BIBP: A BIBLIOGRAPHIC PROTOCOL AND SYSTEM FOR DISTRIBUTED REFERENCE LINKING TO DOCUMENT METASERVICES ON THE WEB

by

Serban G. Tatu
B.Sc., Technical University of Timisoara, 1994

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the School
of
Computing Science

SIMON FRASER UNIVERSITY
July 2000

0-612-61503-0

ABSTRACT

Bibliographic Protocol (BIBP) is a proposed method for the creation and resolution of abstract reference links on the World-Wide Web. In this context, an abstract reference link is a bibliographic citation denoting the referenced document as an abstract entity, independent of any particular manifestation or service with respect to it. Resolution of such a link logically provides access to a document metaservice, which in turn may provide access to a selection of alternative sources, formats or further services with respect to the document.

A distributed system based on BIBP is proposed, which links documents identified by Universal Serial Item Names (USINs) to document metaservers – Web services providing metadata about the documents. USINs are abstract identifiers that can be derived from standard bibliographic information; they can be used in conjunction with the BIBP scheme to form Uniform Resource Identifiers, which in turn can be embedded as hyperlinks in HTML documents. Client software resolves USINs to metadata retrieved from document metaservers chosen by the user. The document metaservers form a network that builds upon a large database of bibliographic information which is acquired in a piecemeal fashion by each metaserver. A prototype BIBP system has been implemented to illustrate the concepts presented in the thesis and to serve as a tool for the evaluation of technical issues.

To Debbie

Acknowledgments

Many thanks to Dr. Cameron, for his guidance, support and patience.

Many thanks to my family, for not letting me quit.

Contents

List of Figures

Chapter 1

Introduction

Reference (citation) links appear when one document refers to another. The realm of these links includes but is not limited to: the world of scholarly communication, where reference links appear in the form of bibliographic citations; the World Wide Web, where hyperlinks are used to create references between Web pages; the world of libraries, where links appear between bibliographic databases and printed matter or bibliographic services.

This thesis is concerned with the task of creating and resolving reference links on the Web; in particular, reference links between scholarly documents are considered (which include citations between journal papers, technical reports or books). The act of resolving a citation on the Web implies the existence of a mechanism that renders the citation "actionable" [41] - that is, the citation behaves like a regular hypertext link, which can be displayed in a Web browser and followed by a user. In order for such a mechanism to exist, a way to uniquely and unambiguously identify the cited entity must exist.

Traditionally, Uniform Resource Locators (URLs) have been used on the World Wide Web as reference links in bibliographic citations. However, they cannot unambiguously identify a document, for the simple reason that URLs were never designed to do that: a URL refers to a digital entity that resides on a networked computing system, whose content can be retrieved via a network protocol. This content may be deleted, modified, or moved; computing systems may cease to exist; network protocols may not be supported by a certain computer application. Moreover, a particular item may not exist in any electronic format – it may only exist in print.

Described above are just a few of the technical problems encountered when attempting to use URLs as identifiers for serial articles. It may be argued that the persistence of URLs is a matter of social engineering, since URLs and the content they refer to may be declared as unchangeable and be kept that way. However, it is a fact of life that URLs become broken, many times for legitimate reasons, and this pattern is not likely to change.

Although the traditional bibliographic citations can be viewed as links that unambiguously refer to published documents, it is commonly accepted [19] that they are not particularly suitable for machine interpretation. The validation and resolution of a bibliographic citation usually entails a human decision process, since there is no unique, standardized way of writing citations; furthermore, there is no guarantee that once written, they are correct (authors frequently omit information or make other errors when citing).

At the conceptual level, document authors want a citation to represent an abstract link to a paper, in other words to specify **what** the paper is, as opposed to **how** to resolve the citation. Since URLs merely specify how to get to a particular copy of an article, they cannot be expected to function well as citation links. Therefore, systems based on other identifiers than URLs have been the subject of research by various groups. Citation linking research is closely related to the development of such identifier systems.

A citation linking solution based on persistent identifiers needs two additional mechanisms [19]: a mechanism for discovering the identifier of an article from a citation, and a mechanism for taking the reader from an identifier to a particular item, or to a service that is relevant to that item. This thesis proposes a framework that includes all the above mechanisms. Universal Serial Item Names (USIN) [16] are used as a system of identifiers specifically designed for items published in a serial fashion. USINs are easy to construct based on just the article citation and standard bibliographic identifiers; in addition, they are mnemonic and scholar-friendly. The BIBP framework, based on the BIBP[1] Internet protocol, is presented as a mechanism for USIN resolution.

The BIBP framework is based on a network of citation meta-servers, which provide access to metadata about serial items, including options for fulltext retrieval or document ordering. Given an article identifier, a meta-server will attempt to find the full bibliographic description, as well as a list of all known services with respect to the identified document. Examples of such services include search interfaces for holdings of documents in libraries, document delivery systems, links to online copies of the document (if any), abstracts, reviews, classifications, and translations.

[1]BIBP – BIBliographic Protocol (pronounced "BIB-Pee")

A key point in the development of the BIBP framework was that readers should be able to designate a specific service for resolving document identifiers, for instance a service operated by a local library. The motivation behind this requirement is that a local service may be able to offer its patrons more extensive information and access to information than a generic service. Libraries often enter contractual agreements with other libraries or bibliographic servers in order to broaden their coverage and improve their service. Thus, libraries can provide their users with access to items otherwise forbidden to the general public. A concrete example: the SFU library has a site license with the ACM Digital Library, which offers fulltext access to all the articles published in ACM journals since 1990. To the SFU community, access to these articles is provided for free as part of the license while to the general public, ACM offers pay-per-view access.

From a researcher's perspective, there are two basic modes of interaction with the BIBP framework: resolution of article citations and contribution of article metadata. When the researcher is trying to follow a citation link in order to gain access to article metadata, the first mode is used. Researchers might also want to contribute citation data about their own papers or about others, in which case they would use the second mode of interaction. The framework also offers help in the authoring process, by providing services for the construction of USINs and bibliographies; these services would ideally be contacted by the researcher's document preparation software.

Researchers are not the only actors in the system: there are also libraries and library consortia, publishers, bibliographic services (such as document delivery or citation indexes) and other BIBP servers. The BIBP system exposes several interfaces based on standard developments in Internet technologies, across which these actors can interact and exchange structured metadata. These interfaces and their interactions are grouped under the BIBP protocol specification, which is a HTTP-based protocol [26] divided into two layers with increasing degrees of complexity:

- The first BIBP level (Level 1) [17] deals with the task of simple resolution of USIN-based hyperlinks embedded in HTML documents and the retrieval of metadata marked up in HTML. A Level 1 implementation allows authors to create citation links in HTML documents based solely on USIN identifiers. Users can then select a BIBP server of choice, view the documents in a Web browser and follow the citation links as if they were regular URLs, in order to retrieve article metadata. Level 1 is designed such that a compliant BIBP server would be trivial to implement, thus exposing the framework to the large public and initiate widespread experimentation. The protocol uses only the most basic HTTP mechanisms (i.e. the `GET` request), in order to allow for the implementation of clients running in Web browsers.

- The second BIBP level (Level 2) adds facilities for retrieval of metadata marked up using various other formats than HTML (e.g. XML[2], RDF[3], BIBTEX) and for contribution of bibliographic metadata. Level 2 also specifies how BIBP servers can communicate with each other and form a scalable network of collaborating metaservers. The initial revision of the protocol is still based on basic HTTP mechanisms; however, it is envisioned that further revisions will employ RPC-based communication between clients and servers, based on the SOAP[4] [13] standard, thus enabling the creation of more sophisticated clients, such as inclusion of BIBP client capabilities in document preparation software.

The rest of this document is organized as follows: Chapter 2 provides the background, terminology and current state-of-the-art with respect to citation linking, placing BIBP in the context thus defined. Chapter 3 presents the architecture BIBP framework, while Chapter 4 describes a prototype BIBP server implementation. Chapter 5 evaluates the BIBP architecture and the prototype implementation. A conclusion and directions for future development are presented in Chapter 6.

[2]XML – eXtensible Markup Language
[3]RDF – Resource Description Framework
[4]SOAP – Simple Object Access Protocol

Chapter 2

Background

2.1 A Brief Introduction to USINs and the BIBP Protocol

This section offers an overview of some of the USIN and BIBP concepts, in order to help the reader place them in the more general context of citation linking. A complete description of the USIN identifier system can be found in [16], while the BIBP framework will be exposed in the chapters to come.

USINs are persistent, unique identifiers assigned to documents published in a serial fashion. Examples of such documents include journal or magazine articles, technical reports, books or any other items that may be published in a series. In theory, one can assert that any published item, regardless of its nature, can be made to belong to a series, because it can be assigned a serial number within a particular numbering space. This project needs not go that far, since the primary concern is bibliographic items and therefore it can use the name space offered by the world of standard bibliographic identifiers.

Given an article citation, one may build a USIN for the article in the following way:

- Find the standard bibliographic identifier for the serial in which the article was published. If the article was published in a journal, then the ISSN number of the journal is used. If the article was published in a conference proceedings book, then the ISBN number of the book is to be used. For other types of publication venue or alternative forms of serial identification, similar identifiers can be built [16].

- Use the numbers provided with the citation. These numbers (i.e. volume, issue, page) follow the numbering pattern employed by the serial and need to be present in a valid citation, unless they can be implicitly deduced by counting (counting is employed for instance in the case of those electronic journals that are unpaginated, where the position of an article within the issue is given by counting through the table of contents).

- Arrange the above elements discovered in their natural, hierarchical order: serial identifier, followed by volume, issue, page and other numbering components.

Suppose an author writing a paper wishes to make the following citation:

R. Abbott and H. Garcia-Molina. *Scheduling Real-Time Transactions: A Performance Evaluation*. ACM Transactions on Database Systems, Vol. 17, No. 3, pp. 513–560, September 1992.

