Towards a web-based centre on Swedish Language Technology

Petter KARLSTRÖM
Graduate School of Language Technology,
Göteborg University
Box 200
405 30, Göteborg, Sweden
petter.karlstrom@gslt.hum.gu.se

Robin COOPER
Graduate School of Language Technology,
Göteborg University
Box 200
405 30, Göteborg, Sweden
robin.cooper@gslt.hum.gu.se

Abstract

We present an upcoming web-based centre for information and documentation on language technology (LT) in Sweden and/or in Swedish. Some ways to collect and represent data using data harvesting and XML are suggested. Users providing data may choose one of four input modes, two of which will allow the user to edit existing documents at her own site. Thus we hope to facilitate data entry and inter-site communication, and reduce the amount of stale data. The entry modes use annotated HTML and an emerging XML-application, respectively.

In conclusion, we propose that an XML-application in the field of LT would facilitate for the LT community to share information, and that automatic data harvesting will emerge as an important technique for building information centres in the future.

Introduction

The website Slate is an information centre on language technology (LT) developed in Sweden and/or for the Swedish language. It is the technical/academic part of a collaborative effort between the National Graduate School of Language Technology in Sweden (GSLT) and Svenska språknämnden (the official organization for matters concerning the Swedish language). Språknämnden will provide the second part, containing popular information. Some material for the popular part will be taken over from the project Svenska.se at the Swedish Institute for Computer Science, SICS. The two parts will as far as possible have the same structure and cross-reference each other. We will inform on the following matters, in both areas:

- Research projects.
- Industrial projects.
- Educational programmes.
- Software.
- LT developed in Sweden for other languages than Swedish.
- Swedish organizations and individuals who have an interest in LT.
- Employment oppurtunities in the area of LT.

Information will be provided in both English and Swedish, depending on whether translated documents exist.

We cooperate via NorDokNet with centra in Denmark, Finland, Iceland and Norway. We also have contact with COLLATE at DFKI and follow their way of organizing information on LT in LT-World. Several more Language Technologists and LT-projects will contribute with information, notably the Ph. D. students in GSLT who have agreed to participate in software tests.

1 Rationale

Since the breakthrough of the Internet, and more specifically the World Wide Web (WWW, web), an information centre is commonly viewed as some sort of web-based service. In its most simple form, such a centre consists of a collection of links, manually updated by a webmaster or editor. Large, more sophisticated websites usually utilize some kind of database backend in order to separate content from display information, to facilitate searching and to provide simple, site-standardized means for data entry and editing.

One of the fundamental principles of the WWW is the idea of globally reachable documents, a fact given in the acronym WWW itself. Given its global nature, the WWW naturally grows to contain a very large amount of data. In order to resolve the issue of actually finding anything in this vast network, a need for search engines, portals, and information centres has arisen. Our website is a response to such a need for an information centre on language technology in Sweden.

The success of an information centre largely depend on the perceived quality of its contents, and how easily those contents are found. Textual quality is not what we will discuss in this paper. Suffice to say that quality depends on, among other things, information being correct, up to date, and comprehensive. For our purposes, this is a data entry/edit issue, which we will address.

1.1 Data entry

As stated above, a common way to manage website is by means of a database that generates HTML web-pages which a user may view in her web browser. The input to the database may be provided by other users, sometimes by means of a web-form (Fig 1).

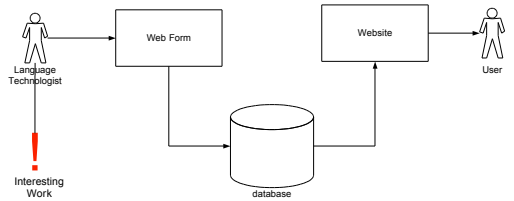

Fig. 1. An information centre/portal Arrows symbolize data-flow

Now, the Language Technologist in this picture is probably more interested in his Interesting Work than filling out a web-form (we hope!). He may already have filled out several such forms, may not see the use of the website in question, or may have done so already and be unwilling to update old information.

