Knowledge Processing Using Ontopia and Source Mage GNU/Linux - II

Ladislav Hagara

Department of Communication and Information Systems
Faculty of Military Technology, University of Defence, Brno, Czech Republic
ladislav.hagara@unob.cz

Abstract. *This paper extends the paper Knowledge Processing using Ontopia and Source Mage GNU/Linux published in Cybernetic Letters in 2009 [1]. The paper presents the Source Mage GNU/Linux Topic Maps Ontology and the possibilities of using it together with Ontopia to support development of Source Mage GNU/Linux. Source Mage GNU/Linux is a source-based GNU/Linux distribution. Topic Maps is a standard for the representation and interchange of knowledge, with an emphasis on the findability of information. Ontology is a formal representation of knowledge as a set of concepts within a domain, and the relationships between those concepts. Ontopia is a complete set of open source tools for building, maintaining, and deploying Topic Maps based applications.*

Keywords

Source Mage GNU/Linux, Topic Maps, Ontology, Ontopia, Onotoa, Visual Understanding Environment, LTM, TMRAP, Tolog, Open Source.

1. Source Mage GNU/Linux

Source Mage GNU/Linux is a source-based GNU/Linux distribution. It allows users the maximum amount of freedom and choice possible. Instead of delivering binaries to users, the source code is compiled. This method allows greater control over the software than precompiled distributions. Individual dependencies can be selected or deselected. For instance, OpenSSH can be compiled without support for X11. One can choose to set cflags, cxxflags, and ldflags specific to their situation. Using a source based distribution is the only way to unlock the full performance of a computer, as binary distributions must compile their software for a wide audience. When a Source Mage spell is "cast", the latest stable release is downloaded from the developer's site. Source Mage does not change anything in packages, so it is immune from the kind of errors resulting from distro developers tampering.

Users can build their own unique systems. Systems can be optimized for specific hardware and with specific options and can contain only necessary software. Users of source-based distributions as opposed to a binary-based can control what each package is compiled for. No needs to install unnecessary software.

Source Mage's tagline is: Linux so advanced, it may as well be magic. Basic terminology is similarily magic-oriented. Program packages are called spells, and a collection of spells is called a grimoire. All installed grimoires likewise make up the codex. Installing a spell is called casting, and removing it is called dispeling. Such terms are not only figures of speech, cast and dispel are also the programs used to perform the said tasks. To install a package you cast that spell. Casting a spell consists of downloading the source code (if it is not already downloaded), checking for dependencies, casting them if necessary, compiling the program, and installing it. To uninstall a package you dispel the spell. Today there are ever 7300 spells in the official codex. Source Mage package management program is called sorcery. There are also other sorcerous commands.

Source Mage users can see exactly what goes on behind the scenes. They can create custom-built Linux systems for their specific PC. Next, it ensures that their systems have only exactly what they want on them - nothing more, nothing less. They can very easily get up-to-date software. Because Source Mage team doesn't provide binaries, updating versions happens almost as new software releases are announced.

Fig. 1. Source Mage GNU/Linux Logo

2. Topic Maps Ontology

2.1 Topic Maps

Topic Maps is a standard for the representation and interchange of knowledge, with an emphasis on the findability of information. The ISO standard is formally known as ISO/IEC 13250:2003. A topic map represents information using

- **topics**, representing any concept, from people, countries, and organizations to software modules, individual files, and events,
- **associations**, representing hypergraph relationships between topics, and
- **occurrences** representing information resources relevant to a particular topic.

2.2 Ontology

In computer science and information science, an ontology is a formal representation of knowledge as a set of concepts within a domain, and the relationships between those concepts. It is used to reason about the entities within that domain, and may be used to describe the domain. In theory, an ontology is a "formal, explicit specification of a shared conceptualisation". An ontology provides a shared vocabulary, which can be used to model a domain - that is, the type of objects and/or concepts that exist, and their properties and relations.

3. Software for ontology design

For the initial draft of the ontology it is recommend to ignore all software products. We should focus only on ordinary paper, pencil and eraser. For the further improvement and especially for possible collaboration the tools for creating mind maps can be used. The Source Mage GNU/Linux Topic Maps Ontology was created, developed and discussed in Onotoa, Visual Understanding Environment (VUE) and Ontopia. These tools have been chosen because we can use them on both platform Microsoft Windows and GNU/Linux. These tools are implemented in programming language Java.

