Acta Cybernetica **18** (2007) 257–274.

Adding XMP support to Firefox

Péter Jeszenszky*

Abstract

XMP (Extensible Metadata Platform) is an RDF-based framework of Adobe Systems Incorporated that supports the embedding of metadata in application files. If it becomes widely used on the web, it will provide a rich source of metadata to semantic web applications, too.

This paper presents a solution to add a unique feature to the popular Firefox web browser, the capability to extract XMP metadata from web resources. The solution is based on the Piggy Bank Firefox extension that turns Firefox into a "semantic web browser".

Keywords: Semantic Web, XMP, Firefox, browser extension, Piggy Bank

1 Introduction

The Semantic Web is a vision that aims at ensuring that web content is machine processable. Many researchers believe that it will be the next logical step in the evolution of the web. If the vision comes true, that will enable us to implement more intelligent information services than at present.

Computer processing of web content is an extremely difficult task because most of the currently available web content is intended for human consumption. Although CSS (Cascading Style Sheets) makes it possible to separate content and presentation, and is now a very popular and widely used technique, the task of locating relevant information in an arbitrary web page is still hopelessly difficult, and should require true AI capabilities.

Although the semantic web is a vision and the current web is far from "being semantic", the underlying standards and technologies, that are still experimental in many cases, have enormous potential.

One of the cornerstones of the Semantic Web is RDF (Resource Description Framework) [23], a flexible and simple framework to represent knowledge on the web. RDF provides a data model to represent metadata about resources in a machine-processable form. A resource may be anything that can be identified by an URI (Uniform Resource Identifier).[1] Resources are described by statements that

*University of Debrecen, Faculty of Informatics, E-mail: jeszy@inf.unideb.hu

[1]These URIs are not used to retrieve the content of the identified resources. This means that URIs may be assigned even to physical objects.

are ordered triplets of the form subject-predicate-object, in which the subject is the described resource itself, the predicate is a property, and the object is a property value. The meaning of such a triple is that the resource has a property, which has a particular value. The standard also defines an XML syntax (RDF/XML) to represent RDF metadata.

In order to ensure that web content is machine processable, data must be available in RDF. That assumes the existence of machine readable web pages written in RDF. The extensive availability of RDF metadata is the main problem. Although certain semantic web applications, for example FOAF [9] are based on the use of such machine readable web pages, it is not likely that millions of users will publish RDF data on the web.

XHTML 2.0 [26] provides an elegant and easy-to-use annotation mechanism, called metadata attributes module to embed RDF in XHTML in such a way that does not put additional burden on web page authors. However, XHTML 2.0 is a work in progress, and it is unknown when it will become a stable standard and supported widely. Furthermore XHTML 2.0 is not backward-compatible with previous XHTML versions.

A radically different approach to obtain RDF metadata is based on the use of RDFizers. The term "RDFizer" was suggested in the SIMILE [24] project. Actually, *RDFizers* is the code-name of one of the SIMILE subprojects [20]. It is a collection of tools to transform data from various sources to RDF. A list of the currently available converters is maintained at the web page of the RDFizers project. These tools are either developed by participants of the SIMILE project or other contributors. For example, they can convert BibTEX files, mailbox files, MS Outlook files or Java bytecode to RDF. The result of a conversion must be a set of RDF triples that reflect the semantics of the underlying data. In the case of highly structured data the conversion is often natural, otherwise human intervention is required. RDFizers try to return as much information as possible with no human intervention, and at the same time keep it potentially useful for the processing applications.

Piggy Bank [19] is a unique Firefox [8] extension that uses a similar technique to obtain RDF data from web pages. (This is discussed later in Section 4.)

XMP [1] is an RDF-based framework that allows metadata embedding in application files providing a rich source of RDF metadata for semantic web applications.

This paper presents a solution based on Piggy Bank that offers a new web browser feature, i.e. the capability to extract embedded XMP metadata from the resources that are accessible from the current web page.

In Section 2 a brief overview of XMP is presented. Next, Section 3 reviews how XMP is supported for the time being. Section 4 introduces Piggy Bank that forms the basis for this work. Section 5 presents the new browser feature together with the implementation details. Finally, Section 6 discusses other alternative uses of the solution presented, showing that it may be utilized as a more general tool to obtain metadata about arbitrary resources.

2 XMP

2.1 What is XMP?

XMP is an RDF-based metadata framework developed by Adobe Systems Incorporated, that provides the following:

- a data model,
- a storage model,
- metadata schemas,
- rules that describe how to embed XMP metadata in various application files.

These together constitute XMP.

The data model makes it possible to associate properties with resources in order to describe them, similarly to the data model of RDF. A resource may be a file, or a portion of it, that may be meaningful to a processing application in itself, and that is also a distinct logical component of the file structure. (For example an image imported into a PDF file may be a resource. On the other hand, a range of pages in a PDF document can not be considered as a resource, since the PDF file format does not recognize such a structure.)

