# CABRef: Cross-Referencing into an Abstracts Database

Ann Apps and Ross MacIntyre
*MIMAS, University of Manchester,*
*Oxford Road, Manchester, M13 9PL, UK.*
*Email: ann.apps@man.ac.uk, ross.macintyre@man.ac.uk*

**Abstract** This paper describes a prototype system, CABRef, which provides seamless linking from a citation in the bibliography of an electronic journal article to a detailed abstract of the cited work, within a particular research domain. Publishers of journal articles will query CABRef to determine the URL links to embed in their article bibliographies which enable end-user cross-reference navigation. The abstracts within CABRef, and their discovery metadata, are encoded in XML and processed with XML-aware software. The CABRef XML abstracts database is updated using data exported from the CABI abstracts production database on a regular basis when abstracts are 'published', involving a simple extra step in the CABI abstracts production process.
**Keywords** : abstract, cross-reference, linking, Dublin Core, XML

## 1. Introduction

CABI *Publishing* [1] create and publish a comprehensive, bibliographic database, including high-quality abstracts, which covers the world-wide published research literature in all aspects of applied life science, including agriculture, forestry, veterinary science and human health and disease. Electronic access to these CABI abstracts and associated material is normally available on a subscription basis or pay-per-view via online hosts.

During recent years, CABI *Publishing* has funded MIMAS [2] at the University of Manchester to perform a programme of electronic publishing research and development. The programme's principal themes were SGML/XML production and linking technologies, with the emphasis on practical implementation rather than theory. The CABRef prototype system, described in this paper, was developed during this collaboration and allows free access to a single abstract for an end-user. Typically this will be via a link from a reference in a journal article, the system being enabled by Web links, incorporating appropriate CABRef identifiers, embedded into the Web display of that article by its publisher. The initial version of CABRef provides access to abstracts of journal articles, which constitute 85% of the records in the CABI abstracts database, but it is possible that later versions of CABRef would include other genres such as conference papers or books.

## 2. Overview

Research into journal article cross-referencing [3] indicates that there are three major components for constructing systems to support reference linking. Firstly, persistent, unique identifiers are needed for the target resources. Secondly, a 'look up' mechanism will be employed to discover the identifier of a resource from a citation. And thirdly, a mechanism is required to deliver a particular, identified item to an end-user. The CABRef identifier is a unique identifier for each CABI abstract. The CABRef system consists of two parts: CABRef *Look Up* is provided for publishers to determine the CABRef identifiers of references in their journal articles using basic citation information; CABRef *Abstract Display* provides a Web display of the CABI abstract of a cited work.

A cross-referencing paradigm could be seen as 'discover – locate – request – deliver'. In CABRef, 'discover' is performed by a publisher ascertaining a CABRef identifier for a citation using CABRef Look Up. CABRef abstracts are currently available in only one database, so 'locate' simply involves the embedding of the CABRef URL, along with the discovered identifier, into a journal article reference. The end-user 'requests' the abstract when activating this link, causing 'delivery' of the abstract by CABRef Abstract Display.

## 3. CABRef Look Up

CABRef Look Up returns the CABRef identifiers for a list of citations supplied in a prescribed text string format, via a Web page. Look up is performed on an XML [4] *Match File*, which is created and updated from information exported from the CABI abstracts database.

### *3.1. CABRef Query*

In general, publishers will request CABRef identifiers for a list of citations via a batch interface. In the initial prototype this is a Web page interface into, and from, which a publisher can 'cut-and-paste' a list of citation queries and replies. It is hoped to provide an email batch citation query service in the future, and a demonstration version of this has been produced.

Each citation query is a list of fields, separated by '|' and terminated by '|#'. The list of citation queries is returned to the publisher with their CABRef identifiers, and possibly comments, appended. For a journal article this query string is:

**J**|*JournalTitle*|*Year*|*Volume*|*Page*|*Author*|*PublisherRef*|#

where: *JournalTitle* is the full journal title; *Year* is the publication year; *Volume* is the journal volume number; *Page* is the start page of the article; *Author* is the first author in plain text as 'FamilyName Initials', and is optional; and *PublisherRef* is the publisher's own reference identifier for this citation. The returned query is:

**J**|*JournalTitle*|*Year*|*Volume*|*Page*|*Author*|*PublisherRef*|*CabRefId*|*CabRefComment*|#

where: *CabRefId* is the CABRef identifier of the citation or 'NotFound'; and *CabRefComment* is a possible diagnostic comment, for example 'UnmatchedAuthor' when the remainder of the citation query indicates a valid bibliographic reference except for the first author's name.