3.1 Onotoa

Onotoa [2] is an Eclipse-based ontology editor for Topic Maps. It has a graphical UML-like interface, an export function for the current TMCL-draft (Topic Maps Constraint Language) and a XTM2 (XML Topic Maps). The export function is powered by tinyTiM.

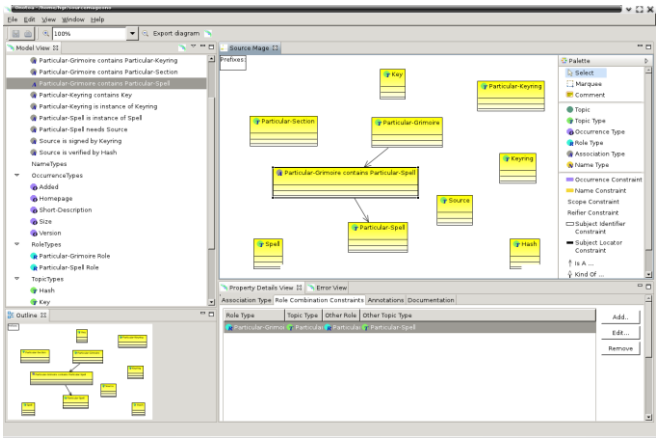

Fig. 2. Source Mage GNU/Linux Topic Maps Ontology in Onotoa

3.2 Visual Understanding Environment

The Visual Understanding Environment (VUE) [3] is an Open Source project based at Tufts University. The VUE project is focused on creating flexible tools for managing and integrating digital resources in support of teaching, learning and research. VUE provides a flexible visual environment for structuring, presenting, and sharing digital information.

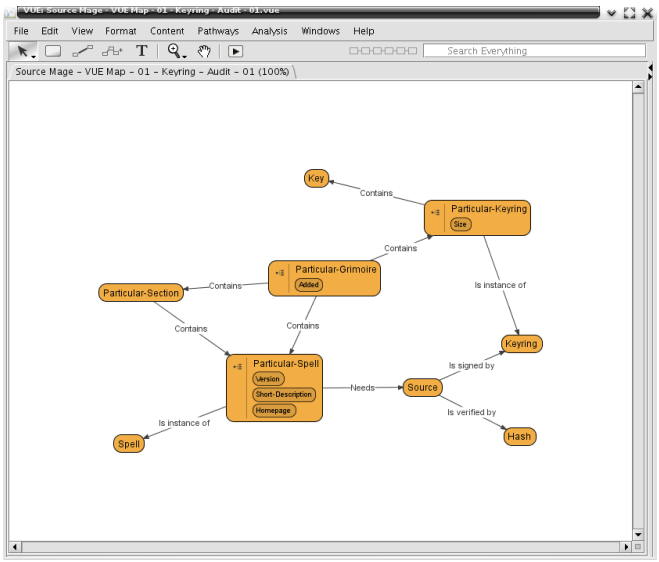

Fig. 3. Source Mage GNU/Linux Topic Maps Ontology in VUE

3.3 Ontopia

The Ontopia is a suite consisted of several interlaced applications. From the user's view there are Ontopoly (Topic Maps Editor), Omnigator (Topic Maps Browser) and Vizigator (Topic Maps Vizigator). Ontopoly is a self-configuring, ontology-driven Topic Maps editor. A topic map consists of both an ontology and an instance of that ontology; with Ontopoly, you can edit each of these. The ontology defines the rules for what can go into the instance, and these rules are used by Ontopoly to automatically

generate the interface for creating and maintaining the instance. Ontopoly is built as a client/server application. As a client, you use your web browser, while the server is a web server bundled with the distribution. The server-side application is built using the Navigator Framework and Web Editor Framework.