The storage model provides an implementation of the data model, it uses a subset of the RDF/XML [22] syntax to represent XMP metadata. Metadata describing a particular resource are serialized as RDF/XML and may be embedded in the resource itself in an XMP packet. The packet is a wrapper around the RDF/XML data that is surrounded by easily recognizable delimiters. An example XMP packet is shown in figure 1. The XMP specification [3] provides for the way of embedding of these packets in many common file formats, such as GIF, JPEG, PDF, PNG, PostScript, TIFF and others.

XMP schemas are sets of predefined metadata elements that can be used by various applications to characterize a wide range of resources, such as electronic documents, digital images, audio and video files.[2] Each schema is identified by a unique namespace and typically contains related metadata terms that can be used to describe resources of a particular kind, or to describe resources from a particular viewpoint. Several standard metadata schemas are provided as part of the XMP specification. For example, the Camera Raw schema defines properties that may be associated with image files produced by a digital camera.

2.2 Why is XMP important?

XMP has obvious advantages:

- It provides a standard and file-format-independent way to annotate digital images and other resources.

[2]Currently XMP does not use RDF schemas or ontologies to define properties. Standard schemas are provided in the form of tables in the specification.

- Embedded XMP metadata will be available to virtually every application. Even an application with no knowledge of the file format may scan a file for embedded XMP packets. (However, this is not recommended, as discussed later.)
- Metadata is shipped together with the embedding resource and thus cannot be lost.

It is not an exaggeration to say that XMP opens up exciting new possibilities for digital photography and image editing applications. If it will be widely used on the web, a rich source of metadata will be available to semantic web applications that may be utilized effectively.

```
<?xpacket begin="" id="W5M0MpCehiHzreSzNTczkc9d"?>
<x:xmpmeta xmlns:x="adobe:ns:meta/">
  <rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <rdf:Description rdf:about=""
        xmlns:dc="http://purl.org/dc/elements/1.1/">
      <dc:format>application/pdf</dc:format>
      <dc:title>
        <rdf:Alt>
          <rdf:li xml:lang="x-default">
            Test File
          </rdf:li>
        </rdf:Alt>
      </dc:title>
    </rdf:Description>
    <rdf:Description rdf:about=""
        xmlns:xap="http://ns.adobe.com/xap/1.0/">
      <xap:CreateDate>2006-12-10T10:00:00Z</xap:CreateDate>
      <xap:CreatorTool>
        pdfeTeX 3.141592-1.21a-2.2 (Web2C 7.5.4)
      </xap:CreatorTool>
    </rdf:Description>
    ...
  </rdf:RDF>
</x:xmpmeta>
<?xpacket end="r"?>
```

Figure 1: An XMP packet fragment.

3 Current XMP support

3.1 Commercial software

XMP is a flagship product of Adobe Systems Incorporated. Their software products, for example Adobe Acrobat, Adobe FrameMaker, Adobe GoLive, Adobe InDesign and Adobe Photoshop support XMP by default. To view or edit XMP metadata in an Adobe application the user must select the “File Info” dialog in the “File” menu. However, the author has experienced problems with Adobe Photoshop CS3 in the case of the PNG file format: XMP metadata can not be added to PNG files. This does not make any sense, since PNG is an open format.

3.2 Open source software

In respect of open source software, the situation is critical. Popular open source image and photo editing applications (for example GIMP) do not support XMP for the time being. Similarly, none of the currently available open source office applications (e.g. OpenOffice.org) can embed XMP metadata in the documents they produce.

The good news is that forthcoming versions of GIMP [10] will provide XMP support, making a significant step forward.[3]

Some of the few XMP-aware open source applications are mentioned here:

- The xmpincl [27] LaTeX package allows the embedding of an XMP packet in a PDF file that is produced by pdfLaTeX. The user must supply XMP metadata as an RDF/XML document. Producing such an RDF/XML document manually may be an extremely difficult task even for an experienced TeX user.
- PdfLicenseManager [18] is a Java application that focuses on licensing metadata. As the name suggests it supports only PDF files, and can display, insert and update Creative Commons licensing information that is stored in an embedded XMP packet.
- ExifTool [7] is a comprehensive set of Perl modules for reading and writing metadata in image, audio and video files. It runs on multiple platforms, and supports not only a wide range of file formats, but XMP also. A command line interface is available to the end-users. It is a great tool, although in the case of many formats it provides only read access to XMP metadata.

3.3 Developer support

Adobe XMP Toolkit (XMP SDK) [2] is Adobe’s official XMP library that allows developers to add XMP functionality to their application programs. It is intended to

[3]The next version will be released as GIMP version 2.4 and it is still under development. According to a post to the GIMP developer mailing list it is likely that the metadata editor feature with XMP support will be available only in version 2.6.

be cross-platform and runs on multiple operating systems, namely on UNIX/Linux, Windows and Macintosh.

The SDK currently consists of two distinct components:

- The XMPCore component provides an implementation of the XMP data model and allows applications to parse, manipulate and serialize XMP. It must be emphasized that XMPCore reads and writes RDF/XML only.
- The XMPFile component makes it possible to locate XMP packets in application files. The detected XMP metadata is returned to the processing application and then may be manipulated using XMPCore. XMPFile also allows applications to add XMP metadata to files and offers metadata update facility. It supports a rich set of file formats, including AVI, JPEG, MP3, MPEG, PostScript, TIFF and WAV.