For the CABRef prototype, journal title abbreviations, for which there is no single standard, are not supported. Supplying the ISSN instead of the journal title is also not available, because ISSNs do not generally appear in journal article bibliographies. A wildcard or truncation character is not provided in CABRef queries, it not being the intention to provide a general search interface into CABI abstracts.

Example citation queries sent to CABRef may be:

```
CITATIONS
J|Telopea|1999|8|297|Mabberley DJ|MR020|#
J|Journal of Bacteriology|1999|181|3193|Wilck A|MR021|#
J|Journal of Great Lakes Research|1998|24|343|Dawson VK|MR022|#
J|Hayvancilik Arastirma Dergisi|1998|8|52|Kurtoglu F|MR023|#
J|Gartenbauwissenschaft|1999|64|97|Krug H|MR024|#
J|Gene|1998|225|67|Zeiss CJ|MR025|#
J|Journal of Park and Recreation Administration|1999|17|65|Norris PA|MR026|#
```

CABRef would return the following, where CABRef identifiers are in the penultimate field of each citation query reply:

```
CABRef CITATIONS
J|Telopea|1999|8|297|Mabberley DJ|MR020|19990310140||#
J|Journal of Bacteriology|1999|181|3193|Wilck A|MR021|19991109241|UnmatchedAuthor|#
J|Journal of Great Lakes Research|1998|24|343|Dawson VK|MR022|19990505783||#
J|Hayvancilik Arastirma Dergisi|1998|8|52|Kurtoglu F|MR023|19991412031||#
J|Gartenbauwissenschaft|1999|64|97|Krug H|MR024|19990310186||#
J|Gene|1998|225|67|Zeiss CJ|MR025|19990107152||#
J|Journal of Park and Recreation Administration|1999|17|65|Norris PA|MR026|19991809359||#
```

In order to assist publishers in resolving individual citations, a single citation query Web form is provided. Details of a citation can be entered into the appropriate fields of this form, and CABRef will return the identifier for the citation. Information about the format of a citation query may be obtained by sending ‘Help’ to the CABRef batch interface.

The CABRef citation query interface is based on the PubMed [5] ‘Citation Matcher’ model, which provides an email and a web batch interface for determining identifiers. CrossRef [6] uses a similar format for Digital Object Identifier (DOI) look up queries [7]. When designing the CABRef query format, this model was enhanced to include: a leading field indicating the genre of the abstracted resource (‘J’ for a journal article), thus allowing for later extension to other document types; a field for a returned diagnostic comment in addition to the identifier; and a terminating character (‘|#’) to obviate problems caused by newline characters, which are sometimes inadvertently introduced by intermediate software.

Possible alternative designs of CABRef queries could have utilised the ‘object metadata zone’ of the emerging OpenURL [8] standard, or defined an XML interchange format. An example, hypothetical CABRef citation query reply encoded as a partial OpenURL [9] might be (without the line breaks):

```
sid=CABI:CABRef&genre=article
&title=Gene&date=1998&volume=225&spage=67&aulast=Zeiss&auinit=CJ
&pid=<pubref>abc1234</pubref>
&<cabrefid>19990107152</cabrefid>&<comment></comment>
```

Encoding the same citation query reply in XML could be:

```
<cabrefQuery>
<genre>journal article</genre>
<journalTitle>Gene</journalTitle>
<year>1998</year>
<journalVolume>225<journalVolume>
<startPage>67</startPage>
<firstAuthor><snm>Zeiss</snm><inits>CJ</inits></firstAuthor>
<pubref>abc1234</pubref>
<cabrefid>19990107152</cabrefid><comment></comment>
</cabrefQuery>
```

For the prototype version of CABRef it was decided to use the more concise '|' separated text string for citation queries. As the designers of the CrossRef query interface [10] found: "one unresolved issue in implementing an all-XML query interface is the lack of an industry-standard XML query language", although this situation may be remedied when the definition of XML Query [11] is complete.

