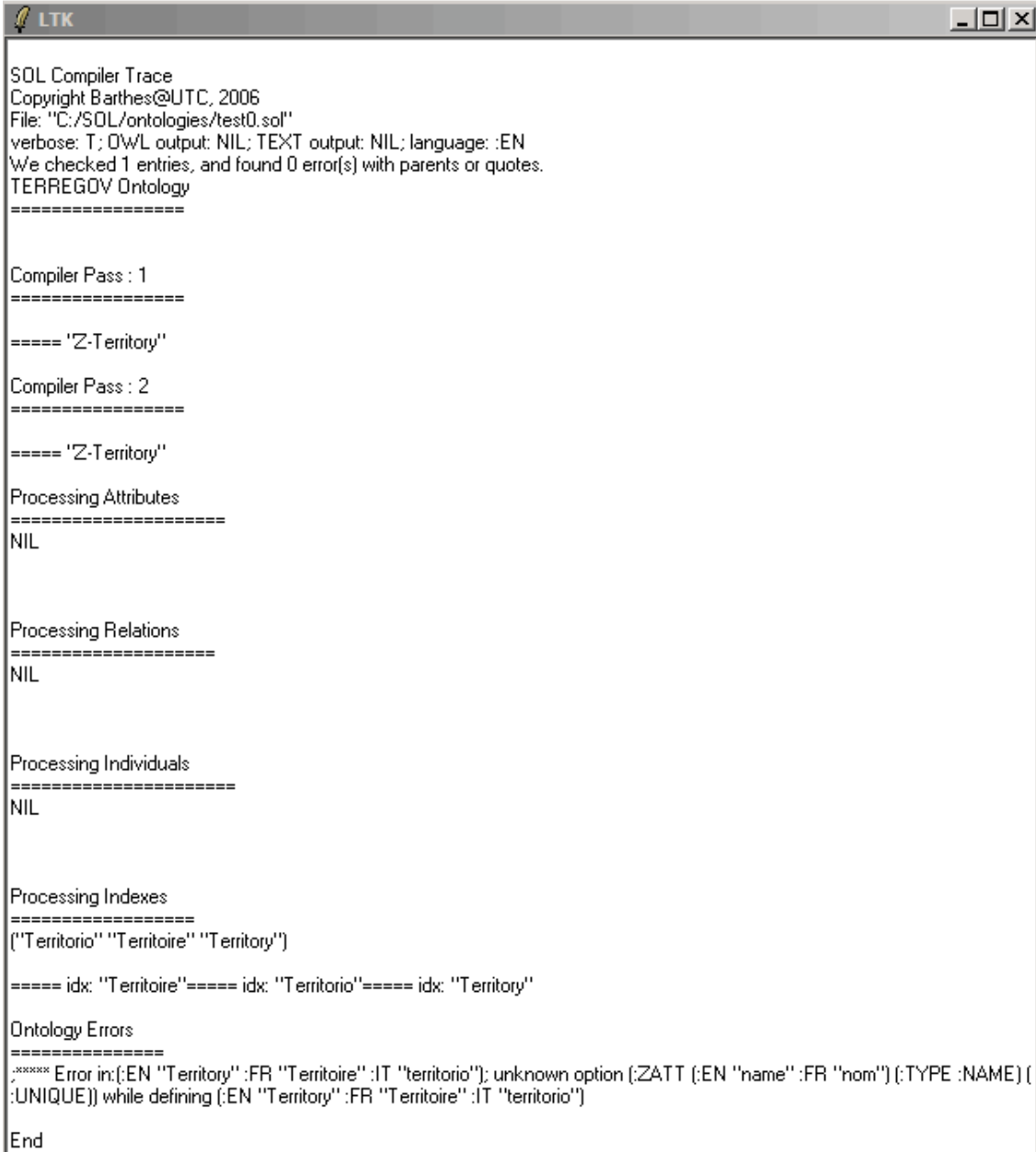
```

**Figure 7 - Missing Tag Error**

Note however that some errors are not caught like a misspelling of the `:type` option in the attribute. In that case the option is simply ignored. This will be corrected in the future.

### Verbose Option

If we check the verbose option, then the compiler trace is much more detailed as shown on the next figure.



