Introduction and presentation of Content Syndication

Introduction to Content Exchange

Protocols and Standards

Existing Tools

Presentation of the case study

Online ressources and references

Practical Work

Cyberburo361 - March 2005

1. Introduction to Content Exchange

a. Necessity of Content Exchange

Internet, today, gives us the ability to communicate, exchange, access contents faster, further and better. Since its creation and adoption, everyday more people use and work with it and everyday more tools are developed in order to help this communication.

Internet has a great potential, but Internet itself is absolutely nothing but computers connected to each other and unable to communicate. It is for that reason common standards and protocols have been created like the IP protocol (Identification Protocol) that we use everyday now, or HTML (HyperText Markup Language) that allows us to share content today (The code of Websites is written in HTML)

As you can see, the first nature of Internet is to share and communicate, and the standards are always trying to improve these capabilities for home, business, government uses and more. The advantages of Content Exchange are multiple and one of them is the ability to share the same data between several unlinked sites.

Content Exchange is now a necessity and a reality, but the process of application may still appear difficult and needs to be well thought. Let's explore the actual solutions, built on durable standards.

b. Different Exchange methods

Before 1998 few companies tried to create closed standards, though they already had a strong interest in Content Exchange. But users and developers had to wait until 1998 for the arrival of the XML standard and the first opened exchange formats.

Two main companies tried to develop their own standards based on XML: Microsoft (with the Push technology – or Active Channel) and Netscape (with NetCaster). Both technologies knew a great development during 2 years, though the interest quickly declined shortly afterwards.

The two formats were similar to each other. Although the main problem was that the implementation was different and the stream content was diffused through JavaScript (that is a client Scripting technology).
As we said, the content structure was common on a few points and stored in separate files (CDF files).

During this time XML was beginning to emerge, and some people thought that they could use this new language to exchange data in a universal way.

c. Limitation of the solutions

The first limitation is the lack of interest of users when it was launched, as well as the fact that most of them were closed to the company's specific needs without considering the solutions' "open" aspect which is most of the time a necessity for a large adoption.

We can also consider that fact that Content Exchange wasn't in the mentality of developers and so applications weren't designed to share information or even use shared information.
Don't forget that Internet did not benefit, then, from a large access as presently.
Finally most of the internet application's servers weren't widely equipped with dynamic languages like PHP (that allows a streamed content to be modified and well presented in HTML).
Please note that Style Sheets where still not sufficiently developed or well adopted to correctly and easily modify a XML document.

It wasn't time for it.

2. Protocols and standards

a. Why use standards?

Internet is an extended network, with more computers than usual. Very well, after all, in a network, computers communicate between each other… more or less, because sometimes the network administrator has to upgrade a computer that is not compatible with the network protocols or internal tools.

On small networks (or administrated networks) you won't encounter big difficulties, but on a large networks opened to everybody; the communication is fragile and the balance easily folds.

Standards allow users to communicate between different platforms via protocols, protocols that respond to standards, even if these standards are closed. On the Internet the most well known standard is the HTML standard (that allows us to see websites thought different browsers and platforms with ascendant compatibility).

b. Standards make life easier for developers' and users'.

The main Internet Standard workgroup is W3C (World Wide Web Consentorium), It was created reasonably late, after the adoption of HTML that was developed and derived from an older and more complex language called: SGML, and they took back HTML and other languages development.

W3C is composed of members from different companies that have an interest in the Internet domain. They work together to improve actual standards and create future ones.

Standards are, most of the time, created by companies or university. They submit it to the W3C who make the choice of approval. But understand that W3C is not the ultimate regulation authority. They give advice and recommendations for developers, although other standards can be widely diffused without being approved by the W3C.

So standards are very important and are used more than we suspect in a computer's everyday life.

c. Which standard to use?

Back to Content Exchange, there is lots of "standards essay", but we will list 3 main standards: 2 for the content Syndication (i.e.: listing of news, threads, or documents) and 1 for content exchange.

The RSS (Really Simple Syndication) and the ICE (Information and Content Exchange) standards, but not validated by the W3C, are used to syndicate content. To exchange more complex content, XML is a better choice even if you can still use a form of the RSS or ICE; it's a choice of preference.