The first task is finding the ISSN number for the journal ACM Transactions on Database Systems. If the author has a copy of the journal, then the ISSN number is printed on it. Otherwise, the author can get the ISSN number from the local library or from the Internet.

The rest of the elements are already present in the citation. What the author needs to do is arrange these elements according to the simple and mnemonic USIN syntax, as follows:

```
ISSN/0362-5915:17(3)@513
```

This USIN can be used as an unambiguous reference for the article cited above. It is an abstract identifier: it does not identify any particular copy of the article, be it on-line or on paper; nor does it identify the intellectual content of the article (and therefore it is not copyrightable); it does not specify how one can get to the article; and it will not change over time, since it only contains publication facts which are unquestionable, permanent truths.

Web hyperlinks created with USINs can be resolved to article metadata by processing entities that implement the BIBP protocol. To continue with the above example, if the author wishes to publish the paper as an on-line HTML document, a simple citation link can be constructed using the standard HTML anchor tag `<A>` and a URI-compliant syntax [11], as follows:

```
<A HREF="bibp:ISSN/0362-5915:17(3)@513">Text</A>
```

The link thus constructed is called a "BIBP link", to signify the fact that the URIs use the BIBP identification scheme, just as the links that use the HTTP identification scheme are called "HTTP links".

While the BIBP protocol is not directly supported by today's browsers, their scripting capabilities allow the construction of client software that recognizes this type of links and transforms them into URLs which are afterwards resolved by the browser. The process of transformation, as well as the URL and response formats, are part of the Level 1 specification of the BIBP protocol.

2.2 Identifier Systems

The ability to effectively qualify and retrieve information items depends on the availability of adequate identification schemes. Over time, many identification schemes have evolved, according to the needs of the communities involved in the production or use of information items. The recent years have seen a convergence of interest towards the Internet as a medium for the exchange of information items, which brings together communities that used to be separated by different interests. These communities need to find ways to reconciliate the various identifier and metadata systems that they have developed and find a common terminology for the items they want identified. The development of a restricted set of universal identifier systems is crucial to the future of electronic data interchange [27]. It is unlikely that a single such system will evolve as a universal solution, since items can and will be grouped according to various criteria [41]; however, it is important to discover these criteria and work towards the development of systems that support them.

Having established the need for article identifiers, the world of learned journal publishing, which includes publishers, document delivery companies, libraries and library consortia, developers of digital library systems and other parties interested in the management of scientific literature on the Internet have sought to reconcile their systems in order to insure interoperability. The scope of the problem has been broadened however by the desire to interoperate with other systems with similar goals but with different standards, such as the music industry.

Recent developments have shown a trend towards developing identifier systems for intellectual property. While this is a perfectly legitimate goal, difficulties arise from the generality of the concept. Finding a common terminology and set of abstractions for describing the nature of the items that need identification is one of the problems; a study published by the International Federation for Library Associations [36], which identifies the entities of interest to users of bibliographic records has been adopted as a starting point by most of the identifier system communities. The taxonomy and vocabulary provided by the study are very general and have been augmented to include other aspects related to ownership and intellectual rights [10]; despite these efforts, the simple task of categorizing a journal article does not yield precise results [19]. Another aspect is that of establishing what is required from a proper identification system. For this purpose, two basic sets of requirements have been recommended [42] for the evaluation of identifier systems: the requirements stated in RFC1737 [46] for identifiers applicable on the Internet and the suggestions made by a member of the ISO TC46/SC (ISO Technical Committee for Information and Documentation Standards), which go beyond technical aspects and delve into business issues and system administration.

While it might seem that the fundamental problems of terminology, taxonomy and basic requirements have been resolved to a certain degree, identifier systems that achieved widespread use and recognition while fulfilling the task of identifying any item of intellectual property have been slow to appear. This review will concentrate therefore on identifier systems that apply to a smaller problem, that of citation linking.

Scientific literature has relied on citations as a system of identifiers for published works. Citations link together related articles; these links are the foundation of scientific communication [41]; the linkage system based on citations has a much longer history than the cyberspace or computers. Citations are primarily meant for human consumption, although computerized finding aids have been developed to help in getting from a citation to the cited paper.

Finding a paper is perhaps the most important aspect of literature research, but one can enumerate many other aspects of scholarly communication that rely on accurate citation processing [15]: cocitation/coreference analysis, evaluation of scholarly work, evaluation of publication venues, current awareness, and so on. The traditional citation text by itself is inappropriate for usage in such endeavours, due to its complexity and ambiguity of interpretation by a computer. Indeed, there are successful systems that employ advanced information retrieval and artificial intelligence techniques in order to extract structured information from citations [12, 14, 38], but they cannot be used in a total linking solution, where 100% accuracy is needed in the identification of a cited article. Hence, there are two possible solutions:

1. Change the citation system to use a structured format that allows computerized processing. Aside from the obvious unlikelihood that such a change would ever happen,

there is a vast mass of legacy literature that cannot be covered by the solution.

2. Use extra information to compute a unique identifier or a key to use in an identifier lookup. This information can be used by itself or in conjunction with the bibliographic information contained within a citation.

It is evident that the second approach needs to be adopted. The extra piece of information that is needed may constitute an identifier by itself or it may be an identifier fragment. Either way, the resulting article identifier should meet the following requirements [19]:

1. *Persistence*: the identifier must have a permanent lifetime.
2. *Uniqueness*: an identifier must denote one single item (although the item may be denoted by several different identifiers).
3. *Multiple Resolution*: the identifier system must support resolution to multiple items. This requirement arises from the fact that multiple copies of (or service with respect to) the same article may exist and therefore it should be possible to build a mechanism whereby an appropriate copy (or service respectively) is selected.

There may be no direct correspondence between the identifier and the information contained within the citation text (i.e., examination of the identifier should not tell anything about the article) – in this case the identifier is called *dumb*. If on the contrary, some meaningful information can be derived from the identifier, then we have an *intelligent* identifier. A related concept is that of *affordance*, or *computability*, which means that one can derive the identifier solely from the examination of the article citation. Finally, *readability* is a concept referring to an identifier designed to be easily interpreted by humans.

A dumb identifier can be embedded in an electronic citation and then used as a key in a resolution database which returns article metadata. Since there is no direct way to discover the identifier from the citation itself, the cited article must be assigned an identifier prior to being used in the citation. Building a system based on dumb identifiers may therefore require a retrospective assignment of identifiers to all the articles published before the system took effect. While dumb identifiers are obviously not an ideal solution for citation linking, it is worth mentioning that, given the perceived difficulty of implementing a system of intelligent identifiers capable of addressing the present and future needs of all the interested parties, dumb identifiers are seen by some as the only viable solution to the more general problem of identifying intellectual content [27].

If retrospective assignment is to be avoided, then some degree of affordance must be present. Standard bibliographic identifiers for books and conference proceedings (ISBN) and journal articles (ISSN) are examples of successful identifiers, the former being (arguably) of the intelligent type, while the latter being of the dumb type. ISBNs and ISSNs cover both paper and electronic content and are amenable to Internet deployment, as they can be grandfathered into Uniform Resource Name schemes [31]. Therefore, they are highly suitable to be used in article identification, as seen in the case of USINs, where they are combined with bibliographic information present in the citation to yield intelligent, affordable identifiers. The association between USINs and the articles they identify is implicit, as it comes into existence the moment the article is published; thus, no retrospective assignment and no reference lookup database are needed to store these associations.

The following subsections provide a short review of several identifier systems, including the USINs, in order to illustrate the concepts discussed so far and evaluate their ability to solve the citation linking problem.

2.2.1 Uniform Resource Names

The Uniform Resource Names (URNs) are part of a larger identifier space, that of the URIs (Uniform Resource Identifiers). A URI is "a compact string of characters for identifying an abstract or physical resource" [11]. URIs can be classified as locators, names or both. A uniform resource locator (URL) is a URI designated to identify a resource via a representation of its basic access mechanism, whereas a uniform resource name (URN) is a URI designated to uniquely identify a resource in perpetuity, even after the resource has ceased to exist.

A URN consists of three components, as follows:

1. The string "urn", announcing that this is a URN.

2. A namespace identifier (NID), specifying which particular namespace is being used (e.g. ISBN, or ISSN).

3. A namespace-specific string (NSS), an opaque string representing the unique label of the resource within the selected namespace.

As such, a hypothetical example of a URN identifying the journal "ACM Transactions on Programming Languages and Systems" might be:

```
urn:ISSN:0164-0925
```

The resolution of a URN represents the process of specifying a resource attribute (e.g. location) from a URN [42]. There are two steps associated with the resolution process:

1. Locate a resolver – a service capable of mapping the URN to information about the identified resource. Sollins [45] proposes a generic framework for resolver discovery. In this architecture, the first task of the client software trying to locate a resolver for the example URN above would be to obtain the reference of a Resolver Discovery Service (RDS) by looking up the "ISSN" namespace in a global NID registry. The response would come in the form of a reference to an RDS, or a rule which the client should apply in order to get a reference to an RDS. Once the RDS is accessed, it would return the set of addresses for resolvers, or a rule for discovering another RDS that may be able to end the process.

2. Communicate with the resolver using a chosen protocol (e.g. HTTP) in order to obtain metadata about the object, possibly including the location(s) at which the object resides.

Functional requirements for URNs have been spelled out as early as 1994 [46] and a syntax for their representation has been formulated [33] and agreed upon by several implementors [7]. However, practical implementations of URN systems imply global consensus with respect to the maintenance of namespace registries, as well as commitment from individual namespace maintainers to support their names in perpetuity. These grand goals have not yet been achieved by any implementation, illustrating the need for a more focused approach.

2.2.2 Digital Object Identifiers

The Digital Object Identifier (DOI) is the seen by some as the most promising identifier system in terms of feasibility of implementation and institutional support at a global scale. The International DOI Foundation is a consortium spearheaded by publishing organizations, having the stated purpose of "supporting the needs of the intellectual property community in the digital environment, by establishing and governing the DOI System, setting policies for the System, choosing service providers for the System, and overseeing the successful operation of the System"[1].