```

LTK
SQL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SOL/ontologies/test0.sol"
verbose: T; OWL output: NIL; TEXT output: NIL; language: :EN
We checked 1 entries, and found 0 error(s) with parents or quotes.
TERREGOV Ontology
=====

Compiler Pass : 1
=====

==== "Z-Territory"

Compiler Pass : 2
=====

==== "Z-Territory"

Processing Attributes
=====
NIL

Processing Relations
=====
NIL

Processing Individuals
=====
NIL

Processing Indexes
=====
("Territorio" "Territoire" "Territory")

==== idx: "Territoire"==== idx: "Territorio"==== idx: "Territory"

Ontology Errors
=====
;***** Error in:(:EN "Territory":FR "Territoire":IT "territorio"); unknown option (:ZATT (:EN "name":FR "nom") (:TYPE :NAME) (:UNIQUE)) while defining (:EN "Territory":FR "Territoire":IT "territorio")

End

```

**Figure 8 - Compiler trace with a verbose option**

## 4.5 Producing Output Files

Of course the game is to produce output files. Assuming that we have no more errors we can remove the check in the no OWL output box and see what happens.

Looking into the ontologies folder one finds several new files :

- test0.fasl
- test0.lib
- test0.owl

Two of the files (test0.fasl and test0.lib) are for internal purposes. The other one (test0.owl) is the produced OWL file. Part of its content is shown on the next figure.

```

<owl:Class rdf:ID="Z-Territory">
  <rdfs:label xml:lang="en">Territory</rdfs:label>
  <rdfs:label xml:lang="fr">Territoire</rdfs:label>
  <rdfs:label xml:lang="it">territorio</rdfs:label>
  <rdfs:comment xml:lang="en">A territory is a part of a geographical entity under the rule of
nation or a part of it.</rdfs:comment>
  <rdfs:comment xml:lang="fr">Un territoire est une Åtendue de terre qu'offre un Åtat, une
province, une ville, une juridiction, etc.</rdfs:comment>
</owl:Class>

  <owl:DatatypeProperty rdf:ID="hasName">
    <rdfs:label xml:lang="en">name</rdfs:label>
    <rdfs:label xml:lang="fr">nom</rdfs:label>
    <rdfs:domain rdf:resource="#Z-Territory"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#Name"/>
  </owl:DatatypeProperty>