The SDK is distributed in source code form. Both the XMPCore and XMPFile components are implemented in C++. A Java version of the XMPCore component is also provided. Note that the Java support offered by the SDK is partial only, since the XMPFile part is not implemented in Java.

Another major deficiency is that building the full SDK on UNIX/Linux systems is not supported, the XMPFiles part is compilable only on Windows and Macintosh. This platform dependency together with the lack of file format support in the case of Java are major problems that may prevent XMP becoming more widely used and popular.

The XMP SDK is open source software. Previous versions were distributed under a license called ADOBE SYSTEMS INCORPORATED – OPEN SOURCE LICENSE. However, this license is not included in the list of open source licenses maintained by the Open Source Initiative [17] and its compatibility with the GNU GPL license is questionable. This nonstandard license did not allow the SDK to be used in many open source projects. The issue generated debate in the developer community.

As an alternative to XMP SDK the project exempi [6] has been created by a developer. It is a C/C++ XMP library that is distributed under the GNU LGPL license. Unfortunately, it is still under development and does not support file formats at the moment. According to the web site of the project, version 2.0 will be a port of Adobe's XMP SDK to UNIX/Linux systems. JempBox [14] is another open source Java library that provides roughly the functionality of the XMPCore component of Adobe's XMP SDK, but offers no file format support.

Recently, Adobe has changed the licensing policy and the latest version of the SDK is now available under the terms of the widely used and accepted BSD License.

4 Piggy Bank

4.1 What is Piggy Bank?

Piggy Bank is a Firefox extension that turns the popular Firefox browser into a “semantic web browser”. It means that:

- It employs techniques and solutions providing a novel browsing experience.
- It utilizes semantic web technologies.

Piggy Bank is developed as an open source software in the SIMILE project, that is a collaboration of W3C, MIT Libraries and MIT Computer Science and Artificial Intelligence Laboratory (MIT CSAIL).

4.2 What is it good for?

Piggy Bank can extract “pure” information from a web page. Then the following options are available:

- Collected information can be saved and stored locally.
- Collected information can be uploaded onto communal information repositories called semantic banks. Semantic banks enable information to be shared with other people.
- Relevant information can be filtered out of the collected information flexibly.
- Novel display methods are available to visualize the collected information. For example, it is possible to display collected information items on a geographic map or on a timeline. The graph structure of collected information can be examined as well.

4.3 How does it work?

If Piggy Bank can extract “pure” information from the current web page, it is indicated by a small “data-coin” icon on the status bar of Firefox. Clicking on the icon will instruct Piggy Bank to collect the available “pure” information. Actually, RDF data will be harvested during the processing of the web page.

Piggy Bank can extract RDF from a web page in the following two cases:

- If the HEAD section of the HTML document contains LINK elements to RDF data, that is either in RDF/XML or in N3 format.
- The URL of the web page matches the URL pattern of an active screen scraper.

A screen scraper is a special JavaScript code that can transform the content of particular web pages to RDF. A screen scraper has an URL pattern. Web pages having an URL that matches that pattern can be processed by the scraper. Typically, the HTML structure of the web pages offered by a given web site is hardwired into a screen scraper that will enable it to extract all relevant information from a page. A screen scraper not only can process the current web page, but it may also load and process related web pages, or invoke a web service to obtain auxiliary information to enrich the content.

Screen scrapers must be installed in Piggy Bank in order to operate. A few screen scrapers are available at the Piggy Bank web site to collect pure information from popular web sites, such as Flickr or the ACM Portal. Another option is to write an own scraper. Solvent [25] is another Firefox extension that is also developed in the SIMILE project. It is a great tool that makes it easier to develop screen scrapers.

Piggy bank uses Longwell [15] to display the collected information, which is web-based faceted RDF browser.