The term "DOI" is used to denote a persistent, unique identifier and a system that processes the identifier in order to deliver services. DOI identifiers can be used as URNs, since they satisfy the URN requirements [40].

The identifier consists of two parts: a *prefix* containing the directory designation and a registrant number, and a *suffix* that uniquely identifies the item within the namespace of the registrant. The prefix and the suffix are separated by a forward slash ('/'). The prefix has two components, separated by a dot ('.'). The first component identifies the Directory Manager which assigned the DOI (at present, only one such service exists, identified by the string "10"), and the second part identifies the naming authority - typically a publisher. When used, a DOI identifier is to be interpreted as an opaque string (a dumb identifier). For instance, a DOI-based URN would be rendered as follows:

```
urn:doi:10.1234/12345679
```

The focus of the DOI system is to enable trading of copyrightable items over the Internet while protecting the intellectual property rights of the copyright holders. The DOI system is concerned with the identification of "creations", which are defined as items of intellectual property - any results of human imagination and/or endeavour where rights may exist.

The International DOI Foundation has reserved the right of allowing an institution to issue and update DOIs in order to preserve a high degree of integrity for its system. The institutions which are issued prefixes typically represent publishers, who are responsible for the upkeep of the metadata associated with the DOI, which is stored centrally at locations managed by the DOI Foundation.

The current implementation of the DOI system uses a resolution database which stores a location (URL) as metadata for each DOI assigned. For the purpose of scalability and multiple resolution, future developments of the system need to store extended metadata which would describe a creation beyond its mere location(s). Designing a usable set of metadata elements has been the subject of extensive research: it was imperative that a core set of elements be developed, which would allow a better description of the creations identified by DOIs, but at the same time be re-usable across many application domains which

[1]The Web page for the International DOI Foundation currently has the URL: http://www.doi.org/

would employ DOIs as identifiers. A "DOI kernel" was proposed as a core set of elements [40], to which extensions may be applied in the context of a particular DOI "genre" (a class of resources considered to have common characteristics). For instance, when used in reference linking, the DOI kernel has to be extended to describe article and journal-specific information, such as volume, issue and page information.

The DOI system has problems of its own, some of which have been exposed by Davidson and Douglas [22]:

- Only commercial and society publishers are currently allowed to register and update DOIs. While this is a legitimate approach by the DOI Foundation, it comes at odds with the stated purpose of identifying all the creations, since most of the individual or non-traditional publishers do not participate in the DOI System.
- The costs related to the upkeep of the identifier system (fees levied by the DOI Foundation) as well as the costs associated with the upkeep of the in-house databases are not marginal; this factor might affect small-scale publishers who cannot afford the participation.
- Being a dumb identifier, the DOI is not suitable for applications outside the digital realm, such as manipulation by humans.
- The DOI will not lead to a more open access to online materials - users who cannot demonstrate adequate access rights will still be shut out regardless of what identifier system is used.

When used in citation linking, more problems crop up:

- If the DOIs are to be used in citations, then only those articles which have received a DOI can be linked. Given that not every publisher can assign a DOI, the percentage of linkable journal articles will be far from 100%.
- Those publishers who are allowed to assign DOIs need to assign them retrospectively to all their articles.
- Due to their lack of affordance and potentially excessive length (up to 128 characters), DOIs are not easy to use in colloquial communications, such as e-mail. The solution lies in choosing intelligent identifiers in the suffix part of the DOI (for instance SICI

codes, see 2.2.3). Thus, the DOI would be dumb when parsed at the DOI site and intelligent when parsed at the publisher site.

- The usability of the DOI would be improved if the metadata stored with a DOI contained pointers to other services than the publisher, which may be able to offer information or even copies of the item identified. It is unclear whether different parties (e.g. a document delivery service) holding rights to a creation are allowed to update metadata for DOIs that have already been assigned to publishers.

Most of the problems enumerated above do not stem from the fundamental concepts that form the base of the DOI system; their resolution depends on managerial and political decisions. But an identifier system whose sole focus would be citation linking can arguably provide a more natural solution to the problem, without expressing any of the deficiencies above.

In terms of practical implementations, the DOI system has seen significant progress only recently, with the completion of a prototype system for reference linking. The CrossRef initiative[2] involved the participant publishers by making them submit bibliographic data for the articles they have published to a DOI metadata database, obtaining DOI for the articles and embedding them into online references. The results of the experiments have been published in a study by Atkins *et.al.*[9].

2.2.3 Serial Item and Contribution Identifiers

The Serial Item and Contribution Identifier (SICI) is a code for identifying issues of serial publications and contributions within them (e.g. articles). SICI is an ANSI/NISO standard adopted in 1991 and revised in 1996 [37].

In SICI terminology, an "item" denotes an issue of a serial, while "contribution" denotes an identifiable part of the issue. Therefore, a SICI for an article identifies it in a physical form, as part of a serially published issue.

A SICI code is structured into three segments separated by punctuation marks:

1. An *Item Segment* describing the issue and containing the ISSN for the journal, the chronology (date(s) of publication) and enumeration (e.g. volume, issue).

[2]The home page of the CrossRef project is currently at URL: http//www.crossref.org

2. A *Contribution Segment*, containing location information for the part of the issue; the location may be a page number and a title code, or an identifier specified by a numbering scheme not known to SICI (for instance a dumb identifier).

3. A *Control Segment*, aiding in the interpretation of the SICI code - in essence, this segment contains metadata.

Building a SICI code is a complicated task; the identifier contains significant intelligence, since it was not only designed for unambiguous identification, but also for aiding in manipulation by various services that need to manage bibliographic information. The contribution segment may contain abbreviations of the article title that have to be made according to precise rules (still yielding ambiguities when two or more article on the same page generate identical abbreviations for their title). The control segment contains descriptors for media and format identification (e.g. print, online, Braille text) and check characters.

As an example, one of the possible SICI codes for the article quoted in section 2.1 is:

```
0362-5915(199209)17:3<513:SRTAPE>2.0.TX;2-V
```

This code was obtained by putting together the following information:

- The item segment:
  - The ISSN of the journal (0362-5915).
  - The chronology for the journal issue (1992, September).
  - The enumeration for the journal issue (Volume 17, issue 3).
- The contribution segment:
  - The page at which the article resides (513).
  - The abbreviation for the article title (Scheduling Real-Time Transactions: A Performance Evaluation ⇒ SRTAPE)[3]
- The control segment:

[3]Note the arcane way in which the hyphenated pair of words "Real-Time" has been abbreviated to "R" because the SICI standard stipulates that words are seen as character strings separated by spaces. This is only a mild example of user-unfriendliness with respect to building a SICI code.

- A *Code Structure Identifier* (CSI), announcing that this is a SICI for an article (as opposed to an issue identifier, or an identifier containing a foreign identifier). The code for articles is 2.
- A *Derivative Part Identifier* (DPI) distinguishing between contributions (articles) and parts of contributions (abstracts, tables of contents and indexes). The code for a whole contribution is 0.
- A *Medium/Format Identifier* (MFI) identifying the presentation format (16 different codes). The code for printed text is "TX".
- The revision number of the SICI standard. Version 2 is used here.
- A check digit obtained by applying a modulus 37 algorithm to the characters in the SICI.

SICIs can be built with whatever information is at hand; the standard allows valid SICIs to have missing (redundant) parts that do not aid in unique identification. Thus, multiple SICIs may exist for the same article; without a specification of how to build a standard canonical form, it is very difficult to perform SICI comparisons for the purpose of establishing logical equivalence, especially when they exist in the third (CSI-3) form, which includes private identifiers, such as bibliographic database keys.

For the purpose of unique identification, SICIs contain unnecesary information. However, the inclusion of redundant information is done in order to make the SICI codes contain enough metadata to support tasks such as serials check-in or document delivery. Although their use as citation identifiers in a reference linking system has been envisioned [42], one has to consider their low degree of affordance, length, redundance and difficulty of establishing equivalence when building a practical system based on SICIs.

2.2.4 Publisher Item Identifiers

The Publisher Item Identifier (PII) [44] was introduced in 1995 by a group of scientific publishers, with the purpose of unambiguous identifications of items that are processed or exchanged in an electronic environment. Although the SICI scheme had been in existence for four years, the publishers involved in the PII development needed a different type of identifier [39]. As such, the PII was designed to be a dumb identifier based on ISSN and ISBN (but with the only purpose of unique identification), which has a one-to-one correspondence with the item identified. The PII can identify formats and items other than printed matter and is generated by the originator (publisher), without the need of any central registry.

The PII is still in use with a small number of publishers, but the revised SICI and more recently, the DOI developments, have contributed to diminish the attention given to this identifier.

2.2.5 Universal Serial Item Names

A short introduction to USINs has been provided in section 2.1 in order to give the reader the opportunity to form a basic idea about their position with respect to the other identifiers discussed. This section aims to broaden this insight and to expound the role USINs have in a citation linking solution and this project in particular.

USINs are intelligent identifiers – one of the primary directives in their design was that they should be mnemonic and reproducible by humans. However, the intelligence carried in a USIN is much reduced from that of a SICI for instance, and for a good reason: the only amount of metadata carried by the identifier is that which is necessary for unique identification; beyond the task of unique identification, no other metadata is required. This characteristic puts the USIN at a certain distance from the SICI, which is fraught with bibliographic information targeted at helping with library tasks, at the expense of being hard to construct by non-specialists; further, it can be argued that the SICI does not even provide unique identification, as cases have been cited in which the same SICI denoted two different articles [42].