W3C has yet no finalized standard to purpose, but we assume that RSS could be a base of work for W3C because, as we will see later, RSS is itself based on W3C's XML standard, like ICE, but more opened than ICE is.

The main problem for a Decider is the choice between several solutions. The existence of standards helps us to reduce the choice and not recreate the wheel, but also help us to choose the one that is the closest to our needs.

d. Remember XML

Before beginning presenting the RSS and ICE standards, remember what XML is: a powerful semantic language, close to HTML and without pre-defined tags, except for the XML declaration code.

Let's have a look at the XML structure with this example file:

```
1:  <?xml version="1.0" encoding="utf-8" ?>
2:  <datas>
3:      <entry>
4:          <row id="34" type="text">Some text here</row>
5:          <date>21:35:45 10-06-2007 GMT+1</date>
6:      </entry>
7:
8:      <entry>
9:          <row id="36" type="text">Some other text</row>
10:         <date>08:03:29 12-07-2007 GMT+1</date>
11:     </entry>
12:
13:     <source location="Antibes" author="John Doe" />
14: </datas>
```

We will first see the structure of the document. As HTML, XML work with tags. Tags are not pre-defined except the ones in the first line that is a declaration tag (opened with <? and closed with ?>)

A new notion in XML is that tags that contain other tags are called "nodes", and sub-nodes for the other ones. Line 3 and 8, <entry> is a sub-node of the <datas> node.

XML obliges you to have at least one node, and then tags or sub-nodes. On the second line is <datas> which is a tag and a node, and its closed line 14. Closing tags are recognizable with the / (forward slash). XML is really strict on this point: tags have to be closed or the document is not valid.
Another way to close a tag that does not require content (like the one line 13) is to close the tag with the / before the >. In this case, and in this case only, closing tags are not needed. In addition, tags are case sensitive, that mean that <datas> is not the same than <Datas>.

Another element in this example is what we call "attributes". Line 4, 9 and 13, we see this attributes. For example, line 4: id="43" is an attribute: "id" is the attribute and "43" the value.
Some other content is what is between 2 tags. Line 4, 5, 9 and 10 include content. For example, line 10, between the <date> and </date> tag, the value "08:03:29 12-07-2007 GMT+1" is contained.

Finally, back to the definition line (line 1), the attribute "version" give us the version of XML employed, and "encoding" the character set family.

All this elements compose a XML document.

Some other standards can be added to XML, as DTD, XPath, XSL/XSLT, SOAP, the use of Namespaces, and more. Here is a quick reminder of these elements you already saw during the XML session but which you will need to remember for the upcoming ones.

- **DTD**: for Document Type Definition come from SGML. DTD is used to define the syntax of a document like:
    - tags that may or must be present in the document
    - how the tags are to be nested

- how often a given tag may be repeated in the document
- tag attributes and their default values
- all valid values for given attributes
- They may semantically define a group of data specifying the different possibilities allowed and when these are possible

DTDs are written in a specific language different of XML. You do not always need to create a new DTD for each document you create or generate. Most of the time, existing DTD are called in the XML page. However, a custom DTD can be useful with more complex applications like with some custom WebServices.[1]

- **XPath and XPointer**: are made for locating information within XML documents. XPointer is an addition to XPath that allow to perform more powerful query (designate an element with an id or keyword)

  The language used is different from XML.[2]

- **XSL and XSLT**[3]: are both made in order to change the appearance or the use of an XML document. XSLT is based on XSL and take back its structure. XSLT allow some of the following:
  - Change an XML document into an HTML document
  - Create another XML document with a different structure
  - Create a VML document (Vector Markup Language, create vectorised Pictures)[4]
  - Change a XML document into an audio document
  - Adapt the output of an XML document
- **SOAP**: is a sort of ancestor of what you'll see during the WebServices session. WebServices use an extended version of SOAP.

  SOAP work this way: thought an Internet protocol, you send a request to a web server. The web server answers you by sending the data asked for thought a XML stream. [5]

- **Namespace**: permit the use of elements with the same name to be used for different purposes and having different significations.[6]

---

[1] You can find a Reference to the DTD specifications in the "Online Resources" section of this document and in the XML session you already followed.