5 XMP in Firefox

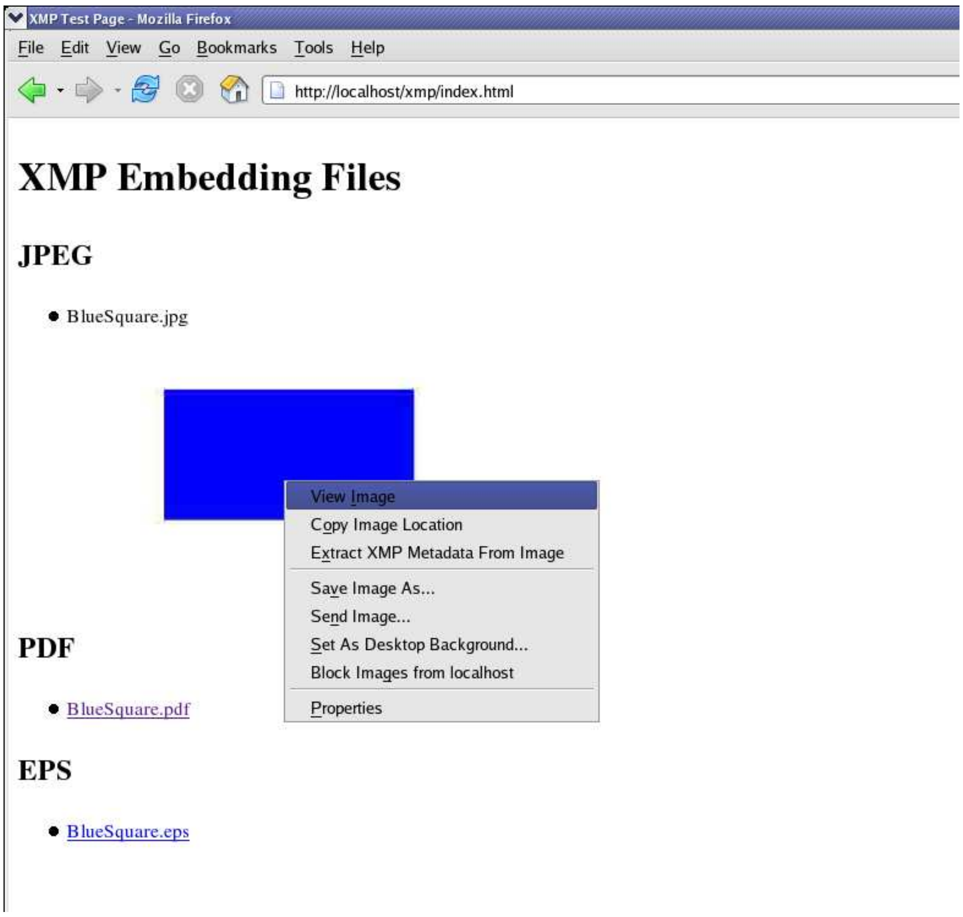
5.1 The main idea

Since the author is an enthusiastic fan of both Firefox and metadata standards including XMP, it was a fairly obvious, and at the same time a very desirable goal for him to add XMP support to his favourite web browser.

A Firefox extension is presented here that provides the following functionality:

- All the embedded XMP metadata can be extracted from the images and the resources that are linked to current web page with one click for browsing.
- XMP metadata can be extracted for browsing from individual images and links, too.

To the best of our knowledge none of the currently available web browsers offer such a function, thus it is a completely new browser feature. It is probable that such a functionality should also play an important role in the popularization of XMP.


XMP Test Page - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/xmp/index.html
View Image
Copy Image Location
Extract XMP Metadata From Image
Save Image As...
Send Image...
Set As Desktop Background...
Block Images from localhost
Properties

XMP Embedding Files

JPEG

- BlueSquare.jpg

PDF

- BlueSquare.pdf

EPS

- BlueSquare.eps

Figure 2: Right clicking on an image in Firefox. Select "Extract XMP Metadata From Image" to view embedded XMP metadata.