  <owl:Class rdf:about="#Z-Territory">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#hasName"/>
        <owl:cardinality rdf:datatype="xsd:nonNegativeInteger">1</owl:cardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

```

**Figure 9 - Produced OWL file**

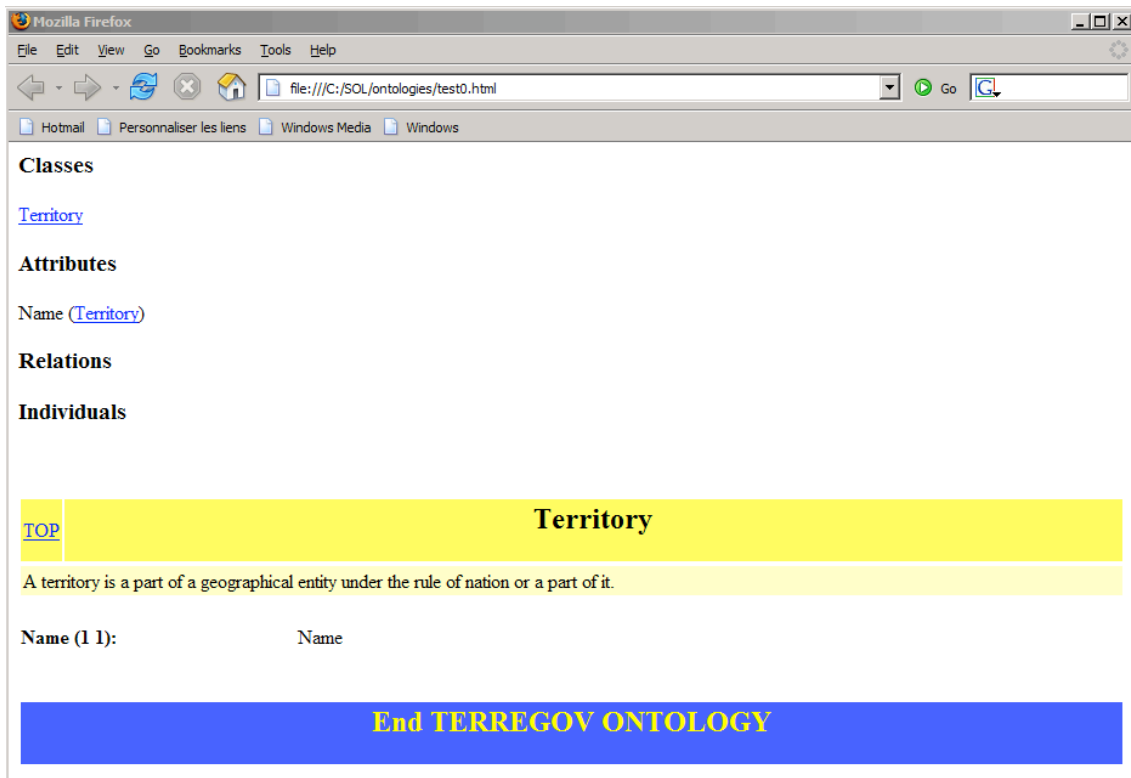
One can see that an OWL class has been produced with identifier Z-Territory, containing the documentation with the corresponding language tags. A datatype property has been produced for the name attribute, and a restriction on the name attribute (cardinality) has been produced. Other objects are produced in the OWL file.

When the HTML and TEXT is checked three more files are produced :

- test0.html
- test0.txt
- test0.isa

The html file is intended to be viewed through a web browser, as shown in the next figure. Note that there is only one concept (Territory) and one attribute (Name) in the file. Note that the file is in English.

The txt and isa files are intended to be used with graphers (currently Touchgraph).



**Figure 10 - HTML view of the ontology (English)**

If the language is set to French before producing the output files, then the output is shown on the next file (note that Name associated with **nom** is the type of data).



**Figure 11 - HTML view of the ontology (French)**

**5****A Second Test: Several Concepts**

Building an ontology with a single concept is not very realistic. In this section we take a second example involving more concepts. This is still not very realistic but slightly more complex.

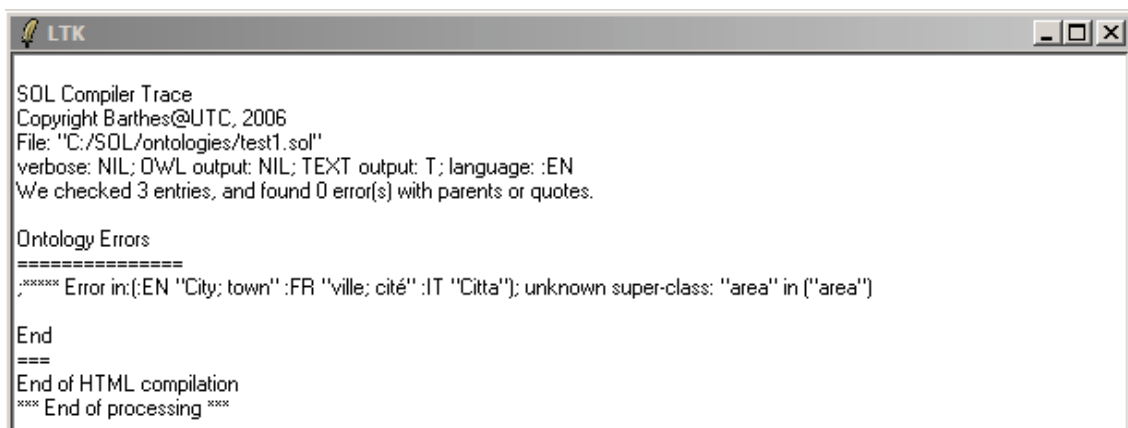
The test file is now test1.sol. It contains 3 concepts Territory, Country, and City. In addition City and Country are subclasses of Territory and City has-country a Country (not that the reference to the class has been set to the Italian label “Stato” indicating that any label can be used to designate the class).

As previously various files may be produced and examined. However, we are interested in the possible errors that could occur, in addition to the errors that we already encountered.

Some errors can occur due to a bad ontology structure.

**5.1 Unknown Superclass**

Let us change the :is-a property of the City concept to “area.” SOL signals an unknown superclass, as shown in the next figure.