[2] You can find a Reference to the XPath specifications in the "Online Resources" section of this document

[3] XSLT's recommendations are included in the "Online Resources" section

[4] You won't need this during the different sessions you'll follow but you can find VML specifications in the "Online Resources" section

[5] You already had an approach of SOAP functionality during the XML session. More information are available in the "Online Resources" section

[6] You can find W3C recommendations about Namespaces in the "Online Resources" section

The power of XML is its structure. As you can see, except for the strict rules (nodes, case sensitive, etc.), XML has a completely free structure. You can use the tags you want, for the purpose you define and the needs you require, and choose to add the standards you need to it.

So XML can be a Content Exchange format, if you define correctly your common XML structure for sharing datas between 2 sites, if you agree on how include to elements, etc.

In fact, as you will see, XML is not a real standard of communication, but a standard communication mainframe.

XML, out of the box, can do nothing on its own. XML allows a liberty of choice in the tags, the user has more liberty in the process of programming, and this can cause a limited use of the language, as the content written in XML can be not understood by someone else, without analyzing the structure of the document and the meanings of the tags.

If you want to create your own data's stream in internal application, you can create your own standard with XML, but outside of the application, data will be hard to exploit.

That's why XML itself is not powerful enough to help separate organizations to share content, and this is where other standards, based on XML, are useful.

e. The RSS standard

The RSS (Really Simple Syndication) is actually maintained by Harvard University. This standard is very close to the CDF format we saw concerning the Push technology from Microsoft and NetCaster Technology from Netscape.

But first of all, let's explain what syndication is. Syndication is not really Content Sharing, but more "headlines" sharing, or "listing" sharing if you prefer. This is what we call a "stream" of datas that a Web server hosts and share with the al world. This stream of datas gives links to other contents (generally HTML pages).

This method is mostly used for news stream but can also be used in a similar situation for document's listing.

We described quickly XML structure; we'll now have a look to the RSS structure. RSS is actually version 2.0, and compatible with ancient versions (ascendant compatibility) that are: 0.91and 0.92. Ascendant compatibility is something required for a good standard and means that a reader for older versions will be able to read a document written with recent specification (by just ignoring tags the reader doesn't know). HTML, again, works with this principal.

We will study the following RSS document based on Reuters RSS stream:

```
1:    <?xml version="1.0" encoding="utf-8" ?>
2:    <rss version="2.0">
3:      <channel>
4:        <title>Reuters: Top News</title>
5:        <link>http://www.reuters.com</link>
6:        <description>Reuters: Top News</description>
7:        <image>
8:          <title>Reuters News</title>
9:          <width>120</width>
```

```
10:         <height>35</height>
11:         <link>http://www.reuters.com</link>
12:         <url>http://wwwi.reuters.com/comX/images/reuters120.gif</url>
13:       </image>
14:       <item>
15:         <title>Top new title</title>
16:         <guid isPermaLink="false">7753661</guid>
17:         <link>http://link.to.the.news</link>
18:         <pubDate>Mon, 23 Jul 2005 07:44:55 GMT+1</pubDate>
19:         <description>News description, can be very long or short,
there is no limitation.</description>
20:       </item>
21:       <item>
22:         <title>Another top news</title>
23:         <guid isPermaLink="false">7755469</guid>
24:         <link>htt://another.link.to.another.news</link>
25:         <pubDate>Mon, 23 Jul 2005 10:03:53 GMT+1</pubDate>
26:         <description>Some happy news here</description>
27:       </item>
28:     </channel>
29:   </rss>
```

As HTML, RSS has dedicated tags, we will describe them now. First of all, you can see the XML definition line (line 1), its look familiar. The second line is the rss node (i.e. the rss tag) which one includes the revision number of the rss standard (rss 2.0). Finally, the “<channel>” sub-node contains the rss content. We will now detail what’s in the <channel> sub-node that contains a sort of definition of the rss stream:

- Required <channel> elements
    - **title**: The name of the channel. It's how people refer to your service. If you have an HTML website that contains the same information as your RSS file, the title of your channel should be the same as the title of your website.
    - **link**: The URL to the HTML website corresponding to the channel.
    - **description**: Phrase or sentence describing the channel.
- Some optional <channel> elements[7]
    - **language**: The language the channel is written in. This allows aggregators to group all English language sites, for example, on a single page. The W3C provide a list of codes to use with countries. [8]

[7] A full listing of these elements are available on the RSS Standard Website, in the “Online Resources” section

[8] A link to this listing is available at the end of this document, in the “Online Resources” section

- **pubDate**: The publication date for the content in the channel. For example, the New York Times publishes on a daily basis, the publication date flips once every 24 hours. That's when the pubDate of the channel changes. All date-times in RSS conform to the Date and Time Specification of RFC 822, with the exception that the year may be expressed with two characters or four characters (four strongly recommended). [9]
- **image**: Specifies a GIF, JPEG or PNG image that can be displayed with the channel.

In the <channel> sub-node, line 14, you can see another element, the <item> sub-sub-node. This element contains the channel content and not the channel definition. Each <item> element contains one syndicated element that is defined with the following elements:

- Required <item> elements
  - **title**: The title of the item
  - **link**: The URL of the item
  - **description**: The item synopsis
- Some optional <item> elements[10]
  - **author**: Email address of the author of the item.
  - **pubDate**: Indicates when the item was published

We will analyze, at the end of this session, 3 different RSS stream to have a better look at the structure.

As you can see RSS is better adapted to share content listing than to share content. But it is possible with only one <item> sub-sub-node. However you have to take into consideration the real interest of this method.

f. The ICE standard

The ICE standard (Information and Content Exchange) is another form of Content Exchange, created by big companies like Boeing because of the need of a strong Content Exchange standard.

We won't describe here the structure of an ICE document; this is more complex and requires some programming skills.[11]

Another interesting point with the ICE standard is that syndication and content sharing are together part of the standard. Some security features are also included within this format as are other tools and functions needed by big companies that want to share confidential or secured content with their clients or sites.

ICE is based on XML and another standard you'll see during the Web Services session such as SOAP. Theoretically ICE is opened but the main problem with this standard is that it only serves some limited interests and does not respond to one of the main criteria of a standard: ascendant compatibility.

---

[9] A link to the RFC 822 norm is available in the "Online Resources" section

[10] A full listing of available optional elements is available on the RSS Standard Website, in the "Online Resources" section

[11] Some links to the ICE standard are given in the "Online Resources" section

The main example of this is the ICE new version (2.0) that is not readable by the ICE 1.0 or 1.1 readers. So the developers have to maintain two versions of the stream at the same time and upgrade progressively softwares compatibles with the new stream.

Secured transactions and business solutions have to look closely at it since it can be a powerful tool. But as you will see during the Web Services session, other solutions exist.

3. Existing Tools

There is today no existing “RSS maker” software, or ICE or XML. But several tools exist to edit with colorized code and multiple code validations tools.

None of the lists above are exhaustive

- **XML validators**
  - http://www.w3schools.com/dom/dom_validate.asp
  - http://www.stg.brown.edu/service/xmlvalid/
  - http://www.ltg.ed.ac.uk/~richard/xml-check.html
  - http://www.xml.com/pub/a/tools/ruwf/check.html
- **RSS validator**
  - http://rss.scripting.com/
- **RSS aggregator**
  - RSS bandit: http://www.rssbandit.org/ (Windows .net)
  - Straw: http://www.nongnu.org/straw/ (Linux Gnome)
  - Shrook: http://www.fondantfancies.com/apps/shrook/ (Macintosh MacOS)
  - NewsMonster: http://www.newsmonster.org/ (Windows, Linux, Macintosh – for Mozilla)
  - Other tools: http://blogs.law.harvard.edu/tech/directory/5/aggregators
- **Colorized editors**

  Please note that theses editors are only done in order to help developers to recognize quickly parts of code and do not produce any particular xml compliant code even if some have parsing error detectors.
  - Notepad++ (Shareware): http://notepad-plus.sourceforge.net/
  - conText (Freeware): http://context.cx/content/
  - PHPDesigner (Freeware): http://www.mpsoftware.dk/phpdesigner.php
  - HTML Gate (Freeware): http://www.mpsoftware.dk/htmlgate.php
  - XmEdil (Freeware): http://www.xmedil.com/ and http://www.freedownloadscenter.com/Programming/Editors/XmEdiL.html
  - SuperEdi (Freeware): http://www.wolosoft.com/en/superedi/

- eControl (Freeware): http://www.econtrol.ru/

4. Presentation of the case study

a. The project

The Weather Agency MWA (Mediterranean Weather Agency) would like to share real time information content with local Mediterranean agencies and publish news on their Internet Website.