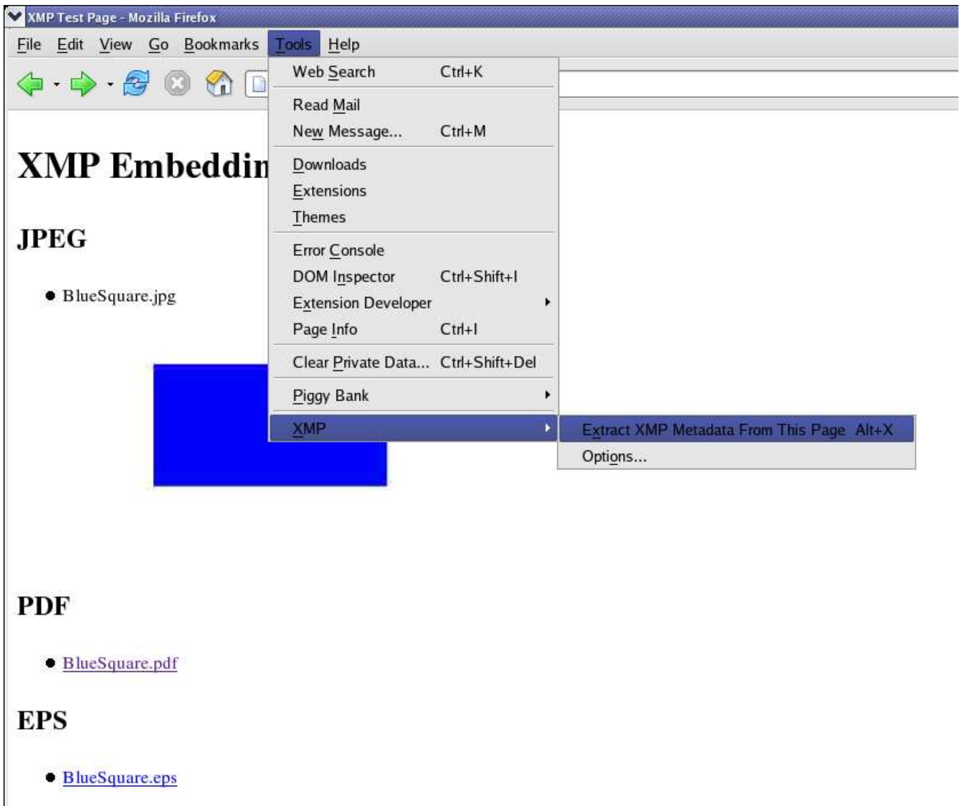
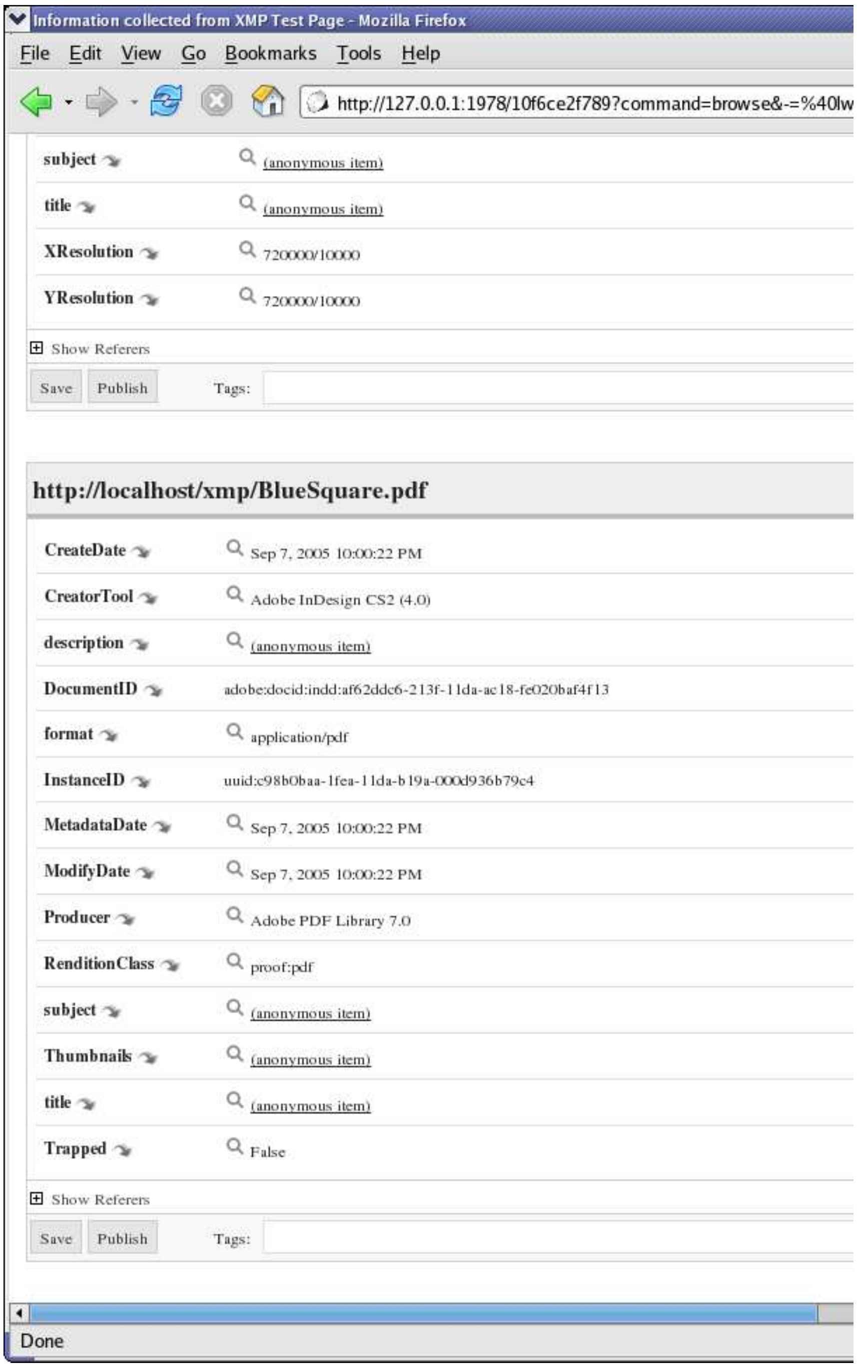


Figure 3: The XMP menu in Firefox.


Information collected from XMP Test Page - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://127.0.0.1:1978/10f6ce2f789?command=browse&-=%40lw
Show Referers
Save
Publish
Tags:
Done

subject	(anonymous item)
title	(anonymous item)
XResolution	720000/10000
YResolution	720000/10000

http://localhost/xmp/BlueSquare.pdf

CreateDate	Sep 7, 2005 10:00:22 PM
CreatorTool	Adobe InDesign CS2 (4.0)
description	(anonymous item)
DocumentID	adobe:docid:indd:af62ddc6-213f-11da-ac18-fe020baf4f13
format	application/pdf
InstanceID	uuid:c98b0baa-1fea-11da-b19a-000d936b79c4
MetadataDate	Sep 7, 2005 10:00:22 PM
ModifyDate	Sep 7, 2005 10:00:22 PM
Producer	Adobe PDF Library 7.0
RenditionClass	proof:pdf
subject	(anonymous item)
Thumbnails	(anonymous item)
title	(anonymous item)
Trapped	False

Figure 4: Browsing the extracted XMP metadata in Firefox.