The USIN is related to the SICI and the PII in its purpose of identifying items published in series, but it attempts to cover a broader spectrum: in addition to those serials uniquely identified by an ISSN (which is the only type of serial possible in a SICI), USINs are applicable to other types of serials, such as technical reports, books and newspaper articles. In concept, a USIN can be built for any item appearing in a series which numbers its items in a hierarchical fashion: the item can be identified by a tuple containing the identifiers for each level in the serial's hierarchy. For current scientific journals organized into volumes and issues, articles may be identified by (volume, issue, page) 3-tuples; if the journal is paginated by volume only, then the issue number is redundant and therefore not required: (volume, page) 2-tuples are preferred. For a newspaper, tuples of the form (volume, issue, edition, section, page, column, item-count) may be appropriate. In short, USINs are envisioned to provide a syntax for the mnemonic expression of hierarchical numbering tuples.

The identification of serial publications is not dependent upon the assignation of an ISSN. In the case of those serials for which no ISSN has been assigned, other schemes may be used. One such scheme involves the usage of Internet domain names for a restricted set of well known societies that issue serial publications; this scheme is identified by the `RDNS` USIN domain (Restricted Domain Name System). As an example, the technical report series for the School of Computer Science at SFU may be identified by the USIN `RDNS(sfu.ca).CMPT/TR`. Another scheme assigns mnemonic names to serial publications; under this scheme, the journal *ACM Transactions on Database Systems* may receive the USIN `S.ACM/TODS`. It is assumed here that the abbreviation `TODS` would have been assigned by an appropriate authority, as would be the name `ACM` to designate the Association for Computer Machinery and the symbol `S` that puts the institution in the domain of scholarly societies.

Although many different USINs can designate the same item, only one of them is considered the *canonical* (or *preferred*) form. USIN processing software should be able to transform any USIN into its canonical form, thus allowing the determination of USIN equivalence. The canonical form allows the usage of USINs in citation manipulation; in fact, USINs are envisioned to be used as keys in a universal citation database [15] - a global citation indexing system in which bibliographic metadata is entered directly by authors upon publication of their papers. The fundamental model underlying the universal citation database concept is that *citation information is entered at the source*, by the authors themselves, and not by third party organizations (namely, abstracting and indexing services); contributions at the source may also be used in a citation linking system based on USINs, as in the case of the project presented in this thesis.

The USIN identification scheme is larger in its coverage than the SICI scheme; however, it is much smaller from this point of view than the URN or DOI schemes. By concentrating on the narrower problem of identifying published serial items, the USIN scheme avoids fundamental problems that confront developers of URN or DOI solutions. In particular, it is clear from the start what is to be identified: serial items are easy to grasp in concept and constitute perfect candidates for items that need to be permanently identified. There need be no discussion as to how to formulate the fundamental concepts underlying information resources, categorize and model their interactions, as in the case of the DOI scheme.

USINs are a work in progress. A syntax has been established for journal articles appearing in paper and electronic journals, books and book articles, and institutional reports.

This suffices for the purpose of this thesis. Research is being conducted on serial identification and development of new syntax specifications for other classes of serial items, such as newspapers.

2.3 Metadata

Metadata constitute an intrinsic part of an identifier system; a system based on abstract identifiers needs the support of adequate metadata that allows its users to derive further information about the item. The concept of metadata used in bibliographic research predates the electronic era, as library cards have been used for a long time to help humans resolve citations.

Metadata can be contained within the identifier itself, for instance when the article citation itself is considered an identifier; identifiers that contain metadata are of the intelligent type. Dumb identifiers contain no descriptive elements and therefore need to be mapped to metadata; in this capability, they effectively function as keys into a metadata database. It can be argued that metadata support is necessary for most of the identifier systems, since regardless of their degree of intelligence, they do not provide a comprehensive description of the identified item. Moreover, including more metadata than necessary in the identifier forebodes major problems of scalability and extensibility [27, 39]; if extensions are to be made, they can be made better at metadata level.

Given the need for metadata, an important question is that of which attributes of the identified item need to be included in the metadata records. The answer to this question is a difficult one and generally depends upon the scope and generality of the identifier. It is much easier to establish common properties that need to be exposed for items belonging to a coherent and tightly knit class, such as the one defined by journal articles. When the identifier system encompasses a wide range of domains with heterogeneous classes of items that share very few (if any) properties, the design of a metadata set gravitates toward establishing *core* (also known as *kernel*) sets of elements for the properties that are common to all items, augmented with *qualifier* sets – elements that further specialize core properties according to a particular domain of activity; a classic example of a core element is `Date`, which may have qualifiers for particular date formats (day/month/year, month/day/year, etc.).

Once the element and qualifier sets have been determined, a key issue is that of how to structure them in order to insure interoperability among applications. The concept of a "well-formed" approach is used to describe a design that facilitates the resolution of the above issue. Well-formed metadata describe entities based on their inherent structure rather than their application roles, with elements that are chosen from a controlled vocabulary (i.e. free-text descriptions are not normally acceptable unless used for self-standing, autonomous labels such as titles) [43].

The main forces in metadata development for the World Wide Web are the Dublin Core element set [47] and the element sets being developed by the INDECS/DOI[4] project. The Dublin Core (DC) element set has been developed in order to aid in resource discovery in a networked environment. Thus, its elements are designed to be general while allowing for extensions (qualifications) that serve the needs of particular applications. The element set of the INDECS/DOI project is similar to the Dublin Core set with respect to resource discovery, with a particular focus on the description of people (individuals or institutions) and intellectual property agreements that exist between them [43]. The two initiatives seek to converge, under the motivation that descriptive metadata is an integral part of rights metadata [10]. In order to do so, both initiatives have adopted the IFLA analysis and model [36] as a common logical model for metadata, and extended it to accomodate concepts for the description of rights and their transferability, and to add further finesse to the description of people.

These convergence efforts will hopefully yield an interoperable element set that can be used to describe bibliographic items and exchange them over the Web, much like the UNIMARC element set [2], which allows interoperation between libraries. Both projects are involved in developing elements and syntaxes for citation metadata: the Dublin Core initiative has started a Citation Workgroup[5], while the DOI foundation has developed a kernel set called DOI-X which has been used in the CrossRef prototype project [9]. At the time of this writing however, there is no de facto core metadata element set for citation linking.

For the syntax and encoding of metadata, XML is the adopted standard by most of the metadata communities; in particular, the Resource Description Framework (RDF) [6] is an application of XML specifically designed to aid in the expression of entity-relationship data

[4]INDECS – Interoperability of Data in E-Commerce Systems

[5]the charter for the Dublin Core Citation Workgroup can currently be found at URL: http://purl.org/dc/groups/citation.htm

models. RDF mappings for the Dublin Core have been recommended [32], with the particular application of qualifiers for journal articles in the works by the Citation Workgroup mentioned above. The DOI-X metadata set currently uses plain XML as its syntax, with the requirement for a direct mapping to RDF. In general, closed-environment metadata, exchanged between entities knowledgeable of a specialized vocabulary seem to work well with XML, while generic solutions need RDF because of its greater semantic power.

2.4 Citation Linking on the Web

When the necessity to provide online services became evident to those involved in the circulation of scientific literature (publishers, abstracting and indexing services, document delivery services and libraries), citation linking was implemented in closed environments delimited by organizational borders (e.g. publishers, such as Elsevier or Springer Verlag) or by academic discipline (e.g. Medline[6]). Such systems have their advantages, chief among them being the ease of control over their links - maintenance of URL-based solutions and proprietary metadata formats is entirely possible. However, article citations transcend organizational or specialization borders, and providing links from one (closed) system to another has become a necessity, as is the the interoperability of their metadata.

Most of the online linking solutions conform to the general model described by Caplan and Arms [19]. In this model, getting from citation to article involves two conceptual steps:

1. *Reference lookup*: the citation is used to query a *reference database* in order to obtain one or more identifiers for the item.

2. *Resolution*: the identifiers are used as keys in a *location database* in order to obtain URLs to the item's content.

The reference lookup step is made difficult by the fact that citations in scientific journals are expressed in relatively unstructured natural language, with a high degree of variety in syntax. A search in a reference database involving keywords found in the citation is necessary in order to obtain bibliographic metadata. A minimal required metadata set should contain enough information to enable the reconstruction of a complete citation; DOI-X and Dublin

[6]Medline is a bibliographic database developed by the U.S. National Library of Medicine. More information about Medline can be currently found at URL:
http://www.ncbi.nlm.nih.gov/entrez/query/static/overview.html

Core are examples of such metadata sets. The set may also contain identifiers to be used in the resolution step.

When unique identifiers are contained within the citation (or can be directly and unambiguously calculated from it), the reference lookup step is bypassed, leaving the resolution step as the only action to be taken. The success of this action is not always guaranteed, due to access restrictions or unavailability of the article in electronic form. Even when resolution is possible, the result set may contain pointers to multiple copies of the same article (or multiple services supplying article content or information), residing at different locations which offer various qualities of service. The properties associated with each copy and the service supplying need to be described in order for the user (or the user's software) to make a choice as to which copy should be retrieved. Therefore, metadata is still needed in the resolution step, whether it is presented to the user for manual selection, or interpreted by the users's software based on preferences (e.g. cost of retrieval, current service load or bandwidth, etc.).

The above model implies that the linking service is decoupled at least in concept from the service providing access to the articles, even though they might be operated by the same organization (e.g. a publisher). The model is also known as a *static* linking model [23], because the links (or their components) are precomputed and stored in the location database.

The way links are computed in systems following the static model varies. In the DOI-X system, publishers submit bibliographic metadata to the service, where DOIs are computed and returned to the publishers, who subsequently embed them in their articles and citations. During resolution, the client software queries the location database with the DOI as a key, obtaining URLs to the text of the article. In the Web of Science® abstracting and indexing system [8], publishers submit URLs to the fulltext along with the rest of the article information.

The *dynamic* linking approach, used by the PubMed[7] and S-Link-S [28] systems involves the building of link specifications – templates from which URLs can be algorithmically built and subsequently embedded in texts.