Fig. 4. Ontopia spell in Ontopoly

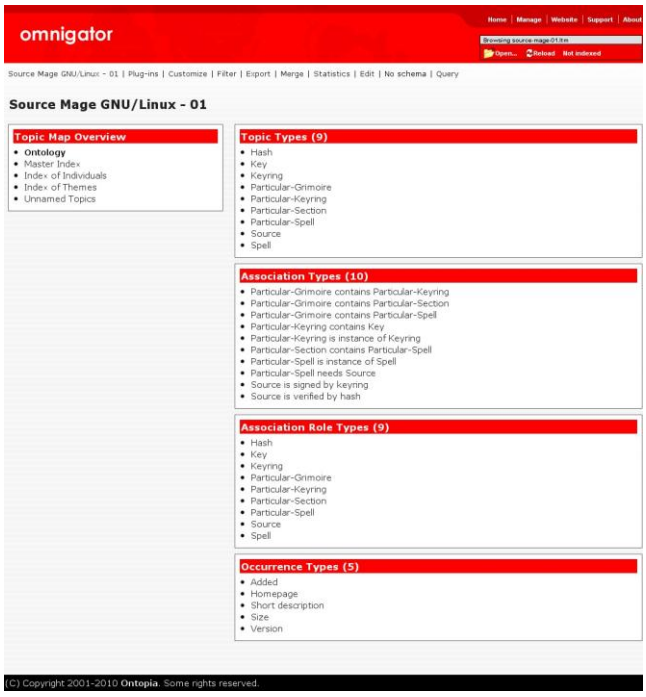

Fig. 5. Source Mage GNU/Linux Topic Maps Ontology in Omnigator

4. LTM

The whole Source Mage GNU/Linux Topic Maps Ontology is generated by BASH scripts. These scripts create text files describing topic maps in the Linear Topic Map notation (LTM). The Linear Topic Map notation is great for it. It is a simple textual format for topic maps. Just like XTM, the XML interchange format, it represents the constructs in the topic map standard as text, but unlike XTM it is compact and simple. The notation can be written in any text editor and processed by topic map software that supports it, or converted into the XML format supported by such software.

5. TMRAP

These Topic Maps and their fragments are sent to the Ontopia server by Topic Maps Remote Access Protocol (TMRAP). The TMRAP is a web service interface to Ontopia which makes it possible to retrieve Topic Maps fragments from a remote Topic Maps server and also to modify the topic maps stored on the server. The interface consists of a number of methods which can be accessed either using plain HTTP or using SOAP.

6. TOLOG

Tolog is a language for querying topic maps, inspired by Datalog (a subset of Prolog) and SQL. Using tolog one can query a topic map in much the same way as a relational database can be queried with SQL. It is possible to ask for all topics of a particular types, the names of all topics of a particular type in a particular scope, for all topics used as association role types, for all associations with more than two roles, and so on.

Example of Tolog Query (1):

How many computers does user lace have?

```
$ cat smgl-tolog
#!/usr/bin/python

import httplib, urllib

BASE = "http://localhost:8080/tmrap/tmrap/"
TOPICMAP = "smgl.xtm"

ltm = """
select $USER, count($COMPUTER) from
user-owns($COMPUTER : computer, $USER : user)?
"""

inf = urllib.urlopen(BASE + "get-tolog?topicmap=" + TOPICMAP +
"&tolog=" + urllib.quote(ltm))
print inf.read()
inf.close()
```

User lace has 2 computers:

```
$ ./smgl-tolog
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<result xmlns:x="http://www.topicmaps.org/xtm/1.0/"
xmlns:l="http://www.w3.org/1999/xlink">
  <head>
    <column>USER</column>
    <column>COMPUTER</column>
  </head>
  <row>
    <value>
      <x:topicRef l:href="file:/opt/ontopia/apache-tomcat/webapps/
omnigator/WEB-INF/topicmaps/smgl.xtm#lace"></x:topicRef>
    </value>
    <value>2</value>
  </row>
</result>
```