5.2 Implementation

As the above functionality fits well into Piggy Bank, it has been chosen to serve as a basis for the implementation. The author has extended Piggy Bank with XMP support.

The implementation consists of two distinct parts:

- XMP extraction functionality that has been developed independently of the browser, as discussed below.
- An user interface part that integrates XMP functionality with the web browser.

The user interface of Piggy Bank is written in XUL [28]. XUL (XML User Interface Language) is a cross-platform user interface language developed by the Mozilla [16] project, that is used primarily in Mozilla applications and web browsers that are also based upon on the Gecko layout engine. The user interface of extensions, and also the entire user interface of Firefox itself is written in XUL. XUL makes use of many standards and techniques, including CSS, JavaScript and RDF. The user interface part of our implementation is also written in XUL.

Only few additional user interface elements were necessary to be added to the extension. Notably an XMP menu (containing the "Extract XMP Metadata From This Page" and "XMP Options" items) has been added to the "Tools" menu (see figure 3), an XMP options dialog has been created (see figure 6) and two appropriate items ("Extract XMP Metadata From Image", "Extract XMP Metadata From Link") have been added to the context popup menu that appears when the user right clicks on an image or a link (see figure 2). Figure 4 shows the final result (extracted XMP metadata) in the browser.

Currently, XMP extraction functionality is available to Piggy Bank as a RESTful web service. The web service accepts an URL in a HTTP [11] GET request and extracts XMP metadata that is returned as an RDF/XML document. This is a convenient solution which completely separates XMP logic from Piggy Bank code. Since screen scrapers may also utilize web services, this approach fits well with Piggy Bank, too. A great advantage of the solution is that it does not require XMP software to be installed on the client side, implementation details are hidden behind the web service.

The XMP extractor web service operates as follows:

1. The web service gets an URL in a HTTP GET request. XMP metadata will be extracted from the resource identified by the URL.
2. The web service initiates a HTTP HEAD request using the URL to query the MIME type and the length of the resource.
    - If the resource is not found or the XMP extractor web service does not support its format determined by the MIME type, then an appropriate response is returned, indicating the error.

- It is also an error if the content length of the resource exceeds a configurable limit.

3. The web service begins to retrieve the content of the resource and scans it for embedded XMP metadata.

    - If the XMP packet is found, it is postprocessed then returned as an RDF/XML document.
    - If XMP metadata is not found, an appropriate response is returned.

A schematic view of the above scenario is also shown in figure 5.

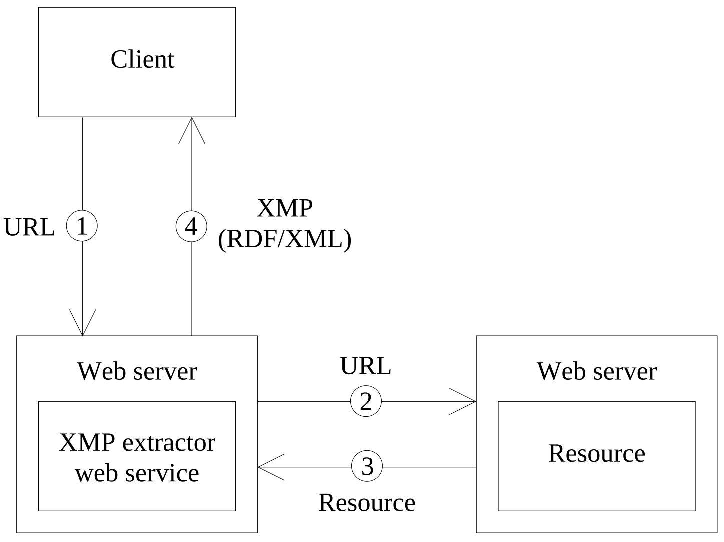
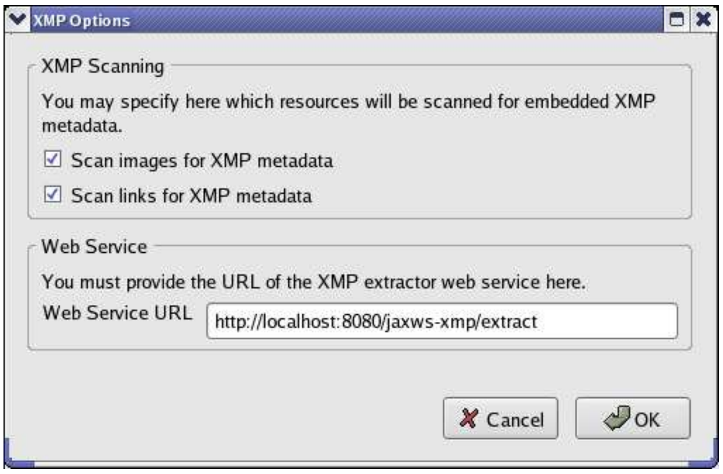


Figure 5: XMP extractor web service.