### *3.2. The CABRef Match File*

CABRef Look Up determines CABRef identifiers by matching citations submitted through the interface described above against XML records in the CABRef *Match File*. The Match File is a single XML file which holds a record for each abstract in the CABRef system, generated from data extracted from the CABI abstracts database. The fields within each record are only those required for citation matching against the citation query requests, rather than the full data record for an abstract. An example XML record in the Match File for an abstract is:

```
<cabcit>
<Identifier>19990107152</Identifier>
<Type>journal article</Type>
<JnlTitle>Gene</JnlTitle>
<JnlVolume>225</JnlVolume>
<JnlPage>67</JnlPage>
<Creator><snm>Zeiss</snm><inits>CJ</inits></Creator>
        <!— Publication Year -->
<Date Qualifier=" Created"  Scheme=" W3CDTF" >1998</Date>
        <!— CABRef load date -->
<Date Qualifier="Available"  Scheme=" W3CDTF" >2001-04-01</Date>
</cabcit>
```

Because the purpose of the Match File is for 'look up' text matching, the contents of its contained XML elements are all plain text and do not contain any XML character entities or typesetting elements. Only journal article records are included in the Match File, although the source data extracted from the CABI abstracts database will contain information about other types of work such as books and conference papers.

Look Up is performed by an OmniMark [12] program which extracts the elements of the citation queries from the input string, and then searches for the citations in the Match File. The Match File is processed as an XML file by parsing it against its Document Type Definition (DTD).

The initial prototype version of CABRef will hold data from 1997 onwards. It is hoped that CABRef will include earlier data in the future. The Match File will be updated at regular intervals, possibly weekly, with newly published CABI abstracts. It is possible that for future versions of CABRef this simple Match File will be replaced by a database. Although the Look Up data is essentially of simple structure, mapping to a single table in a relational database, holding it in a database may improve the performance of CABRef Look Up.

## 4. CABRef Abstract Display

It is expected that a publisher of a journal article will ascertain the CABRef identifiers corresponding to the references in the bibliography section of the article using CABRef Look Up described above. The publisher will then embed a CABRef display URL, appended by the appropriate CABRef identifier, alongside a reference in the HTML of the Web version of the article, to provide a link for an end-user to follow. The process used by a publisher for finding CABRef identifiers and embedding links from the references would be similar to that already used for including links to PubMed. When an end user follows one of these CABRef links the corresponding CABI abstract is displayed

### *4.1. CABRef Abstract XML Files*

The CABI abstracts available within CABRef are held as XML files, which are created and updated at the same time as the Match File, using data extracted from the CABI abstracts database. Display of an XML abstract is by an ‘on-the-fly’ conversion to HTML using an OmniMark program.

An example CABRef abstract in XML is:

```
<crab>
<dc:title>TIMP-1 expression is increased in X-linked progressive retinal atrophy despite its
exclusion as a candidate gene.</dc:title>
<dc:creator><cr:snm>Zeiss</cr:snm><cr:inits>C. J.</cr:inits></dc:creator>
<dc:creator><cr:snm>Acland</cr:snm><cr:inits>G. M.</cr:inits></dc:creator>
<dc:creator><cr:snm>Aguirre</cr:snm><cr:inits>G. D.</cr:inits></dc:creator>
<dc:creator><cr:snm>Kunal</cr:snm><cr:inits>R.</cr:inits></dc:creator>
<dc:creator><cr:aff>James A. Baker Institute for Animal Health, College of Veterinary Medicine,
Cornell University, Ithaca, NY 14853-6401, USA.</cr:aff></dc:creator>
<dc:subject Scheme=“ CABRef” >UU600</dc:subject>
<dc:subject Scheme=“ CABRef” >LL070</dc:subject>
<dc:description>X-linked progressive retinal atrophy (XLPRA) of dogs is an animal model
for…</dc:description>
        <!— Publication Year -->
<dc:date Qualifier=“ Created”  Scheme=“ W3CDTF” >1998</dc:date>
        <!— CABRef load date -->
<dc:date Qualifier=“Available”  Scheme=“ W3CDTF” >2001-04-01</dc:date>
<dc:type Scheme=“ CABRef” >Journal article</dc:type>
<dc:identifier Scheme=“ CABRef” >19990107152</dc:identifier>
<dc:identifier Scheme=“ CRCite”>
        <cr:JnlTitle>Gene</cr:JnlTitle>
        <cr:JnlVolume>225</cr:JnlVolume>
        <cr:JnlIssue>1/2</cr:JnlIssue>
        <cr:JnlPages><cr:ppf>67</cr:ppf><cr:ppl>75</cr:ppl></cr:JnlPages>
</dc:identifier>
<dc:language>En</dc:language>
<dc:relation Qualifier=“ IsPartOf”  Scheme=“ ISSN” >0378-1119</dc:relation>
</crab>
```