Source Mage GNU/Linux Topic Maps Ontology in LTM notation

```
#TOPICMAP~ source-mage-01
[source-mage-01 = "Source Mage GNU/Linux - 01"]

/* --- Topic Types and Occurences Types --- */

[particular-grimoire = "Particular-Grimoire"]
[added = "Added"]

[particular-section = "Particular-Section"]

[particular-spell = "Particular-Spell"]
[particular-spell-version = "Version"]
[particular-spell-short = "Short description"]
[particular-spell-website = "Homepage"]

[spell = "Spell"]

[particular-keyring = "Particular-Keyring"]
[particular-keyring-size = "Size"]

[keyring = "Keyring"]
[key = "Key"]

[source = "Source"]
[hash = "Hash"]

/* --- Association Types --- */

[particular-grimoire-contains-particular-section = "Particular-Grimoire contains Particular-Section"
                                                 = "Particular-Section comes from Particular-Grimoire" / particular-section]

[particular-grimoire-contains-particular-spell = "Particular-Grimoire contains Particular-Spell"
                                               = "Particular-Spell comes from Particular-Grimoire" / particular-spell]

[particular-grimoire-contains-particular-keyring = "Particular-Grimoire contains Particular-Keyring"
                                                 = "Particular-Keyring comes from Particular-Grimoire" / particular-keyring]

[particular-section-contains-particular-spell = "Particular-Section contains Particular-Spell"
                                              = "Particular-Spell comes from Particular-Section" / particular-spell]

[particular-spell-is-instance-of-spell = "Particular-Spell is instance of Spell"
                                       = "Spell occurs in Particular-Spell" / spell]

[particular-keyring-is-instance-of-keyring = "Particular-Keyring is instance of Keyring"
                                           = "Keyring occurs in Particular-Keyring" / keyring]

[particular-keyring-contains-key = "Particular-Keyring contains Key"
                                 = "Key is included in Particular-Keyring" / key]

[source-is-signed-by-keyring = "Source is signed by keyring"
                             = "Keyring is used to sign source" / keyring]

[source-is-verified-by-hash = "Source is verified by hash"
                            = "Hash is verifying source" / hash]

[particular-spell-needs-source = "Particular-Spell needs Source"
                               = "Source is needed by Particular-Spell" / source]
```

Example of Tolog Query (2):

Source tarballs are signed by gpg keyrings.
Which keyring is the most used?
How many source tarballs are signed by this particular keyring?

Fig. 6. Tolog query result in Omnigator. The most used keyring is gurus.gpg, it signs 1111 source tarballs.

It is not necessary to always enter all Tolog queries into Omnigator. These queries can be prepared and written in Ontopia installed files. Queries can be prepared by someone more experienced and ordinary users can use them passive. Of course they can also modify them. For example Tolog query "Find key" will find all the keys containing the string "Source Mage". To find all keys containing another string it is sufficient to override the "Source Mage" string.

Fig. 7. Prepared Tolog queries Find key and Find spell in Omnigator

To add these queries into Ontopia you have to edit two files in directory /opt/ontopia/apache-tomcat/webapps/ omnigator/plugins/query.

The file **tolog.jsp**.

If you open topic map named source-mage-01.ltm or postgresql-62801 in Omnigator in Query extension the Tolog queries Find spell and Find key will appear. Their internal names are exSpell and exKey.

```
<%
} else if (tmid.equals("source-mage-01.ltm")
|tmid.equals("postgresql-62801") ) {
%>
<option value="">Example queries:</option>
<option value="exSpell">Find spell</option>
<option value="exKey">Find key</option>
```

The file **query-samples-tolog.js**.

The queries exSpell and exKey are defined here.

```
} else if (exName == "exSpell") { // ----------------------------------
document.queryform.query.value =
'/* tolog query that returns all spells\n' +
' contained "crypto" string */\n\n' +
'import "http://psi.ontopia.net/tolog/string/" as str\n' +
'select $A from\n' +
'instance-of($A, spell),\n' +
'topic-name($A, $B),\n' +
'value($B, $VALUE),\n' +
'str:contains($VALUE,"crypto")\n' +
'?';

} else if (exName == "exKey") { // ----------------------------------
document.queryform.query.value =
'/* tolog query that returns all keys\n' +
' contained "Source Mage" string */\n\n' +
'import "http://psi.ontopia.net/tolog/string/" as str\n' +
'select $A from\n' +
'instance-of($A, key),\n' +
'topic-name($A, $B),\n' +
'value($B, $VALUE),\n' +
'str:contains($VALUE,"Source Mage")\n' +
'?';
}
```

7. Ontopia and PostgreSQL

The standard installation of Ontopia works with Topic Maps only in computer memory. For not so huge Topic Maps it is an advantage. All is working by default. Problems appear in case of power failure or non standard Ontopia stopping. The data stored only in computer memory are lost. Another problem arises in case of huge Topic Maps. When the Topic Maps is addressed it has to be copied from hard drive into computer memory. This can last minutes. The Topic Maps can be also bigger than computer memory. If this occurs the Topic Maps have to be put into relational database.