The Open Journal project [29] also experimented with the dynamic linking of electronic journal articles in a post-hoc manner: the article content was served through a specialized

[7]An overview of the PubMed system can be currently found at URL:
http://www.ncbi.nlm.nih.gov/entrez/query/static/overview.html

proxy which retrieved the text from its original site (e.g. publisher), parsed it and superimposed URLs where appropriate. The project used processing entities called *citation agents* to parse language structures; the agents were tuned to the house style of publication and document structure. For metadata, a bibliographic database was provided by a commercial vendor (ISI), covering only the most significant journals in a discipline. Citations outside the closed universe (about 60%) did not have links to fulltext; instead, a link to metadata (abstracts) was provided. The implementors recognized in their final report [30] the instability problem introduced by the use of URLs and advocated the use of application-independent, stable ways to identify articles. On-demand calculation and embedding of links did not seem to go well with some users who found the service too slow, as stated in the final report. However, the link service has the advantage that the application of links does not require editorial control over the articles.

Another example of dynamic linking is the SFX system [24, 25], which approaches the citation linking problem from a different perspective, in that it starts with bibliographic metadata in general (whether coming from an article citation or a different source, e.g. a database record) and attempts to provide links to all the services that are relevant to it. The SFX system uses just-in-time resolution: when bibliographic metadata is presented to the service, it attempts to match it against descriptions of services stored in a database. All the relevant services that match the bibliographic metadata are presented as the result, but links to them are only constructed when the user attempts to select one of them; as a consequence, links may not always be valid. The SFX system links together library resources and Web sites with academic relevance through a unified interface.

Since the accessibility and quality of bibliographic material largely depend on who the user is, a useful citation linking system should allow the selection of the most appropriate information; in doing so, the system would implement a solution to what is known as the "Harvard Problem" [18]. Generally, the user identity can be detected either by storing user profiles with a global resolution service, or by employing local resolution services and having the user choose one of them. The former solution presents maintenance problems which make it less likely to be implemented [19]. The latter solution has been used in the "SFX@Ghent and SFX@LANL" experiment [25], where the service links that are presented in response to a query depend upon the user's environment (i.e., the user is a patron of the Ghent library or the Los Alamos National Laboratory digital library) and the existence of a local resolution service.

Research libraries are a prime candidate for services that provide local resolution, mainly because they offer a plethora of online bibliographic services aside from the classical library catalogue: fulltext access to electronic journals via site licenses, discounts for document delivery, interlibrary loan, access to abstracting and indexing information, etc. Libraries may also participate in consortia – forums whose goals are generally targeted towards improving their services through resource and knowledge sharing, thus enlarging their coverage. Since all this information is localized and readily available to library patrons, a citation linking solution deployed by a library seems to be a natural approach.

The customary way to access library information sources is via separate interfaces; unification of all these interfaces into a one-stop shopping solution has been the goal of several library projects (e.g. Library of Congress[8] or SFU[9]). One missing feature is the ability to link directly from an article citation into a library service. Linking via identifiers into a library catalogue is not generally possible at the article level, since the lowest granularity usually offered by library catalogues is the journal issue, for which holdings information is obtained by searching into a bibliographic index. Linking to extended library services is possible if the resolution service can be customized to compute links based on available local information: the resolution service provides the basic citation data which is then fed into the library service to obtain information at the article level (e.g. interlibrary loan or document delivery forms). If article-level resolution is not possible, a local resolution system would be useful even if it only provided links to the entrance pages of each service and let the user perform the search from there.

2.5 Citation Linking with BIBP

The BIBP Framework is a Web-based citation linking system which uses USINs as identifiers and the BIBP protocol as its communication mechanism. The framework intends to demonstrate an improvement over the existing practice by focusing on citation linking in the context of the World Wide Web, without considering it a particular case of a more general problem. The focus on citation linking is present in all the essential three areas: identifiers, metadata and resolution mechanisms.

The framework constitutes an *open* citation linking system, in which any institution can

[8]The home page for the Library of Congress is currently at URL: http://www.loc.gov/

[9]The homepage for the Simon Fraser University Library is currently at URL: http://www.lib.sfu.ca/

participate by deploying a local BIBP server. USINs are not based on proprietary identifiers and can be freely built without looking them up in a proprietary database; therefore, their usage is not confined within any organizational or specialization border. The framework can be categorized as a *dynamic* linking system because similarly to the SFX system, the links to services returned in response to a resolution request are dynamically constructed from XML link templates that follow a syntax close to that of the PubMed and S-Link-S systems.

In the BIBP framework, the reference lookup step is extremely simplified by the fact that only identifiers for serials need to be looked up given a standard bibliographic citation. In many cases, identifiers are readily available from the copy of an article an author cites, or because the serial has been assigned a mnemonic universal name that is widely known and accepted in the field of science to which the article belongs (e.g. `TOPLAS`). Therefore, the reference database is arguably not needed in most of the cases, or if needed it can be implemented with a minimum of effort.

The BIBP resolution step does not return URLs pointing to a copy of the item, as in the Caplan and Arms model; instead, citation metadata is directly presented to the user. The citation metadata may contain URLs pointing to the item content, if URLs are available; but more importantly, the result of a resolution query in the BIBP framework is constituted by (1) basic bibliographic metadata accompanied by (2) information about and links to citation metaservices. Citation metaservices are Web-based services that can offer extended information about the item, including but not restricted to, the item content. This is one of the most important characteristics of the system: instead of directly storing and serving extended item information, the BIBP framework stores basic information and service descriptions - metadata about services relevant to the item, rather than about the item itself. Metadata is presented to the client software in a format of choice, leaving it to the user to decide which links to follow.

Another important aspect of the BIBP framework is the solution given to the multiple resolution ("Harvard") problem. The construction of the BIBP protocol is such that the selection of resolution services by a client is flexible: a *local* resolution service is used by default; authors of online documents can designate a *preferred* resolution service that may be used to glean additional information or in case the local service is not available; and finally, if none of the above services are available, a *global* resolution service is used. The philosophy behind the design of the Harvard problem solution in the BIBP framework is to insure maximum configurability while maintaining user friendliness by employing appropriate defaults.

Chapter 3

The BIBP Framework

3.1 Requirements

The main requirement for the BIBP framework is to serve as a technology for the support of article citation linking with USINs in a Web environment. In order to fulfill this requirement, the framework interacts with various actors, whose roles can be categorized under three major types:

1. *Users*: people doing research work, who are the main beneficiaries of the system; they want to get as much information as possible about an article. They submit USINs to the BIBP framework for resolution and get article metadata in return.

2. *Contributors*: individuals or organizations uploading bibliographic data to the framework. Examples: publishers, libraries, document delivery services, bibliographic database owners, individual contributors.

3. *Bibliographic services*: on-line services that can be consulted to provide data about articles or article contents. Examples: fulltext databases, document delivery services, citation indexes.

Note that one entity may assume more than one of the roles described above; for instance, a researcher might assume the roles of a user and contributor, or a publisher may act as both a contributor and a service.

The need to interact via the World Wide Web translates into the requirement that USINs be usable in HTML documents. Therefore, USINs need to be made actionable within today's most popular browsers. An additional, non-functional requirement imposed to the system is that it should be easy for authors and regular Web users to create HTML pages containing USINs and follow the hypertext links created by actionable USINs.

The BIBP framework needs to support multiple resolution. As a consequence, it requires (1) technical means for a local institution to deploy and maintain BIBP services and (2) a mechanism that enables users to choose a particular resolution service (for instance a server deployed by a local library). Non-functional requirements for the selection mechanism are simplicity of implementation by network administrators and an intuitive user interface.

Contributors require means for uploading citation data. The BIBP framework should publish submission formats for contributors and define uploading mechanisms. The submission formats should be easy to understand and use; if there are commonly accepted bibliographic formats for citations (such as BIBTEX), then try to accommodate them; otherwise, define the formats in a standard data interchange format, such as XML. The submission formats should also accommodate article reference information (i.e. a submission may contain bibliographic data about the article itself and for all the articles referenced by it). If this requirement is fulfilled, a universal citation database can be built; the benefits of having such a database are described extensively in [15].

Bibliographic services that wish to participate in the BIBP framework require the means to (1) describe their services and (2) specify whether a certain service is applicable to an article. The latter requirement is analogous to specifying holdings for a library; therefore, the USIN holdings format [16] may be used in its resolution.

The type of citation links supported by the framework is limited initially to articles published in registered journals (i.e. journals having an ISSN number) and some institutional reports. While this represents a significant amount of literature (there are almost one million records in the ISSN Register at the time of this writing [1]), the framework is required to be extensible, in order to embrace other types of serial publications, such as technical reports, pre-prints and books.

A consequent requirement is that the BIBP system be scalable, to accommodate the large amount of information that needs to be stored and manipulated. It was estimated [37] that in 1995 there were about 660,000 scientific, technical and medical articles (STM) written by USA authors only. Given that the number of US journals accounts for 35%-40%

[1]Source: ISSN International Centre Web page, currently at: http://www.issn.org/

of the world's STM literature, it can be assumed that there are around 2 million articles written every year. If bibliographic information was stored about each of these articles for a span of 30 years, it would yield a ballpark figure of 60 million article records. These articles arguably cover a very important part of the literature; however, if books, conference papers, technical reports and other serial items are also taken into consideration, it can be assumed that the storage requirements go beyond the capabilities of a single workstation.

Thus, a distributed architecture is required, where information is spread across different computing systems, precisely in order to cope with any scalability problems, both in terms of storage capacity and performance.

One of the initial goals of the BIBP framework is to provide a simple paradigm for citation linking, with concepts and technologies readily available to those interested. Layers of sophistication may be added later if the system is widely adopted. A consequent requirement is that the system specification follow levels of complexity that allow for evolution and staged implementations.

A final non-functional requirement is that the system should adopt and leverage widely recognized Internet standards, in order to ensure interoperability with other entities on the Web and to take advantage of existing open-source software.