The fields of the abstract are described within the XML files using Qualified Dublin Core [13] elements (in a *dc* namespace) where applicable, with extensions within a CABRef (*cr*) namespace. The *dc:subject* fields contain CABI subject categories, which are not used or displayed by CABRef at present, but may have future uses when adding functionality such as 'more like this'. The *dc:language* field gives the language of the corresponding article, according to a CABI coding scheme but translated into the language's name for display, rather than the language of the abstract which is generally in English. Other fields in the XML record for a CABRef abstract which are not displayed are: *dc:type* which is always 'Journal article' in the CABRef prototype; and the 'Available' *dc:date* which is the date when the abstract was added to CABRef and for internal use only. Within the XML DTD, a further *Scheme* 'DOI' is allowed for *dc:identifier*, which would allow for the future addition of a Digital Object Identifier (DOI) of the abstracted article. CABI abstracts contain more data fields than are captured in a CABRef abstract, such as species and geographic descriptors, but it was decided to restrict the fields displayed through the free CABRef interface.

### *4.2. Display of an Abstract*

The above example XML abstract would be displayed to the end user as:

**CABRef Abstract 19990107152**

**TIMP-1 expression is increased in X-linked progressive retinal atrophy despite its exclusion as a candidate gene.**

**Author(s):** C. J. Zeiss, G. M. Acland, G. D. Aguirre, R. Kunal
James A. Baker Institute for Animal Health, College of Veterinary Medicine, Cornell University, Ithaca, NY 14853-6401, USA.

**Journal:** Gene
ISSN: 0378-1119
Volume 225 Issue 1/2, 1998, pp 67-75

**Abstract:**

X-linked progressive retinal atrophy (XLPRA) of dogs is an animal model for …

**Language:** English

### *4.3. Dublin Core for Journal Article Metadata*

Using Dublin Core elements to capture the abstract is not strictly necessary because these XML elements are used only within the CABRef system. The main purpose of Dublin Core metadata is for simple resource discovery and for interoperability. However it seems reasonable to use a standard encoding scheme and to follow best practice rather than inventing a specific CABRef one. If, in the future, there is a requirement to make CABRef interoperable with some other system the task of transforming the metadata elements to standard ones will be simplified.

Using Dublin Core to capture journal article metadata does present some problems, particularly in capturing author details and citation information, hence the addition of application specific elements within a CABRef namespace. In addition, some elements have application-specific *CABRef* schemes which are not strictly Qualified Dublin Core.

Dublin Core does not currently allow for a means of capturing the separate elements of an author's name. Thus CABRef includes sub-elements of *dc:creator* to capture the family name (*cr:snm*), the initials (*cr:inits*) and any suffix (*cr:extra*), or a corporate author (*cr:corpau*). It also includes a further sub-element, used in a separate instance of *dc:creator*, to hold the affiliation (*cr:aff*). CABI abstracts contain only one affiliation which means it is not necessary to link affiliations to authors within this application.

Within the Dublin Core community discussion is on-going to define a recommended way of capturing the citation information for a journal article, i.e. its containing journal, volume, and issue and its page range within that journal volume. A Dublin Core Citation Working Group recommended a best practice encoding scheme using either a structured string or XML sub-elements (a 'structured value') within the *dc:identifier* element. This recognises the fact that this set of citation information *identifies* the journal article. Although this 'dc:citation' encoding has not been endorsed by the Dublin Core Metadata Initiative it has been used in practice for capturing this information about journal articles [14]. The citation information encoding within CABRef largely follows this recommendation, albeit with slightly different sub-element labels within the CABRef (*CRCite*) namespace. The journal's ISSN is captured within a *dc:relation* element using an *isPartOf* qualifier. This reflects the fact that the ISSN is information about the containing journal rather than the article itself. In fact, all the citation information is about the containing journal volume and issue, except for the page range which is pertinent to the article itself. Some may argue that this information should not appear in the metadata record for the article itself, but should be pointed to from the article's metadata. However this approach does not seem viable within CABRef where all the information about an article's abstract is held in one record, with little knowledge of, or ability to access, information about the journal apart from its ISSN. The end-user will expect to see all the information from the abstract including its citation details within one web page.