```

LTK
SQL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SQL/ontologies/test1.sol"
verbose: NIL; D\WL output: NIL; TEXT output: T; language: :EN
We checked 3 entries, and found 0 error(s) with parents or quotes.

Ontology Errors
=====
,***** Error in:(EN "City; town" :FR "ville; cité" :IT "Citta"); unknown super-class: "area" in ("area")

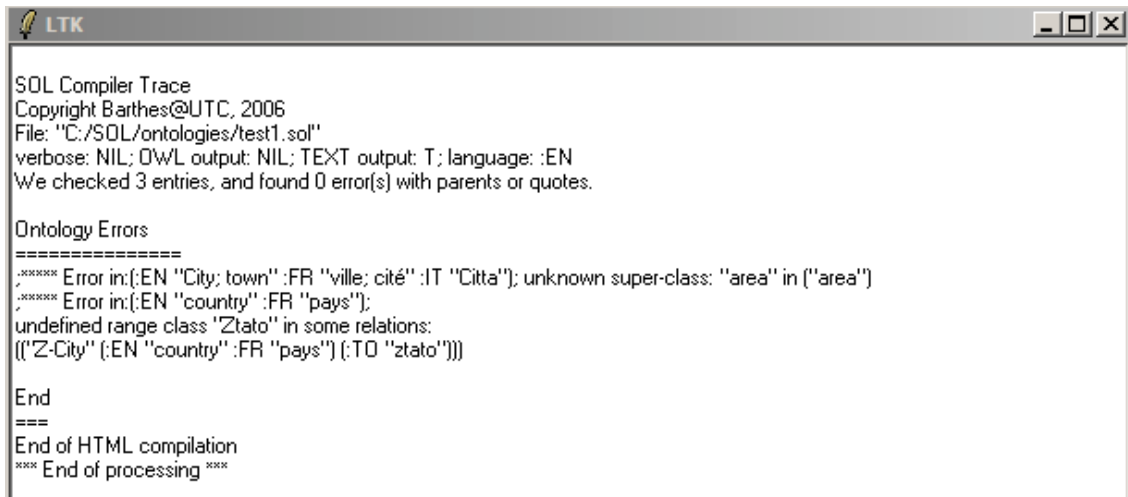
End
===
End of HTML compilation
*** End of processing ***

```

**Figure 12 - Unknown superclass**

**5.2 Undefined Range Class**

If we modify the range of the City class replacing stato with ztato to simulate a typing error, then we obtain the errors shown Figure 13.



```

LTK
SOL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SOL/ontologies/test1.sol"
verbose: NIL; OWL output: NIL; TEXT output: T; language: :EN
We checked 3 entries, and found 0 error(s) with parents or quotes.

Ontology Errors
=====
.:xxxx Error in:(:EN "City; town":FR "ville; cité":IT "Citta"); unknown super-class: "area" in ("area")
.:xxxx Error in:(:EN "country":FR "pays");
undefined range class "Ztato" in some relations:
[("Z-City" (:EN "country":FR "pays") (:TO "ztato"))]

End
===
End of HTML compilation
*** End of processing ***

```

**Figure 13 - Undefined Range Class**

## 6 **Third Test: Individual Concepts/Instances**

The ontology can contain individual concepts or instances (OWL jargon).

Let us consider the test2.sol file that contains two instance of a city and of a country, as follows:

```

defindividual "city"
  (:name :en "Bordeaux")
  ("pays" "france"))

(defindividual "pays"
  (:name :en "France" :fr "France")

  ("nom" (:en "France" :fr "France")))

```

The resulting OWL file contains entries for the individual concepts as follows:

```

<Z-Country rdf:ID="z-france">
  <rdfs:label xml:lang="en">France</rdfs:label>
  <rdfs:label xml:lang="fr">France</rdfs:label>
  <hasName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">(EN France FR
France)</hasName>
  <isCountryOf rdf:resource="#z-bordeaux"/>
</Z-Country>

<Z-City rdf:ID="z-bordeaux">
  <rdfs:label xml:lang="en">Bordeaux</rdfs:label>
  <hasCountry rdf:resource="#z-france"/>
</Z-City>

```

Note that the file contains additional objects.

The result may be displayed using the HTML file as shown Figure 14.



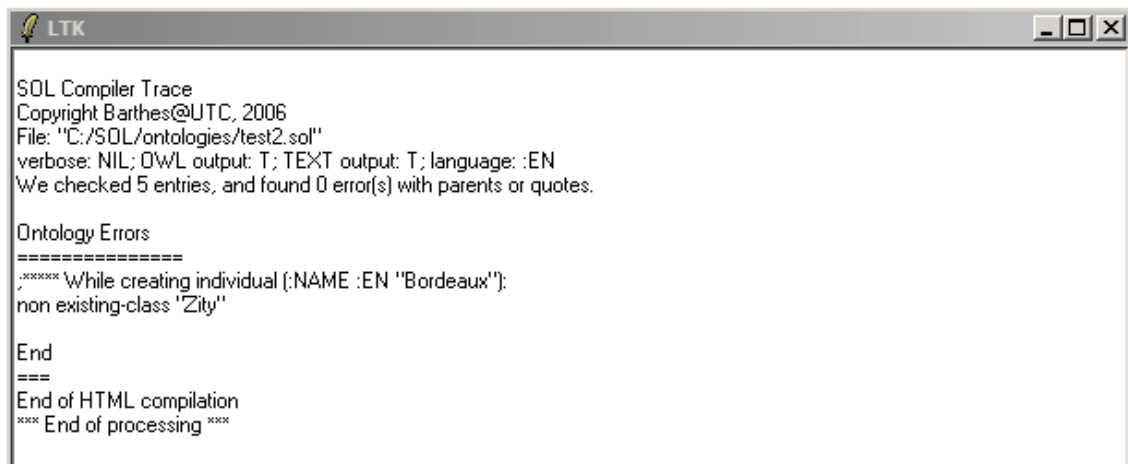
<a href="#">TOP</a>	<b>Country / Land</b>
a Country or a state is an administrative entity.	
Specific of:	<a href="#">Territory</a>
<a href="#">TOP</a>	<b>Bordeaux</b>
Class:	<a href="#">City</a>
Country:	<a href="#">France</a>
<a href="#">TOP</a>	<b>France</b>
Class:	<a href="#">Country</a>
Name:	(EN France FR France)

**Figure 14 - HTML file showing individual concepts**

Of course there may be some errors in the definition of the individual concepts.

### 6.1 Non Existing Concept

If an individual concept makes a reference to a non existing concept of the ontology, then an error message is printed (Figure 15).



```

LTK
SQL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SQL/ontologies/test2.sol"
verbose: NIL; OWL output: T; TEXT output: T; language: :EN
We checked 5 entries, and found 0 error(s) with parents or quotes.

Ontology Errors
=====
;***** While creating individual (:NAME :EN "Bordeaux"):
non existing-class 'Zity'