3.2 The BIBP Network

The BIBP network represents an association of servers and clients supporting the resolution of USINs, acquisition and dissemination of bibliographic information, and load sharing. Figure 3.1 presents a schematic view of the network, identifying the main components and their interconnections. The BIBP terminology is built around the client-server model, as follows:

- *User Agents* are client processing entities which can communicate with a BIBP server chosen by the user, via the BIBP protocol. The user agents submit USINs for resolution and retrieve bibliographic metadata for the purpose of displaying it on the user screen. The term "user agents" is used instead of "clients" to illustrate the fact that their main applications will be as part of Web browsers and also because there are other processing entities in the BIBP framework that may assume client roles.

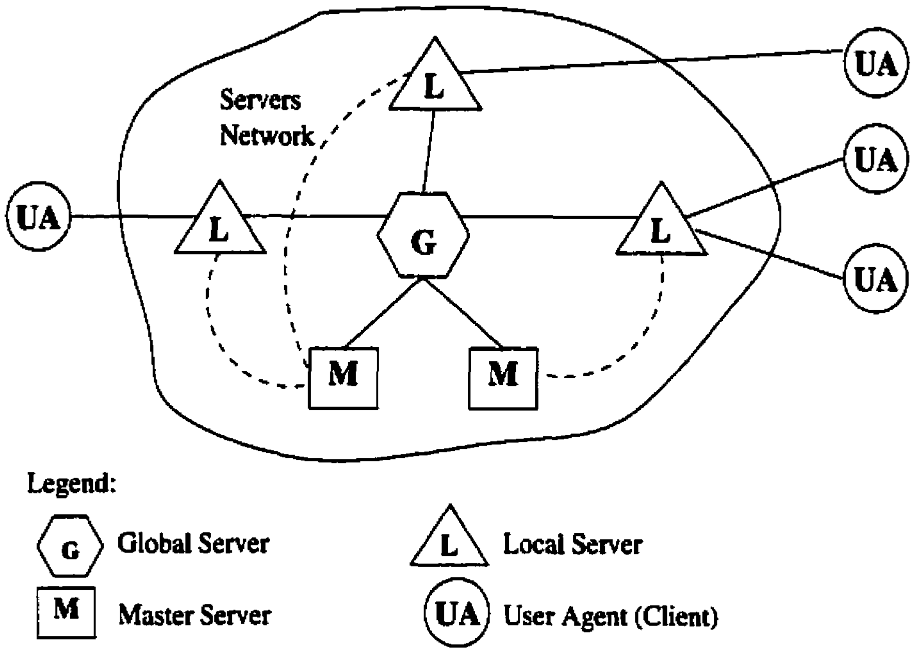


Figure 3.1: BIBP Network Topology

- *Servers* are processing entities capable of resolving USINs to bibliographic information and accepting contributions. There are three flavours of BIBP servers: *global*, *master* and *local*; they all cooperate to ensure that a shared, distributed database of bibliographic information is being built and maintained consistent. Note that a server is not identified with one single computer; instead, a server may be distributed over many computing systems, as necessities dictate. Also note that the level 1 of the BIBP protocol only specifies the place of local and global servers in the framework; the discussion in this chapter however presents the design of the BIBP system as supported by a level 2 implementation of the BIBP protocol.

In order to better illustrate the operations performed by the servers, each server is assumed to implement two primitives: `RESOLVE(USIN)` and `SUBMIT(metadata)`, where the `RESOLVE` primitive resolves a USIN to a set of metadata, while the `SUBMIT` primitive arranges for the storage of bibliographic and service metadata somewhere on the network.

3.2.1 User Agents (Clients)

A user agent is a processing entity capable of resolving USIN-based hypertext links via the BIBP protocol. The user agent may be a Web browser application or an application running inside a Web browser, such as a script, applet or plugin; or it may constitute parts of a document preparation system, such as BIBTEX or Microsoft® Office.

Under the level 1 of the BIBP protocol, there are only two strict requirements for a compliant user agent, which have been formulated based on the features offered by current browser technologies, in order to enable at least the construction of script-based user agents. The first requirement is that a user agent be able to choose an appropriate BIBP server for the resolution of a BIBP link, according to the following algorithm:

- If the host name `bibhost` can be resolved under the Domain Name System (DNS) to an Internet Protocol (IP) address, and if the host employs a BIBP server, then use it for resolution.
- If the previous step fails to provide a resolution server, check whether the document specifies a preferred server. If such a server name is present and it can be resolved to an IP address and it implements the BIBP protocol, then use it for resolution.
- If the previous two steps fail, use a global BIBP server.

The name `bibhost` represents a DNS alias used to locate a BIBP service within a particular Internet domain, similar to the `mail` or `www` aliases used to implicitly resolve e-mail addresses or URLs to hosts that do not employ the names `mail` or `www` respectively. This DNS lookup mechanism is instrumental in the distribution of BIBP, since the `bibhost` alias can be created by configuring only the local DNS name server, without requiring any changes to the DNS configurations on individual client machines.

The second requirement is that a user agent be able to formulate a BIBP request for the resolution of a USIN to a page containing bibliographic metadata marked up in the Hypertext Markup Language (HTML). In the level 2 of the BIBP protocol, a user agent may choose to request other markup formats for the metadata, such as BIBTEX, XML, or RDF.

The BIBP protocol allows authors of documents containing USINs to specify a *preferred* resolution server for their documents. A user agent is required to encapsulate the address of this server inside every BIBP request that it formulates. The resolution server may then use the preferred server to garner more bibliographic information or to simply redirect the request to it.

3.2.2 Global Servers

The global BIBP server is the hub of the BIBP network; this server (or system of servers) is also the starting point when building the framework.

The global server is the default point of service for any USIN-based query. User agents are required to contact this server whenever they cannot contact any other server. Local servers are also required to query the global server whenever they cannot resolve a USIN. When resolving a query, the global server attempts to find an authoritative master server for the USIN, and if one is found, the query is redirected to it (in this capacity, the global server acts as a Resolver Discovery Service - section 2.2.1). Local servers store or cache master server locations in order to minimize the number of requests made to the global server and the number of hops required for resolution.

A second function for the global BIBP server is to act as the default point of service for all the contributions of bibliographic information. If there are authoritative master servers covering the journal articles contributed, then the contributions are redirected to them; otherwise, the information is stored locally and later offloaded to the appropriate master servers as they become available.

The primitive operations performed by the global BIBP server are illustrated in the following segment of pseudo-code.

```
RESOLVE( USIN )
BEGIN
   m = find_authoritative_master_server( USIN );
   if( found( m ) )
      r = m->RESOLVE( USIN );
   else
      r = find_bibliographic_record( USIN );
   if( found( r ) )
      return r ;
   else
      emit_error();
END
SUBMIT( metadata )
BEGIN
```

```
   m = find_authoritative_master_server( metadata );
   if( found( m ) )
      m->SUBMIT( metadata );
   else
      store_metadata( metadata );
   emit_submission_results();
END
```

3.2.3 Master Servers

Master servers are BIBP servers dedicated to a group of serial publications; their purpose is to serve bibliographic information for a number of journals, thus taking some of the load off the global server. It is envisioned that master servers will typically be deployed by digital libraries or publishers and will serve bibliographic information for one to several hundred journals.

There may be more than one master server for a particular journal, due to mirroring or overlaps; the decision of which one is to be used for resolution can be taken in order by the user agent, local servers and finally by the global server, based on metrics stored in metadata records for each server, such as proximity, reliability, etc.

Master servers can be authoritative or non-authoritative; the distinction is made based on the quality and comprehensiveness of the served data, and the reliability of the institutions deploying them. Establishing an authoritative server is a matter of social engineering rather than a technical one, but one possible solution is to find trustworthy institutions willing to commit to the long-term upkeep of such a server (e.g. publishers or national libraries). The global server always chooses an authoritative master server to redirect a request; however, local servers may be configured to choose a master server that is best suited to the needs of the local institution.

The following segment of pseudo-code illustrates the primitive operations performed by a master server.

```
RESOLVE( USIN )
BEGIN
   r = find_bibliographic_record( USIN ) ;
   if( found( r ) )
      return r ;
   else
      emit_error();
```

```
END
```

```
SUBMIT( metadata )
BEGIN
   if( is_applicable_metadata( metadata ) ) {
      store_metadata( metadata );
      emit_submission_results();
   } else {
      emit_error();
   }
END
```

3.2.4 Local Servers

Local servers are deployed in order to provide caching and specialized services for an exclusive group of users. For instance, a library would deploy a local server in order to provide its patrons with an integrated linking solution for its catalogs and other services. The Simon Fraser University Library is such an example: the library has developed a service called Generalized Online Documents, Ordering and Texts (GODOT) [34], which allows members of the SFU community to use the same interface for document access in the library or associated libraries, inter-library loans and document delivery. If the library were to deploy a local BIBP server, then the SFU community would have access to this service directly from documents containing USINs as hypertext links.

A local BIBP server is bootstrapped using only bibliographic information relevant to its holdings. After the initial data has been entered, the deployers submit metadata for the particular services that they have to offer to their user communities. The server is then ready to resolve USINs and can be automatically selected by a user agent through a Domain Name System (DNS) lookup as discussed in section 3.2.1.

For those USINs which cannot be resolved locally, the server will contact one of the master servers for the publication identified by the USIN, or the global server if no master server is found. The local server may choose to store the bibliographic information obtained this way for caching purposes or to update its core data.

A local server may choose to accept contributions and store them locally, or submit them to the global server, or both.