Therefore we suggest an alternative way to provide information (Fig 2). The alternative builds on the notion that people, projects, companies, etc. already have or want to have websites or homepages of their own. Most of the information is thus already available there. We will try to harvest this information by asking the data-providers to do one of four things:

a) Annotate their own HTML website with tags for our data-harvester.
b) Provide information in an XML-format, compatible with ours.
c) Submit a web-form.
d) Contact our web-editor for manual entry.

The last two entry modes are a step back to older methods, and are provided as a cautionary measure for two reasons. First, not all data-providers will adopt to the new way of data-entry. A cause for not adopting is that our XML-application and attributed HTML-tags will probably undergo several changes and demand that early adopters change their documents accordingly. The second reason is of pragmatic nature, namely that we need to build our site before the data-harvesting application and the XML-application is finished.

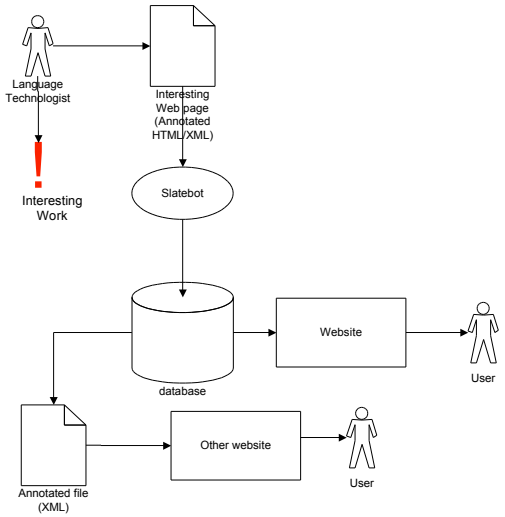

Fig 2. An information centre utilizing data harvesting. Arrows symbolize data-flow

There are three benefits with entry modes a) and b):

1. Data entry is facilitated by making use of existing web-pages.
2. The amount of stale data is reduced by updating the database as the data-provider updates her own website.
3. The ability to output in XML format. This will allow for other websites, for

example those in NorDokNet and LT-World to use their own harvesters on Slate's material.

2 Data harvesting

We will now describe what documents that will be harvested should look like, and how our harvester will work. Understanding this part of the paper requires some knowledge of HTML and XML

2.1 Document structure

In order for our harvesting engine to retrieve and store documents properly they need to conform to some standards. Contributions may be in HTML or XML. In the case of HTML the contributing site will need to annotate existing documents with tags that tell us what is what. We have chosen to use the META and SPAN tags since they may be used as containers for generic meta-information. Use of the META-tag is uncontroversial, since it is commonly used for providing data for search engines. The SPAN tag is more commonly used for style information, but may be used to tailor HTML to ones own needs and tastes, according to W3C (1999). Some may still see the use of this tag as controversial, since we use it for semantic, not style information. We argue that this is not an issue because the tags will not affect the website in question (presuming the information was already there), whether users implement a style-sheet or not. Alternatives might have been HTML-comments or scripting languages. The drawbacks with these methods are that they may force data-providers to use invalid HTML and that they make it harder to make use of existing information.

As for XML, documents will have to conform to a DTD under development by us in cooperation with NorDokNet. We also have contact with LT-World, in order to resolve compatibility issues with their information centre before they appear. Since websites using XML is not yet commonsight, documents will usually have to be written from scratch. No web browsers currently support XML satisfactorily, so the documents are not yet of much use outside of Slate. Therefore, this method is aimed towards early adopters who see the future benefits of XML and wish to participate in that development.

2.2 Examples

An HTML file that we should retrieve and file under people should contain this META-tag:

```
<meta http-equiv="slate"
content="people">
```

somewhere in its head. The document body may contain constructs like this:

```
I am a <span class="title">Ph.D.
student</span> in computational
linguistics at <span
class="affiliation">the Department
of Computer and Information Science
at Svedala University</span>.
```

Note that this is a real-life example (somewhat anonymized and edited). The Ph. D. student's existing web-page was annotated with SPAN-tags attributed to classes in the Slate database/XML-application. If the Ph. D. student changes affilitiation, he will probably want to update his own webpage. The updated information will be automatically retrieved by SlateBot.