After much deliberation with the MWA and local agencies, we have concluded that the RSS syndication protocol was well designed for this project. We have also decided to use the lasts version of RSS, the 2.0 standard.

The RSS stream would send content from a Database that will be filled by the MWA through a simple form application. An interface thought their Internet Website (with secured access) will also be available to authorize local agencies to insert news in the main stream.

The format of the RSS file will be: year-month-day.rss and stored by the MWA Web Server. Files of the day will be generated when needed (a cache system will be placed to improve performances) and older ones physically written on the server. The server will launch each day, at 0:00:00 AM, the process of generation of the last day stream.

The Internet website of the MWA will use this RSS stream to display the real time stream news and give access to each news bulletin.

Content of the news will be stored in an html page, without presentation elements using only titles html tags (<h1> to <h6>), paragraph elements will be in paragraph tags (<p>) and images will use classical <img… /> tags. The number of available tags will be limited by the html editor input interface that will allow anybody who has access to enter news in the stream.

Each item will be identified by the login of the writer in order to localize the information and detect any security issue.

b. Used tools

Used tools for this project will be:

WebServer: Linux Fedora Core 3 with Apache WebServer and Mono.net extension

Database: MySQL 4.0 with Mono.net connector

Development Tools: Mono.net compiler and development tool for both Web and Window applications (Mono offer compatibility with Windows, Linux and Macintosh)

Validators: The W3C validator for HTML website output and HTML content template. The RSS validator is used for the RSS stream.

c. Application Structure

We'll now have a quick overview of the application structure. We'll the main principles in order to better understand the meaning of the project.

The application structure is made to work this way:

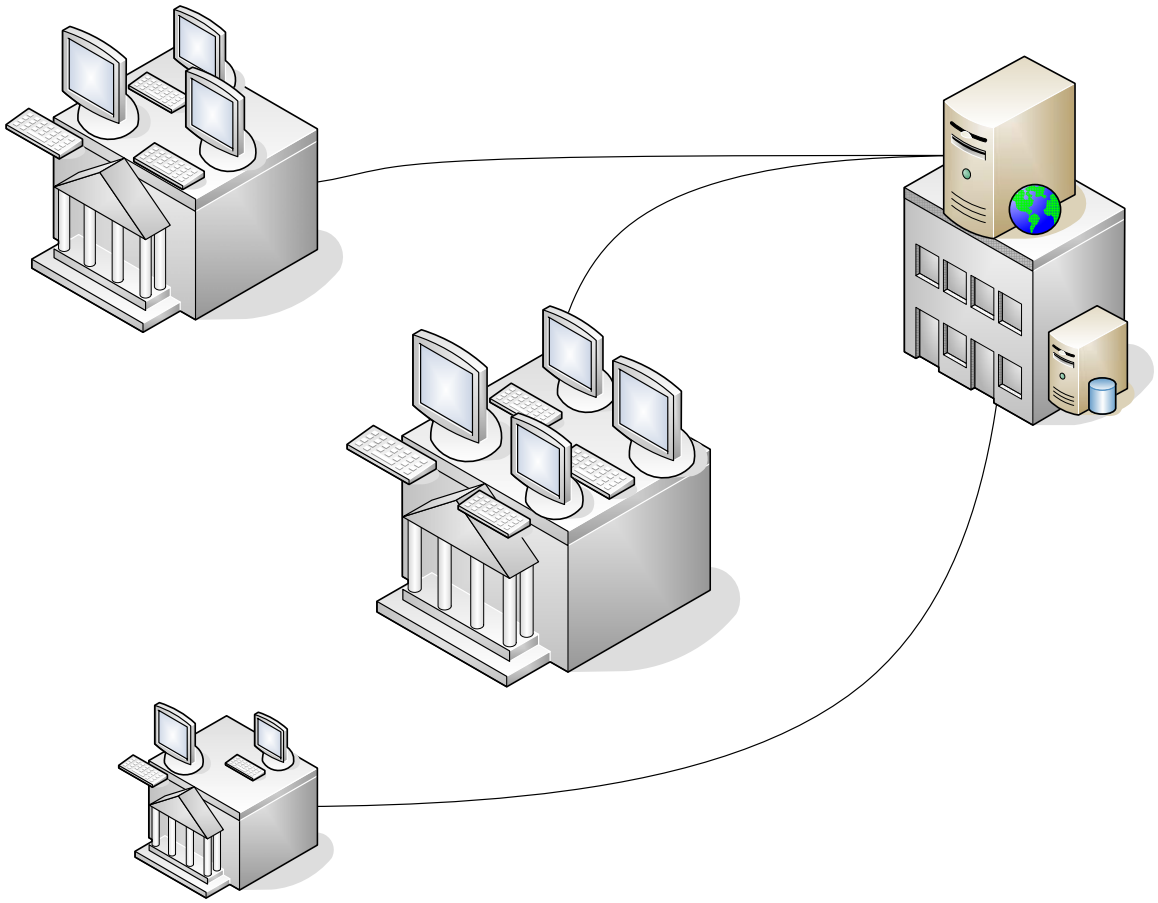

Fig 1

The main centre serves data to local sites. Data are centralized into the main centre.

The RSS stream content will be used by two different applications:

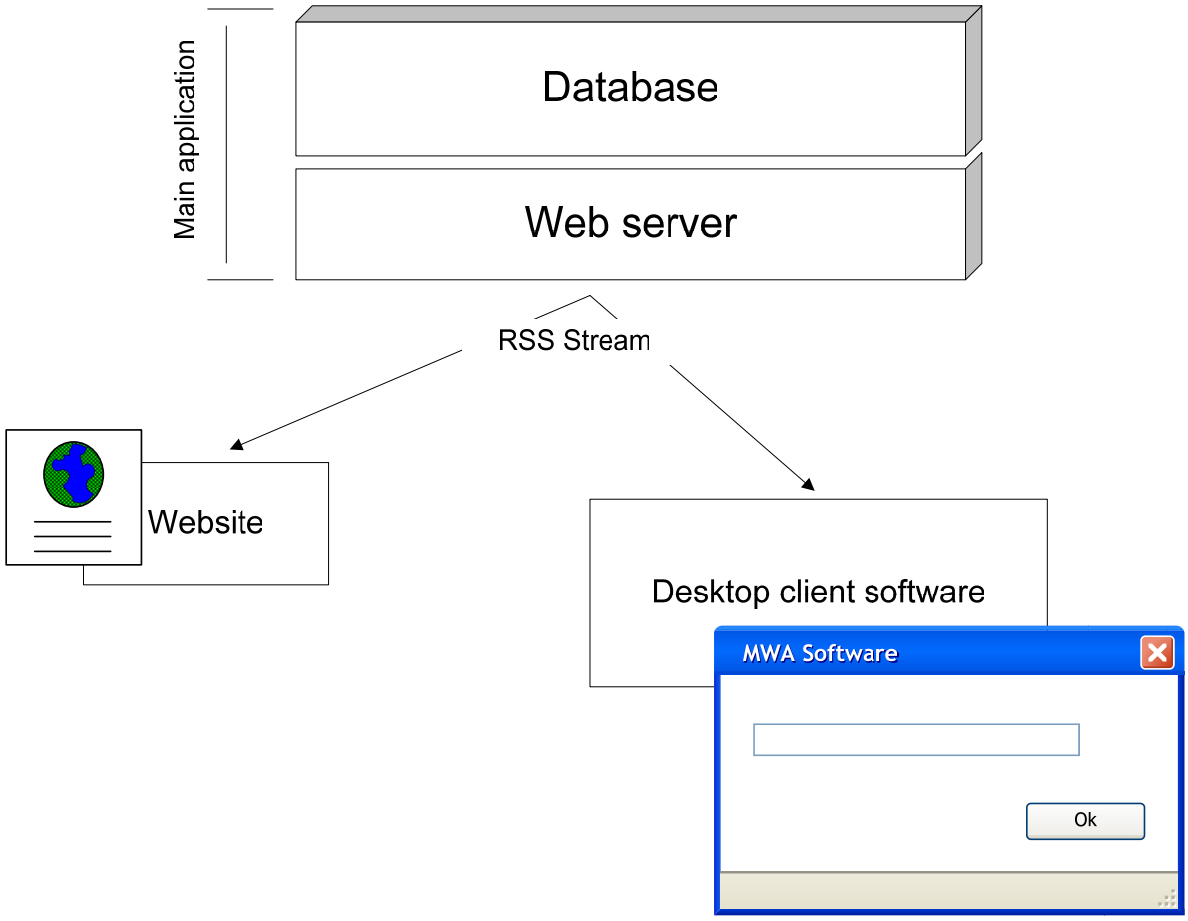

Fig 2

The Website will use the same stream than the desktop application. Both of them will have to translate the stream for their own need.

The Website will use a simple RSS translate program, and the Desktop application will use a cache system in addition:

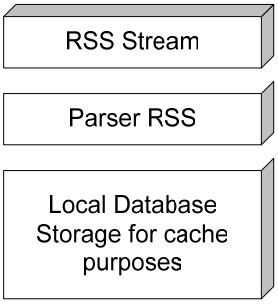

Fig 3

The cache system will store into databases the last RSS stream that has been transferred in order to prevent any system blackout in case of main server reboot or unavailability.

Finally, the application can work. When launched, the user chosen to access the News and then the listing appears. He has just to select the news that he want to access in order to have it.

Fig 4

The user never sees a piece of XML or RSS code in this application. Everything is transparent. We could even reformat the HTML output in order to display it the way we like in the application or integrate the browser into the application (in order to prevent the Operating system main Internet Browser opening for each news).

5. Questions & Answers

Please feel free to take notes or write your questions here.

6. Online Resources

XML

XML Presentation – June 2002 – EMWIS: http://www.emwis.org/documents/html/Rome_200020624.htm

W3C's official XML presentation: http://www.w3.org/XML/

W3C's XML specifications: http://www.w3.org/TR/REC-xml/

XML Recommandations: http://www.w3.org/TR/xmlbase/

W3C's DTD specifications: http://www.w3.org/XML/1998/06/xmlspec-report

W3C's XPath specifications: http://www.w3.org/TR/xpath

W3C's VML specifications: http://www.w3.org/TR/1998/NOTE-VML-19980513

W3C's XSLT specifications: http://www.w3.org/TR/1999/REC-xslt-19991116

W3C's WebServices and SOAP specifications: http://www.w3.org/2002/ws/

W3C's Namespaces recommendations: http://www.w3.org/TR/1999/REC-xml-names-19990114/

RSS

RSS official Website: http://blogs.law.harvard.edu/tech/rss

RSS Channel elements: http://blogs.law.harvard.edu/tech/rss#requiredChannelElements

RSS Items elements: http://blogs.law.harvard.edu/tech/rss#hrelementsOfLtitemgt

ICE

ICE official Website: http://www.icestandard.org/

ICE specifications : http://www.icestandard.org/specification/

General

The Web Developer's Journal: XML Content Syndication - http://www.webdevelopersjournal.com/articles/xml_syndication.html

XML.com: Inside the RSS validator - http://www.xml.com/lpt/a/2003/02/26/dive-into-xml.html

7. Supervised practical work

During this exercise, we will give a quick look to 3 different RSS news stream (or RSS feed). You will have to determine the RSS version employed, the different item used, and the type of news content.

- http://www.rnde.tm.fr/francais/Rnde_rss.xml
  - Version: ……
  - Tags used:
  - Type of content: ……..
- http://news.bbc.co.uk/rss/arabic/news/rss091.xml
  - Version: ……
  - Tags used:
  - Type of content: ……..

- http://www.actu-environnement.com/ae/news/archives/rss.php4
    - Version: ……
    - Tags used:
    - Type of content: ……..