The following segment of pseudo-code illustrates the primitive operations performed by a local server.

```
RESOLVE( USIN )
BEGIN
   r = find_bibliographic_record( USIN );
   if( found( r ) ) {
      return r ;
   } else {
      m = find_master_server( USIN );
      if( found( m ) ) {
         r = m->RESOLVE( USIN );
      } else {
         g = get_global_server() ;
         r = g->RESOLVE( USIN );
      }
      if( found( r ) ) {
         store_bibliographic_record( r );
         return r ;
      } else {
         emit_error();
      }
   }
END

SUBMIT( metadata )
BEGIN
   if( is_bibliographic_metadata( metadata ) ) {
      store_bibliographic_metadata( metadata );
      m = find_master_server( metadata );
      if( found( m ) ) {
         m->SUBMIT( metadata );
      } else {
         g = get_global_server();
         g->SUBMIT( metadata );
      }
   } elsif( is_service_metadata( metadata ) ) {
      store_service_metadata( metadata );
   }
   emit_submission_results();
END
```

3.3 BIBP Metadata

The BIBP metadata is designed to support an efficient implementation of the two primitive operations performed by its processing entities: resolution of USINs and contributions of bibliographic information.

It is envisioned that the following types of metadata will be necessary for USIN resolution:

- basic information about a serial: title, format, publisher, other identifiers and publication pattern.
- basic information about a serial volume or issue: date, pages, tables of contents.
- basic information about a serial item: title, authors, publication facts, other identifiers.
- extended information about a serial item: abstract, index keywords, subject classifications, etc.
- citation information for a serial item: items cited by it, and items citing it.
- service metadata: information about the services available with respect to the item queried (e.g. libraries, document delivery services, abstracting and indexing services, fulltext databases, document delivery, translations, etc.)

Note that for illustration purposes, the serials are arranged into volumes and issues. While this is the most widely used scheme with respect to STM journals, it is by no means the only one, as discussed in section 2.2.5. The BIBP framework is designed to accommodate any kind of serial item identifiable by a USIN.

The basic bibliographic information about serials, volumes, issues and articles constitute publication facts which are not copyrightable; however, some of the extended information for certain serial items (e.g. abstracts) may be copyrightable. The implementations of the resolution and contribution operations have to take into account the possible intellectual property issues that may exist. In that respect, although the framework allows the circulation of all the above types of metadata, the enforcement of copyright policies is left to the local deployers of BIBP servers; the global server will not present to the users any data that is subject to copyright laws, unless specifically permitted by the rights owners.

The following types of contributions and their associated data are envisioned in the initial phase of development:

- contribution of serial information: serial title, identification codes, abbreviations, publisher information.
- contribution of issue information: serial identification, issue identification (volume number, issue number, chronology), issue contents (article information).
- contribution of serial item information: title, authors, publication identity (volume, issue, page/article number), the list of references.
- contribution of service information: descriptive metadata about a bibliographic service, allowing the user to establish the way to access it and obtain information about a serial item.

The most challenging task with respect to the design of contribution metadata sets is that of service description. While it is impossible to identify all the bibliographic service types and access patterns, the design seeks to provide an element set that is generic enough to cover a broad range of services. The following properties of a service are described:

- The type of service: associated library, publisher, document delivery service, abstracts and indexing service, bibliographic database, etc.
- The information that can be obtained from the service: fulltext, abstracts, bibliographic records.
- The media types used to store the information: paper or electronic.
- The access to the service: public or restricted.
- The costs associated with the retrieval of relevant information: free, pay-per-view, etc.
- Holdings information: what serials are covered, and what is the coverage, down to individual issues. This information is necessary to establish the relevance of a service to a serial item.
- Internet links to the service: if it is possible to construct a link directly to the page on the service provider's site that is relevant to a particular serial item, then a description

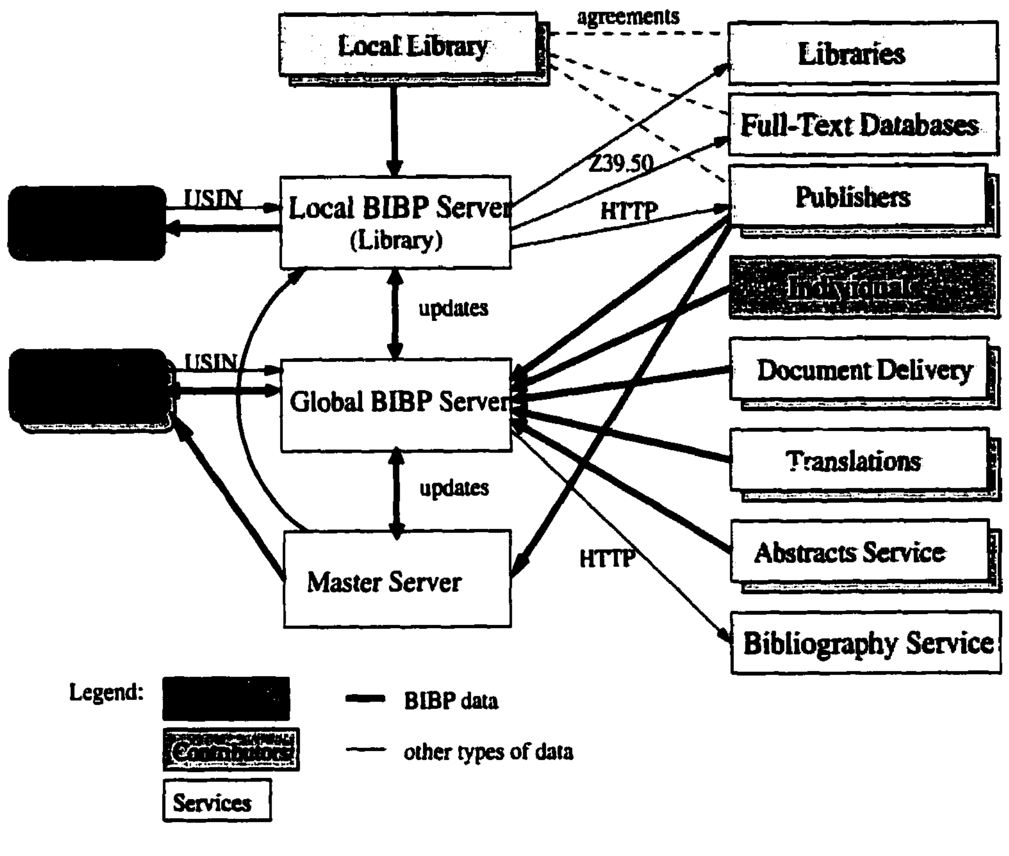


Figure 3.2: Dataflow in the BIBP Framework

of how to build the link is needed. Two approaches served as inspiration for the construction of link template metadata: S-Link-S and LinkOut[2].

The metadata flows along the paths depicted in Figure 3.2. BIBP servers may obtain metadata from a local store, or by contacting other services using various protocols (e.g. Z39.50 [4] or HTTP). For communication between BIBP servers, data is encoded in XML and carried over the Internet by the BIBP protocol. User agents communicate with BIBP servers using the BIBP protocol, in order to retrieve metapages encoded primarily in HTML, with additional encodings provided by the level 2 of the protocol.

The BIBP metadata sets are designed with interoperability in mind. As seen in section 2.3, there are various schemes for the exchange and description of bibliographic information

[2]LinkOut is a system developed by the U.S. National Library of Medicine for the linking of PubMed records to document providers. More information can be currently found at URL: http://www.ncbi.nlm.nih.gov/PubMed/linkout.html

using XML, such as DOI-X and Dublin Core, but they are all work in progress. Before a standard emerges, the intention is to use metadata schemes that are as close as possible to these similar approaches. The BIBP metadata design will adopt an open and neutral attitude towards other efforts, with the hope that convergence will be achieved in the near future.

3.4 Acquisition of Bibliographic Information

The BIBP framework relies on a distributed database of bibliographic information. The database is built in a piecemeal fashion using three strategies: (1) a contribution-based strategy, (2) an agent-based strategy and (3) a discovery-based strategy.

Ideally, authors of scientific papers would contribute to the database by submitting bibliographic information about every paper they write, including all the references. The benefits of such an endeavour have been discussed extensively in [15] and section 2.2.5. From a technical standpoint, this act would also guarantee that all USIN-based hypertext links contained in on-line papers work (i.e. the USIN resolution would yield at least basic bibliographic information).

Publishers are in position to contribute a fairly rich set of metadata about their journals. Most of the large publishing houses already offer on-line access to the fulltext of some of their journals, and free access to bibliographic information such as tables of contents and article abstracts. It is expected that the move towards publication in electronic formats and on-line access to scientific literature will continue in the future. Therefore, publishers should be interested in a portal service that channels traffic to their sites. By simply submitting descriptions for their on-line access services to the framework, links are automatically created (via USINs) for all the articles they ever published or will publish.

The BIBP framework cannot rely solely on a passive approach to information acquisition, as given by the contribution process. A pro-active strategy has been envisioned whereby information is collected from two types of sources: one is constituted by on-line services such as publisher sites or bibliographic database, while the other one is the World Wide Web at large. Acquisition from on-line bibliographic services is done via specialized *agents*, capable of downloading and extracting relevant information from Web pages or other data structures, whereas acquisition from the World Wide Web involves the use of discovery robots and search engines. Both types of acquisitions can be done in two ways: asynchronously with respect to user requests, or synchronously (i.e. the agent is launched on demand, when a resolution is requested). It is expected that the agent-based approach would yield a larger "catch", since it works with specialized sites, while the search engine approach is targeted at giving the user some orientation as to where to look for more information.

Since bibliographic information is entered in the database from a wide variety of sources, the quality of data should be questioned. While there is not much the framework can do to prevent attempts at contributing incomplete or erroneous data, a BIBP server will perform several checks before entering it in the database, ranging from simple sanity checks based on existing data (e.g. non-existent serial, or page overlaps) to inferences made based on publication patterns for serials (e.g. such an inference would detect that a particular issue or page number cannot exist because the serial enumeration does not allow it). The BIBP server will also attempt to perform unification of partial records from previous [incomplete] submissions, until valid bibliographic structures can be created and entered into the database.