The RDBMS Backend Connector adds relational database persistence support to the Ontopia Topic Maps Engine. The persistence is transparent and users of the topic map interfaces don't have to take any additional steps in order to persist topic maps. Applications that uses the RDBMS Backend Connector must properly demarcate transactions and follow the general rules in a transactional system. This includes managing the life-cycle of Topic Maps stores and transactions.

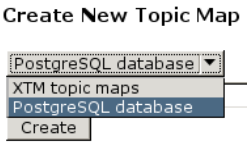

Fig. 8. Topic Map can be created in PostgreSQL

Using RDBMS Backend certainly help Ontopia administrator. Administrator can use to backup Ontopia Topic Maps the same tools as he regularly uses to backup another PostgeSQL based applications. He can use the same tools to monitor Ontopia or compare current Topic Maps version with previous one. For example he can use pgAdmin [5].

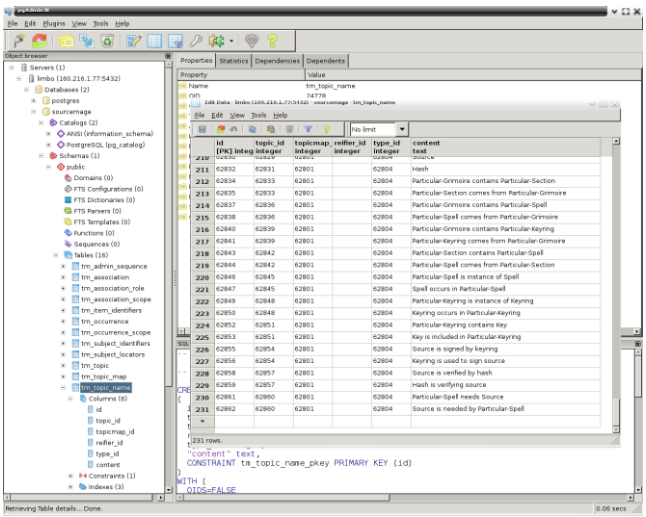

Fig. 9. Association Types in pgAdmin

We have to create database in PostgreSQL and set permissions for Ontopia to save Topic Maps in PostgreSQL.

```
# psql -U postgres
psql (8.4.4)
Type "help" for help.

postgres=# CREATE DATABASE sourcemage;
CREATE DATABASE
postgres=# CREATE USER smgl WITH PASSWORD 'smglontopia';
CREATE ROLE
postgres=# ALTER ROLE smgl LOGIN SUPERUSER INHERIT
CREATEDB CREATEROLE;
ALTER ROLE
```

The name of database, the user name and password must be saved in db.postgresql.props file.

```
# cat /opt/ontopia/apache-tomcat/common/classes/db.postgresql.props

net.ontopia.topicmaps.impl.rdbms.Database=postgresql
net.ontopia.topicmaps.impl.rdbms.ConnectionString=jdbc:postgresql:sour
cemage
net.ontopia.topicmaps.impl.rdbms.DriverClass=org.postgresql.Driver

net.ontopia.topicmaps.impl.rdbms.UserName=smgl
net.ontopia.topicmaps.impl.rdbms.Password=smglontopia

net.ontopia.topicmaps.impl.rdbms.ConnectionPool=true
```

Next step is to edit the file /opt/ontopia/apache-tomcat/common/classes/tm-sources.xml.

```
<source
class="net.ontopia.topicmaps.impl.rdbms.RDBMSTopicMapSource">
<param name="propertyFile" value="db.postgresql.props"/>
<param name="id" value="postgresql"/>
<param name="title" value="PostgreSQL database"/>
<param name="supportsCreate" value="true"/>
<param name="supportsDelete" value="true"/>
</source>
```