## 5. CABRef System Architecture

An outline of the CABRef system architecture is shown in Figure 1.

Source abstract data records are exported from the CABI abstracts production database, for published abstracts, on a regular basis, in a tagged text format. These are converted into two sets of XML [4] records, one for the Match File and one for the CABRef abstracts database, which is a set of XML files. This translation is by an OmniMark program which verifies the resulting XML records by parsing them against the relevant DTDs. CABRef abstract XML files are organised within a file directory structure according to their CABRef identifiers.

When a publisher sends in batch citation queries, ascertained from references within journal articles, these are processed and returned by the Look Up program, another OmniMark program which searches the Match File to find the citation CABRef identifiers.

The publisher would use the returned CABRef identifiers to add Web links to journal article bibliographies. When an end-user clicks on one of these links, the Abstract Display program, also OmniMark, is called via the Web's Common Gateway Interface (CGI). This program processes the requested abstract's XML file by parsing it against its DTD and generating appropriate HTML which is returned to the end-user's Web browser.

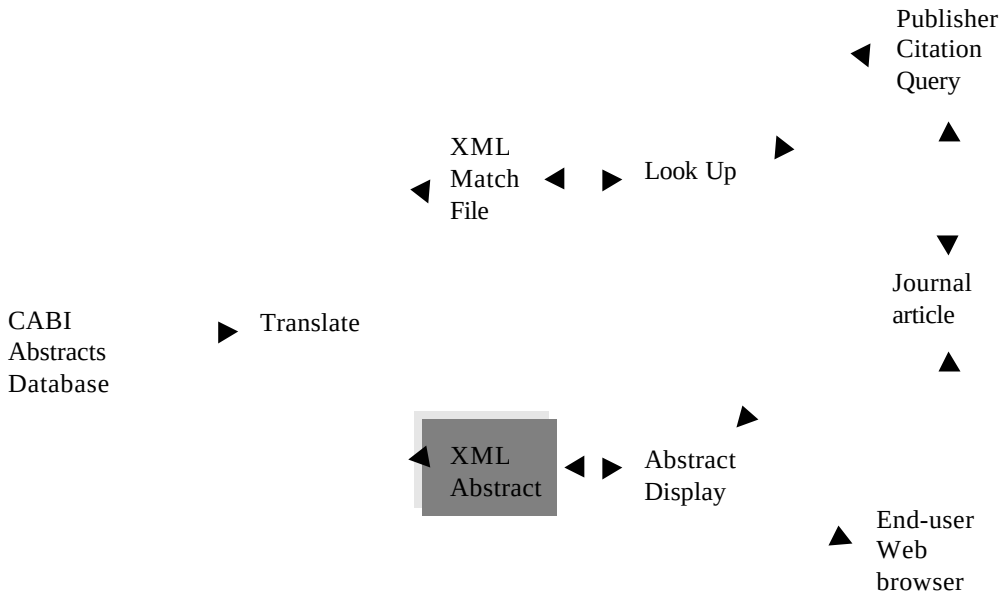


Figure 1. CABRef system architecture.

OmniMark [12] is an SGML/XML-aware 4th generation programming language which includes powerful pattern analysis and manipulation facilities. OmniMark was chosen as a software platform because, being XML-aware, it seemed appropriate for developing an XML application. The language includes Web CGI utilities which simplify the coding of the Web interface, is portable, and is available for most generally used platforms. At the beginning of the project OmniMark version 5.5 was freely available and was used for SGML-based journals production work within CABI, both of these influencing the choice.

CABRef is hosted on a Windows NT machine with an IIS Web server, but the code was initially developed on a Unix Solaris machine running an Apache web server. Porting the code required little intervention other than configuring the IIS web server to run OmniMark.

## 6. Future Development

CABRef is currently a prototype system and so has not been released for general use. It is hoped that several publishers within the applied life science domain will be encouraged to look up and include CABRef identifiers and links within the references of their electronic journal articles. It is intended that such links will be included within the primary full text journals published by CABI. Only when some of these links are in place can CABRef be developed and become generally used. But it will be essential that the CABRef abstract display URL remains persistent once it has been embedded into published articles.