The web service is implemented in Java using the JAX-WS API [12, 13] and deployed to Apache Tomcat [4]. Java was a natural choice, as it is the "native language" of the author. Moreover, it is not tied to any particular vendor's operating system or platform.

To extract XMP metadata the author did not make use of Adobe's XMP SDK at all. The main reason lies in the previously mentioned licensing problems of the SDK that had existed when this work was done originally and also in portability problems. Since then problems have been solved only partially. Although the license has been changed, the XMPFiles part of the SDK that can perform XMP extraction is still not available on UNIX/Linux systems, as mentioned earlier in subsection 3.3. The author found this platform dependency annoying, as he prefers platform-independent solutions (he also prefers Linux).

Although there is a platform-independent Java version of the XMPCore component, it may be used only to parse the extracted XMP metadata. However, that is



XMP Scanning

You may specify here which resources will be scanned for embedded XMP metadata.

☑ Scan images for XMP metadata

☑ Scan links for XMP metadata

Web Service

You must provide the URL of the XMP extractor web service here.

Web Service URL: http://localhost:8080/jaxws-xmp/extract

Figure 6: The XMP Options dialog in Firefox.

unnecessary. If an XMP packet is available in RDF/XML, it can be fed into Piggy Bank directly after a postprocessing step is applied on it.

As a workaround, the web service utilizes a proprietary Java class library that has been developed by the author to extract XMP metadata from files (see figure 7).

The ExtractorFactory class and the Extractor interface makes up the core of the API. In the following, a software module that is used to extract XMP metadata from a file is called an extractor. Our implementation uses format specific extractors, as does the XMPFile component of Adobe's XMP SDK also.

Even an application that is not aware of the file format may scan a file for XMP metadata, since the task is to detect special XMP packet delimiter character sequences. However, this is not recommended, and should be considered only as a last resort, because:

- XMP packet delimiters may be encoded as UTF-8, UTF-16 or UTF-32, and the byte order may also vary between big-endian and little-endian. This implies that old and tried pattern matching algorithms can not be used directly. (It is unknown which byte sequence will represent a specific character.)

- XMP metadata may be stored in the embedding file in compressed form. In that case, an application that is not aware of the file format will not be able to detect XMP packets.

- If there are multiple packets, the application will not be able to determine which one is the main. For example, the incremental save feature of the PDF file format allows applications to update XMP metadata without overwriting, keeping the obsolete XMP packet.

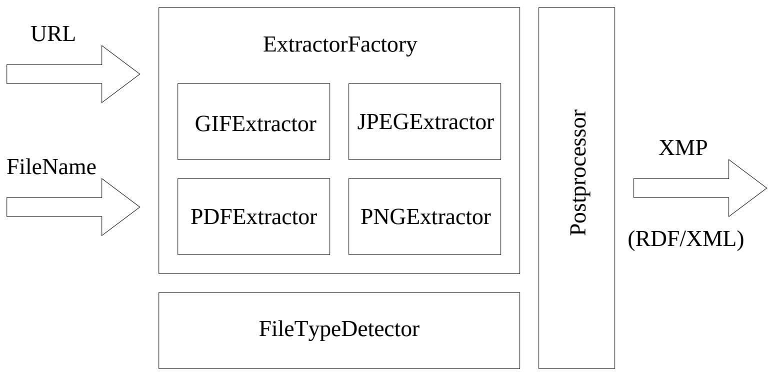


Figure 7: Java classes for XMP extraction.

- The XMP specification defines the way of embedding in the case of many popular file formats. Knowing the file format may help to locate XMP metadata more efficiently, without scanning the entire file for special delimiters. (The file format may determine possible locations.) This is very important, especially in the case of large files.

Thus the only reliable and efficient way to locate XMP metadata in a file is to use a processing application that recognizes the file format.

The ExtractorFactory class manages the available extractors. Currently the GIF, JPEG, PDF and PNG file formats are supported. (The development of a PostScript extractor is already in progress.) Each of these extractors is “smart” enough to know where to look for XMP metadata and so it does not scan the entire file.

Additional extractors can be plugged-in as well. When an URL or a file name is passed to the ExtractorFactory class, it returns an extractor object that will be able to perform XMP extraction. The ExtractorFactory class tries to determine the file format, then selects the appropriate extractor class.

The previously mentioned postprocessing step is not explained in detail because of its low-level technical nature. The result of XMP extraction is an XMP packet, which in fact is an XML document fragment enclosed by special delimiters. During postprocessing the XMP packet must be modified in order to get an appropriate and meaningful result, that may be processed by an RDF application. For example, lines 1, 2, 25 and 26 in figure 1 must be removed, otherwise traditional (non XMP-aware) RDF parsers may not be able to process the input as valid RDF/XML.