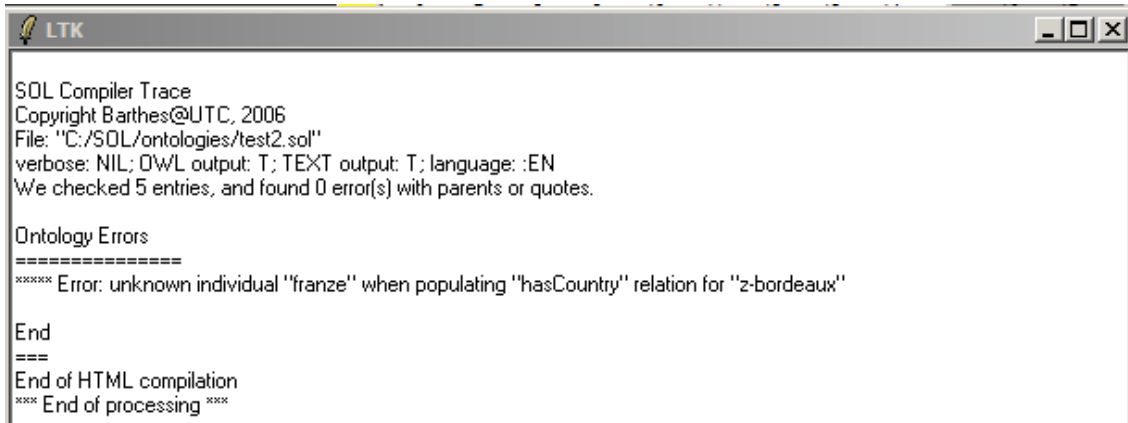
End
====
End of HTML compilation
*** End of processing ***

```

**Figure 15 - Non Existing Concept**

### 6.2 Unknown Individual

If an individual refers to a non existing individual in a relation, then an error message is printed as shown on the next figure.



```

LTK
SQL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SOL/ontologies/test2.sol"
verbose: NIL; OWL output: T; TEXT output: T; language: :EN
We checked 5 entries, and found 0 error(s) with parents or quotes.

Ontology Errors
=====
***** Error: unknown individual "franze" when populating "hasCountry" relation for "z-bordeaux"

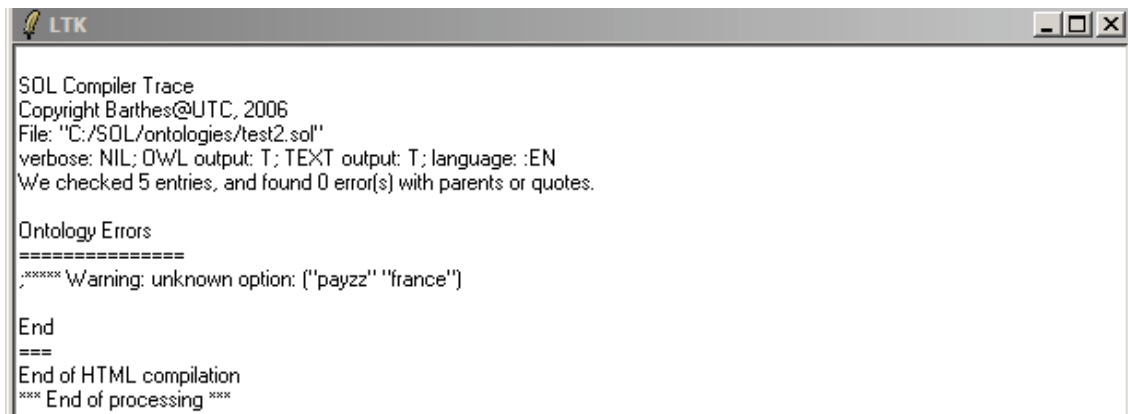
End
===
End of HTML compilation
**** End of processing ****

```

**Figure 16 - Unknown individual**

### 6.3 Unknown Property

An error may occur following the misspelling of a given property as shown in the next figure.



```

LTK
SQL Compiler Trace
Copyright Barthes@UTC, 2006
File: "C:/SOL/ontologies/test2.sol"
verbose: NIL; OWL output: T; TEXT output: T; language: :EN
We checked 5 entries, and found 0 error(s) with parents or quotes.

Ontology Errors
=====
***** Warning: unknown option: ("payzz" "france")

End
===
End of HTML compilation
**** End of processing ****

```

**Figure 17 - Property error**

## 7

## Fourth Test: Ontology Structure

The previous examples show some of the errors that can occur while defining the concepts and individuals of an ontology. The ontology however in the OWL format requires additional informations to be added in front of the OWL file. This is done with the defontology declaration as shown in the test3.sol file.

In addition an ontology may be complex. It is possible to structure it in terms of chapters and sections, like a book, by using the defchapter and defsection directives. The structure will be transparent for the OWL file but will appear in the HTML file.

Run the test3.sol example and check the organization of the HTML file.

**8****Fifth Test: Rules**

In any complex ontology one must limit the number of classes/concepts in order to simplify its management. The SOL formalism allows declaring virtual classes or virtual properties that will translate into rules to be run in the reasoning engine.

The two following rules define a virtual class “Adult” as a Person more than 18 years and a virtual relation “uncle” as the composition of brother and mother (another rule would handle the case brother of father).