The CABRef look up searching mechanism currently requires exact matching on the journal title, the volume number and the start page number. This primitive search algorithm has been built as a ‘proof of concept’. A production version of CABRef will probably need the introduction of a database, rather than the simple XML Match File, to improve searching performance, although, because this is a batch operation, ‘instant’ response is not a requirement. It is also likely that some fuzzy matching algorithm will be necessary to improve the ‘hit’ success rate.

At present, CABRef provides links to abstracts of journal articles only, but it could be extended in the future to include other genres such as books and conference papers. CABRef could also be enhanced to provide associated information and 'more-like-this' abstracts to CABI subscribers, but free access is likely to be always to a single abstract.

Future development possibilities could be envisaged of linking on from the abstract to its corresponding full text, via initiatives such as CrossRef [6], SFX Context Sensitive Reference Linking [15], or other resolution services. For older works possibilities could be envisaged of linking to library OPACs or Inter Library Loan document delivery services. The CABRef Abstract XML DTD already allows for inclusion of the article's DOI and CrossRef provide a mechanism for ascertaining DOIs. Resolution services such as SFX which attempt to find the 'appropriate copy' of an article for an end-user make use of OpenURLs [16,8] to transport the article's metadata. The CABRef XML abstract contains all the metadata elements, based on Dublin Core, which would be needed in such an OpenURL [9], thus making its on-the-fly generation easily possible. Adding future functionality becomes viable because the CABRef instances of CABI abstracts are held as XML files which can be processed using XML software.

CABRef will provide free 'taster' access to high quality abstracts for researchers within the applied life science domain, adding value to the bibliographic references in a journal article. Thus CABRef is a step on the way to providing academic researchers with seamless, 'joined-up' cross-referencing within their subject domain.

## 7. Acknowledgements

The CABRef prototype was developed as part of a research project within MIMAS at the University of Manchester funded by CABI Publishing. The authors would like to thank Andrea Powell, Product Development Director, and members of the CABRef team, in particular Carol Steel, Fiona Maclean and Christopher Plummer, at CABI Publishing, for their contribution within the project.

## 8. Note about Authors

Ann Apps has been actively involved in the Dublin Core Citation Working Group and is a member of the recently formed NISO OpenURL committee. Ross MacIntyre is the MIMAS senior project manager for digital library and electronic publishing research and development.

## 9. References

1. CABI *Publishing*, the publishing division of CAB International. http://www.cabi.org
2. Electronic Publishing at MIMAS, a UK Higher and Further Education data centre. http://epub.mimas.ac.uk
3. Caplan, Priscilla and Arms, William Y. (1999) Reference Linking for Journal Articles. *D-Lib Magazine* **5**(7/8) (July/August 1999). doi://10.1045/july99-caplan
4. XML (eXtensible Markup Language). http://www.w3.org/XML
5. National Center for Biotechnology Information, National Library of Medicine, PubMed. http://www.ncbi.nlm.nih.gov/PubMed
6. CrossRef. http://www.crossref.org
7. CrossRef Query Specification. http://www.crossref.org/docs/CrossRef_query_spec.pdf
8. OpenURL Syntax Description. http://www.sfxit.com/OpenURL/openurl.html
9. Powell, Andy and Apps, Ann. (2001) Encoding OpenURLs in Dublin Core metadata. *Ariadne* **27** (March 2001). http://www.ariadne.ac.uk/issue27/metadata

10. Atkins, Helen; Lyons, Catherine; Ratner, Howard; Risher, Carol; Shillum Chris; Sidman, David and Stevens, Andrew. (2000) Reference Linking with DOIs. *D-Lib Magazine* **6**(2) (February 2000). doi://10.1045/february2000-risher
11. XML Query. http://www.w3.org/XML/Query
12. OmniMark Technologies Corporation. http://www.omnimark.com
13. Dublin Core Metadata. http://www.dublincore.org
14. Apps, Ann and MacIntyre, Ross. (2000) Dublin Core Metadata for Electronic Journals. *Proceedings of the 4th European Conference on Research and Advanced Technology for Digital Libraries (ECDL2000)*, Lisbon, Portugal, 18-20 September 2000. *Lecture Notes in Computer Science* (Springer-Verlag) **1923** pp 93-102.
15. SFX, Context Sensitive Reference Linking (Ex Libris). http://www.sfxit.com
16. Van de Sompel, Herbert and Beit-Arie, Oren. (2001) Open Linking in the Scholarly Information Environment Using the OpenURL Framework. *D-Lib Magazine* **7**(3) (March 2001). doi://10.1045/march2001-vandesompel