6 Generalization

As already mentioned in Section 1, RDFizers are tools to transform various data sources to RDF, thus making them available to semantic web applications. It should be noted that the XMP extractor web service is actually an RDFizing web service. It is obvious that the web service may be replaced by any other RDFizing web service that accepts an URL and transforms the identified resource to RDF. The result is viewable in the browser just like XMP metadata.

The author is currently experimenting with an RDFizing web service that may be regarded as a generalization of the original one. It will continue to provide XMP extraction functionality but will be able to handle a much broader range of resources. In the new web service the ExtractorFactory class is replaced by a class named RDFizerFactory that maintains a list of available RDFizers. Each RDFizer handles a specific file format and transforms resources of that type to RDF. The web service accepts an URL, determines the MIME type of the resource, then selects the appropriate RDFizer to transform it to RDF.

The list of available RDFizers utilized by the web service contains the original XMP extractors and a few additional RDFizers that are also developed by the author [21]. Namely, an RDFizer that handles the RPM package file format used by many Linux distributions, and another one that handles torrent files used by the BitTorrent P2P file sharing system. Using the RDFizer web service the user may explore RPM package information (for example description, required dependencies, list of provided files) or torrent file metadata in the browser, by simply right clicking on a link that points to an RPM package or a torrent file.

The developers of Piggy Bank may be working on something similar. When the author mentioned the XMP extractor web service to the Piggy Bank developers on a mailing list they said that they also had a web service called Babel [5] that is now publicly available and that can convert between various formats, including RDF. At the moment just a few formats are supported, but we think it would be easy to extend it to support many other RDFizers also.

Although Piggy Bank collects information from web pages utilizing scrapers that may also use an RDFizer web service, the currently available scrapers do not exploit this opportunity. These scrapers return RDF by processing only the text of web pages, locating relevant information. The novelty of the work presented in this paper is that it demonstrates that Piggy Bank may also be used to obtain RDF from non-textual resources (XMP embedded in binary files).

7 Conclusions

The paper introduces a new feature that works with the Firefox web browser, the ability to display XMP metadata embedded in resources that are accessible from the current web page.

The implementation is based on the Piggy Bank Firefox browser extension, that turns Firefox into a “semantic web browser”, and also employs a new web service.

The web service accepts an URL and scans the identified resource for embedded XMP metadata that is returned. Using the presented browser extension, images and hyperlinks can be processed by the web service with a single mouse click, and returned XMP metadata is viewable in Piggy Bank. The novelty of the presented work is that it uses Piggy Bank to obtain RDF from non-textual resources (for example, from images on a web page).

Although work presented in this paper may appear to be a small step towards the semantic web, it has resulted in a quite general tool that can be used right now, and may be of interest to a wider audience.

References

[1] Adobe XMP
`http://www.adobe.com/products/xmp/`

[2] Adobe XMP SDK
`http://www.adobe.com/devnet/xmp/`

[3] Adobe XMP Specification
`http://www.adobe.com/devnet/xmp/pdfs/xmp_specification.pdf`

[4] Apache Tomcat
`http://tomcat.apache.org/`

[5] Babel
`http://simile.mit.edu/babel/`

[6] exempi
`http://www.figuiere.net/hub/blog/?Exempi`

[7] ExifTool
`http://www.sno.phy.queensu.ca/~phil/exiftool/`

[8] Firefox
`http://www.mozilla.com/firefox/`

[9] FOAF
`http://www.foaf-project.org/`

[10] GIMP – The GNU Image Manipulation Program
`http://www.gimp.org/`

[11] Hypertext Transfer Protocol – HTTP/1.1
`http://www.ietf.org/rfc/rfc2616.txt`

[12] Java API for XML Web Services (JAX-WS)
`http://java.sun.com/webservices/jaxws/`

[13] JAX-WS Reference Implementation
`https://jax-ws.dev.java.net/`

[14] JempBox – XMP Compatible Java Library
`http://www.jempbox.org/`

[15] Longwell
`http://simile.mit.edu/longwell/`

[16] Mozilla
`http://www.mozilla.org/`

[17] Open Source Initiative
`http://www.opensource.org/`

[18] PdfLicenseManager
`http://media.polito.it/masala/plm2_index.html`

[19] Piggy Bank
`http://simile.mit.edu/Piggy_Bank`

[20] RDFizers
`http://simile.mit.edu/RDFizers/`

[21] RDFizers by Peter Jeszenszky
`http://www.inf.unideb.hu/~jeszy/rdfizers/`

[22] RDF/XML Syntax Specification
`http://www.w3.org/TR/rdf-syntax-grammar/`

[23] Resource Description Framework (RDF)
`http://www.w3.org/RDF/`

[24] SIMILE Project
`http://simile.mit.edu/`

[25] Solvent
`http://simile.mit.edu/solvent/`

[26] XHTML 2.0
`http://www.w3.org/TR/xhtml2/`

[27] The xmpincl macro package
`http://www.ctan.org/tex-archive/macros/latex/contrib/xmpincl/`

[28] XML User Interface Language (XUL)
`http://www.mozilla.org/projects/xul/`