XML files are more rigid since they follow a DTD. Their structure should not be surprising. Being XML-documents they contain a declaration and a few containers:

```
<?xml version="1.0" encoding="ISO-
8859-1" standalone="no" ?>
<!DOCTYPE people SYSTEM
"http://slate.gslt.hum.gu.se/people
/people.dtd">

<people>
<person>
<first_name>John</first_name>
<last_name>Doe</last_name>
…
</person>
</people>
```

The XML-compliant reader may note that our structure does not adhere to any standards such as RDF or OLAC. This will be attended to in later versions (see Discussion, below).

2.3 Harvesting engine

The core parts of our data harvesting engine, *SlateBot*, have been implemented and alpha tests have commenced using a very small database and XML-application listing people.

SlateBot is an application being developed in Perl, in order to acommodate for entry modes a) and b) above. The application consists of five main parts, making use of corresponding Perl-modules:

- HTML head-parser
- HTML parser
- Link extractor
- XML parser
- Database Interface

For input, it takes a list of links of participating sites. The list is maintained by our webmaster. For each entry in the list, it makes a HTTP GET request, just like a regular web-browser. It then checks whether the document in question is written in HTML or XML, and runs either the HTML parsers and the link extractor or the XML parser. The information returned from the parsers is then translated into XML and updated by the database interface.

The reason for having three different HTML parsers (including the link extractor) is, of course, that they do different things. The head parser only parses information in the HTML-HEAD part of the document, in order to check whether the document belongs in our database at all, and if so, in what main category. The HTML parser looks for SPAN tags, and returns those that correspond to our categories. Finally, the link extractor searches the document for links to other documents. This is provided for future development, in case we need our harvester to follow links in documents.

The XML parser is simpler, because we can assume that the XML file conforms to our DTD. We simply parse the file to find out which of our categories are implemented in the document, and send those to the database interface.

2.4 Tests

We have made some preliminary tests of the engine. Ph. D. students from GSLT and a few others were asked to provide information on themselves in either of three of the four entry-modes above. Web-forms were left out of the first tests, because we needed to ascertain that we could delete the database without consequenses.

The Ph. D. students at GSLT come from a wide range of backgrounds, so it was expected that some would be more interested in XML than others. Three of the entries we received were in the form of links to HTML-files and three were links to XML-files. One reply was in the form of an e-mail request for entry. Naturally, the information base is too small to draw any real conclusions concerning user preferences, but it does seem that we are headed in the right direction in providing the different modes.

The tests have helped us iron out a few of the unforeseen bugs that come with all software development, and we are now ready to enlarge the database model.

3 Other related projects and standards

We are not alone in realizing the benefits with meta-data. In fact, there are quite a lot of buzzwords and hype surrounding XML-development. These are some projects and standards that we should keep in mind, particularly with respect to data-harvesting and future development. Please note that this is in no way a complete list, due to the novelty of the field.

3.1 Semantic Web

The Semantic Web is an attempt to give information on the web meaning, in order to facilitate searching, automation, etc (W3C, 2001).

3.2 Resource Description Framework

The *Resource Description Framework*, RDF is a language for providing metadata to support the Semantic Web. It can be used for describing resources that can be located via a URI or other identifier (W3C, 2001). RDF is developed side by side with the Dublin Core (below), and both standards may be used in one document.

3.3 Dublin Core

The *Dublin Core Metadata Element Set* (Dublin Core, DC) is, as its name implies, a set of elements for metadata. There are fifteen elements, strictly defined using a set of ten attributes (DCMI, 1999). The elements' main uses are for information or service resources, e.g. bibliographies and card catalogs.

3.4 DocBook

DocBook is an SGML or XML format for technical documentation. It is intended for authoring, and can be converted to other formats for reading (Harold and Means, 2001).