In database we have to create tables. The format of tables is described in the file postgresql.create.sql.

```
# psql -U postgres -d sourcemage -f
/opt/ontopia/rdbms/setup/postgresql.create.sql
```

Now we can test it.

```
# tests/runtests-rdbms.sh /opt/ontopia/apache-tomcat/common/classes/
db.postgresql.props

Ontopia Topic Maps Engine 5.1.0 (2010-06-28 08:31 by geir.gronmo)
Success: All required classes found.
Running tests
.....
Time: 25.749
OK (340 tests)
Asserts: 1062
```

The prepared ontology source-mage-01.ltm can be imported into PostgreSQL by the tool RDBMSImport. In Omnigator we see it as postgresql-62801.

```
# java net.ontopia.topicmaps.cmdlineutils.rdbms.RDBMSImport  \
/opt/ontopia/apache-tomcat/common/classes/ db.postgresql.props \
source-mage-01.ltm

Importing source-mage-01.ltm into M62801
Done. 122 ms.
```

8. CONCLUSION

Source Mage GNU/Linux Topic Maps Ontology can improve the development of Source Mage GNU/Linux distribution. Source Mage GNU/Linux Topic Maps Ontology helps developers to understand relations inside of Source Mage GNU/Linux. The Ontopia together with the Source Mage GNU/Linux Topic Maps Ontology described in the LTM format is presented. Topic Maps Ontology generation by BASH scripts, communication using the TMRAP protocol, interconnection between the Ontopia and the PostgreSQL and data mining in the Omnigator environment by using the Tolog query language is analyzed.

There are plans to extend this Topic Maps Ontology and create Topic Maps application based on Ontopia to improve the development and make Source Mage GNU/Linux even more reliable and secure.

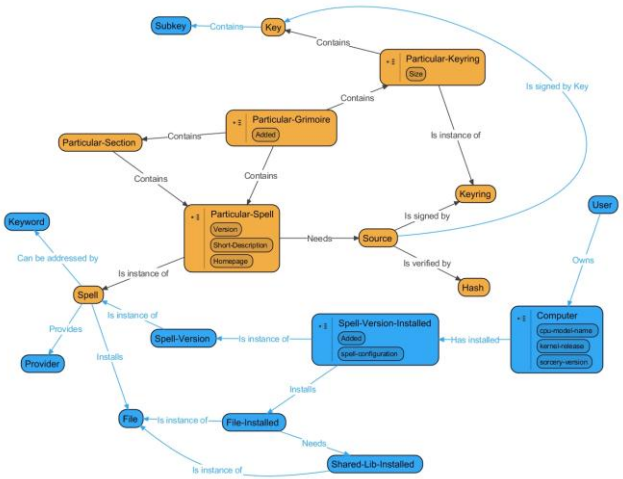

Fig. 10. WIP version of Source Mage GNU/Linux Topic Maps Ontology

References

[1] HAGARA, Ladislav. Knowledge Processing using Ontopia and Source Mage GNU/Linux. Cybernetic Letters. Special Issue, 2009, 7 pp. [Online]. [Cit. 2009-12-23]. ISSN 1802-3525.
Available from: http://www.cybletter.com/

[2] Onotoa, Eclipse-based ontology editor for Topic Maps.
Available from: http://onotoa.topicmapslab.de/

[3] Visual Understanding Environment (VUE).
Available from: http://vue.tufts.edu/

[4] Ontopia developers. Ontopia Documentation. Version 5.1.2, builddate 2010-10-30. [Online].
Available from: http://www.ontopia.net/doc/current/

[5] pgAdmin, administration and development platform for PostgreSQL
Available from: http://www.pgadmin.org/

About Author

Ladislav HAGARA is a Head of information systems and programming group of Communication and Information Systems Department of Faculty of Military Technology at University of Defence, Brno, Czech Republic. He has Master of Science degree in computer science from the Military Academy in Brno and Ph.D. degree from the University of Defense. His main fields of interest are operating systems, computer security and knowledge management. He is one of the lead developers of Source Mage GNU/Linux.