```
(defvirtualconcept
  (:name :en "Adult" :fr "Adulte")
  (:is-a "person")
  (:def
    (?* "age" > 18))
  (:doc :en "An ADULT is a person over 18."))

(defvirtualrelation
  (:name :en "uncle")
  (:class "person")
  (:compose "brother" "mother")
  (:doc :en "an uncle is a person who is the brother of the
mother.")
)
```

In addition to the previous files, SOL outputs a file for rules: test4-jena.rules, with the Jena format.

```

// Rules expressed in Jena
@prefix terregov: http://www.utc.fr/~terregov#
@include <RDFS>

// An ADULT is a person over 18.

[Adult:
  (?* rdf:type terregov:Z-Adult)
  (terregov:Z-Adult rdf:type owl:Class)
  (terregov:Z-Adult rdfs:subClassOf terregov:Z-Person)
<-
  (?* rdf:type terregov:Z-Person)
  (?* age ?V9)
  greaterThan(?V9, 18)
]

// an uncle is a person who is the brother of the mother.

[Uncle:
  (terregov:hasUncle rdf:type owl:ObjectProperty)
  (?V10 terregov:hasUncle ?V12)
<-
  (?V10 mother ?V11)
  (?V11 brother ?V12)
]

```

**Figure 18 - Rule output with the Jena format**

The header of the rule file is produced by the

## 9

## Sixth Test: Full Example

A run with a full ontology can be done with the file `tg-ontology18.sol` that contains the main concepts and rules for the TerreGov project.

## 10

## Initialization File

The SOL suite includes a special file containing default options (Figure 19). The file is located in the **sol-tools** folder.

Possible options are:

### *Ontology Title*

The ontology title is used as a default in all the SOL tools. The value is superseded by the title of the defontology directive.

```
ontology.title = Terregov
```

## Ontology Languages

A list of pairs indicating the different languages used in the ontology. They will appear in the language choice of the SOL command panel.

```
ontology.languages = ("English" :en) ("FranÃ§ais" :fr) ("Italiano"
:it) ("Polski" :pl)
```

## Default Input File

The name of a file to appear as a default in the SOL command panel. The file should be located in the ontology folder in the SOL main folder.

```
ontology.input-file = test.sol
```

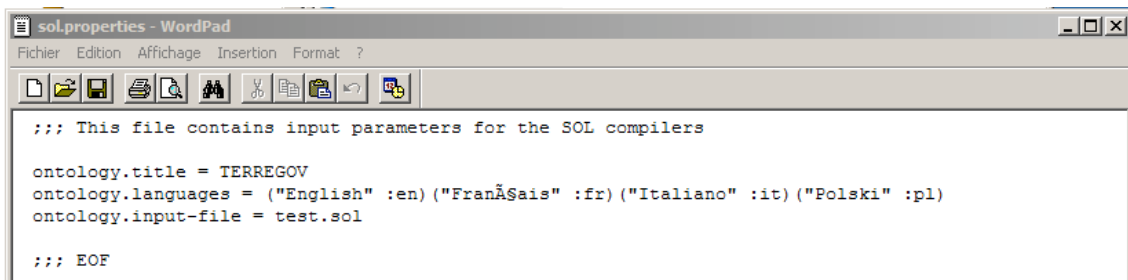


Figure 19 - sol.properties initializing file

## 11

## Possible Problems

### 11.1 The SOL command panel does not appear

You may have an incompatible version of the Windows system. TCL hangs up on the Windows 2000 system.

### 11.2 Unjustified parenthesis or quote error

A message may appear telling that there is a parenthesis or quote error, but visual inspection detects no such error. The error may come from the presence of an invisible character appended at the end of the SOL definition. A good candidate is a **tab**.

Remove the trailing invisible characters and relaunch the application.