3.5 Open Archives Initiative

The *Open Archives Initiative,* OAI is an experimental initiative for efficient dissemination of content. It uses the Dublin Core. Historically, the main intention of OAI was providing a meta-data language for e-prints, but this has been expanded to related domains as well. There is an OAI protocol for data-harvesting. Version 2.0 of that protocol is scheduled for release in June 2002 (OAI, 2001).

3.6 OLAC

OLAC, the *Open Language Archives Community* is a community who develop methods for digital archiving of language resources and a network for housing and accessing such services (OLAC, 2001). They use methods very similar to ours, though aimed at language in general rather than language technology. The Dublin Core forms the basis of their meta-data set. Alpha tests have commenced, and the project has recently been launched in Europe (May 2002).

3.7 Web services and SOAP

Web services is a way to share application methods over the Internet, by means of some standard interface such as *Simple Object Access Protocol,* SOAP (W3C, 2000). The methods implemented may for example provide access to a site's data.

4 Discussion and future development

As stated above, our tests were carried out with the help of a rather small group of subjects, and using a small database. In order for us to expand the database and the XML-application, we see a need for a more standardized XML for LT. A standard would be of use to all in the community, since it could be used for anything from printed matters (via translation to for example LaTex) to web pages and information centres such as ours. The standard would ideally be developed as a collaborative effort in the LT community, perhaps building on other successful standards such as DocBook or Dublin Core. Coordinating something of that magnitude is a rather large task, and not within the scope of our project. We will take the approach of developing some XML in NorDokNet, while keeping our eyes open towards the world, in particular stay in contact with LT-World. In non-formal discussions with representatives from DFKI, there is a growing interest in interchange of data and data-harvesting, and a belief that the work of OLAC could prove very useful.

In order to reach data-providers unwilling or unable to edit XML files, an editing tool would probably prove helpful. We will not develop such a tool in the immediate future, since other parts of Slate are more prioritized. With some luck, general XML-editing tools may appear on the market or better still as open source projects. In addition, browser support for XML (with style sheets) would probably encourage users. However, our control of browser development is, of course, nonexistent.

As seen above, there are other, larger projects and standards we should keep in mind and probably adapt to. The XML given in the examples should thus be treated as just examples to demonstrate the capabilities of SlateBot. Emerging global standards will be incorporated as consensus will dictate.

Conclusion

Data harvesting seems to be a feasible way of collecting information to an information centre, technically speaking. Our notion that different

data-providers will make use of different entry-modes seems to be true but need to be tested more. We believe that XML together with harvesting engines and other access methods can and will change the way we view the web, and that the LT community could and should take advantage of this.

Acknowledgements

Our thanks go to all other participants in NorDokNet and the Ph. D. students of GSLT for providing feedback and testing.

References

Note: all URIs to web-pages were tested on 01.07.2002. The documents may have been moved or removed after this date.

DCMI org., (1999) *Dublin Core Metadata Element Set, Version 1.1: Reference Description*, web-publication, http://dublincore.org/documents/dces/

Harold, E. R. and Means, S. W. (2001) *XML In a Nutshell,* O'Reilly & Associates

OAI org. (2001) *OAI FAQ,* web-publication, http://www.openarchives.org/documents/FAQ.html

OLAC org. (2001) *OLAC Metadata Set,* web-publication, http://www.language-archives.org/OLAC/olacms.html

W3C org. (1999) *HTML 4.01 Specification*, web-publication, http://www.w3c.org/TR/html401/

W3C org. (2000) *Simple Object Access Protocol (SOAP) 1.1*, web-publication, http://www.w3.org/TR/SOAP/

W3C org. (2001) *Semantic Web Activity Statement*, web-publication, http://www.w3.org/2001/sw/Activity

Other websites mentioned in the text

LT-World, http://www.lt-world.org/

NorDokNet, http://www.nordoknet.org/

Slate, http://slate.gslt.hum.gu.se/