3.5 The BIBP Protocol

BIBP is an application-level protocol used for communication between the processing entities of the BIBP framework. The protocol is built on top of the Hypertext Transfer Protocol (HTTP) [26], in order to take advantage of the current Web client applications and the wide support for HTTP in various security applications. One of the most compelling reasons to build on top of HTTP is that its traffic can pass through most of the firewall solutions in operation today. Building on top of HTTP also has the advantage of simplicity and leverages a large existing codebase, providing the BIBP implementors with a multitude of technologies to help them write BIBP applications.

As mentioned in chapter 1, the protocol is divided into two levels of complexity, to ensure that a basic (Level 1) BIBP solution can be implemented with a minimal effort. The first level specifies only interactions between user agents and BIBP servers, while the second level adds specifications for the communication mechanisms that are to be used by BIBP servers in order to implement the `RESOLVE` and `SUBMIT` primitive operations.

The protocol specification consists of rules and recommendations. Rules specify interfaces, data formats and modes of interaction that must be obeyed by compliant BIBP entities in order to ensure interoperability between BIBP software developed by different parties.

Recommendations establish optional interfaces and efficiency guidelines that implementors may use; they may become rules in a later revision of the protocol. For instance, there are rules that must be obeyed by a compliant user agent in order to find a resolution server, as seen in section 3.2.1; however, the implementations of the `RESOLVE` and `SUBMIT` primitives as illustrated in sections 3.2.2, 3.2.3 and 3.2.4 have a recommendation status - implementors are free to implement the primitives however they consider appropriate, as long as the rules governing the format of the BIBP requests and responses are satisfied.

Similarly to other protocols, the BIBP specification will not be considered complete until a reference implementation has been finalized; until then, drafts may be released in order to stimulate discussions and help the implementation process. Thus, the examples given below serve only to illustrate the workings of BIBP without claiming full compliance with the protocol in its release form.

3.5.1 User Agent to BIBP Server Communication

A BIBP user agent must be able to formulate resolution requests and direct them to an appropriate server. Once the identity of the BIBP server to be contacted has been established, the user agent generates a well-formed HTTP request to that server. Suppose the user agent attempts to resolve the following BIBP link:

```
<A HREF="bibp:ISSN/0362-5915:17(3)@513">USIN</A>
```

The user agent uses the algorithm described in section 3.2.1 to obtain the name of a BIBP server - assume that `srv.domain` is that name.

If the author of the document containing the above link has specified a preferred resolution server, for example `prefsrv`, the user agent will translate the above link into the following URL:

```
http://srv.domain/bibp1.0/resolve?citehost=prefsrv&usin=ISSN/0362-5915:17(3)@513
```

In the absence of a preferred server, the URL becomes:

```
http://srv.domain/bibp1.0/resolve?usin=ISSN/0362-5915:17(3)@513
```

In its simplest form (Level 1), the response from the server comes in the form of an HTML page describing the article denoted by the above USIN and offering links to local library holdings and other services (e.g. links to the ACM Digital Library that point to the article abstract, reviews and the fulltext). A request to a server implementing the level 2 of the protocol may contain additional parameters that specify a different encoding for the returned data. For instance, the URL:

```
http://srv.domain/bibp2.0/resolve?usin=ISSN/0362-5915:17(3)@513&markup=bibtex
```

would request a BIBTeX record for the article, possibly for embedding in a document (the agent may be part of a document preparation system). The response would come encoded in the `text/plain` media type, as follows:

```
@Article{ ISSN/0362-5915:17(3)@513,
author  = {Robert K. Abbott and H{\'e}ctor Garc{\'\i}a-Molina},
title   = {Scheduling Real-Time Transactions: {A} Performance Evaluation},
year    = {1992},
journal = {ACM Transactions on Database Systems},
volume  = {17},
number  = {3},
month   = {September},
pages   = {513--560},
}
```

Contributions from a user agent may be made via an HTTP request using the `POST` method, to a URL that specifies the submission method and the markup of the submitted metadata, e.g.:

```
http://bibpserver/bibp2.0/submit&markup=xml
```

The media type of the contributed bibliographic metadata is `multipart/form-data` [35]; this encoding allows for a simple implementation of submission as an HTML page that contains a file upload form.

3.5.2 BIBP Server to Server Communication

Depending upon implementation, BIBP servers may communicate with each other to support requests for resolution and submission. A server may assume a client role and issue a USIN resolution request to another server, following the same request encoding rules as the user agent. However, the data exchanged between BIBP servers uses XML markup only – no other formats are supported.

In order to distinguish between the requests made by servers and those made by user agents, servers use the verb `query` instead of `resolve` when forming the HTTP requests. For instance, a server may open an HTTP connection to another BIBP server and issue the following HTTP request using the GET method [26]:

```
GET /bibp2.0/query?usin=ISSN/0362-5915:17(3)@513 HTTP/1.1
Host: srv.domain
... (omitted HTTP headers)
```

The response from the contacted BIBP server may be in the form of article metadata (including service information), or it may be a redirection – an HTTP response with the code 302, indicating that the requested resource resides under a different URI. A redirection response may be given by the global server in order to point a local server to a master server. Future resolutions of the USIN by the local server should not involve the global server, as the location of the master server should be remembered by the local server.

The `query` verb has other envisioned uses besides article metainformation retrieval and redirection information. One of these uses involves the exchange of capabilities between servers: the server initiating the request may add headers describing its characteristics (e.g. physical location, authentication or authorization information); the server receiving the request may then choose its response based on this information. For instance, BIBP servers operated by associated libraries may cooperate better if they could authenticate each other beyond basic host/domain comparisons; thus the notion of local resolution would be extended beyond the realm of a local organization.

Chapter 4

A BIBP Prototype

4.1 Introduction

One of the goals of this thesis was to implement a minimal BIBP system that would serve as a proof of concept for the ideas introduced by the USIN/BIBP framework and drive the further development of the BIBP protocol specification. The system would also serve as an evaluation testbed for various technical issues.

The project involved the development of client and server modules that allow the installation of a prototype global server which can also be used as a local server for the SFU community. The server provides a starting set of features that were deemed important for the proof of concept.

The coverage of the system is primarily journal literature, with a representative sample directly stored and a considerably larger set available through agent technology. Linking of electronic journal articles, technical reports and RFC publications (reports of the Internet Engineering Task Force) is also demonstrated.

In the initial phase, the granularity of resolvable bibliographic items is articles; for instance, resolution at the volume or issue level is not seen as an essential initial feature and thus is not supported. However, metadata at the journal, volume or issue level is accepted and stored in the database.

The implementation is level 1 compliant and adds several level 2 features, such as submission of article and service information and retrieval of metadata with various markups

(BIBTEX, XML, RDF). The server also supports dynamic acquisition of bibliographic information via agents; a prototype agent that links the PubMed system via USINs has been implemented.

4.2 User Agent Implementation

Since the current Web browsers do not have native support for the BIBP identification scheme and protocol, extensions have to be written to handle BIBP links. A simple way to resolve the BIBP links is to write a JavaScript applet that replaces them with HTTP links which are understood by the browser. This transformation (known as "URL munging" [48]) involves the building of a URL having a path component pointing to a BIBP server and a query component that encodes a call to the `RESOLVE` primitive on the server. The script can then be included directly in the source of an HTML document, or it can be referenced from it via standard HTML mechanisms.

In addition to creating URLs that encapsulate BIBP requests, the client script needs to choose a resolution server, according to the algorithm described in section 3.2.1. The security restrictions imposed to scripts running in Web browsers prevent them from accessing connection or network information. A HTTP extension mechanism (which is based on the exchange of extra HTTP headers) cannot be used with a scripting solution in order to determine the availability and capabilities of a BIBP server. Therefore, the BIBP protocol specifies that a BIBP server must identify itself by making available an image or a script located at pre-established locations, whose existence can be checked by a script.

Authors can specify a preferred BIBP server to be used in resolution by the local or global server. For the scripting solution, the authors need to set a script variable to the location of the server. If browsers become BIBP-aware (thus rendering the scripting solution obsolete), the location of the server can be specified via the standard HTML element `LINK`.

In the simplest case (i.e., no preferred hosts specified), the following steps can be used by authors in order to implement BIBP links in HTML documents:

1. Write the BIBP links in the body of the document, as regular HTML Anchor elements.

```
<HTML>
<!- ... (header omitted) -->
<BODY>
```

```
    <!-- ... content of the paper ... -->
    <!-- declaration of references: -->
    [Abbot-Molina92]
    R. Abbott and H. Garcia-Molina. Scheduling Real-Time
    Transactions: A Performance Evaluation. ACM Transactions on Database
    Systems, Vol. 17, No. 3, pp. 513-560, September1992.
    <A HREF="bibp:ISSN/0362-5195:170513">USIN: ISSN/0362-5195:170513</A>
    <!-- ... other references ... -->
</BODY>
</HTML>
```

2. In the header of the HTML document, the code of the resolver script must be pasted as shown below; alternatively, a reference to the script may be supplied via a URL.

```
<HTML>
<HEAD>
    <TITLE>Paper Title</TITLE>
    <!-- ... -->
    <SCRIPT TYPE="text/javascript">
    <!-- here goes the script code -->
    </SCRIPT>
</HEAD>
```

Full details of this scripting solution, including support for the optional preferred BIBP server and also showing the source of the resolver script are presented in Appendix A. The resolver script currently handles a representative variety of browsers, browser versions and operating systems; more work is required to ensure portability across the whole browser and operating system spectrum. Nevertheless, given the stringent limitations imposed by browsers with respect to URL and protocol handling, it can be argued that the script provides a fairly reasonable solution: the script is simple, compact, and completely transparent to the user.

4.3 BIBP Server Components

The server design follows a methodology in which modular components are assembled to build a particular type of server. The interactions between modules occur over well-defined interfaces, thus allowing for independent development of each module. Interoperability at the binary level is ensured by the usage of the Java language.

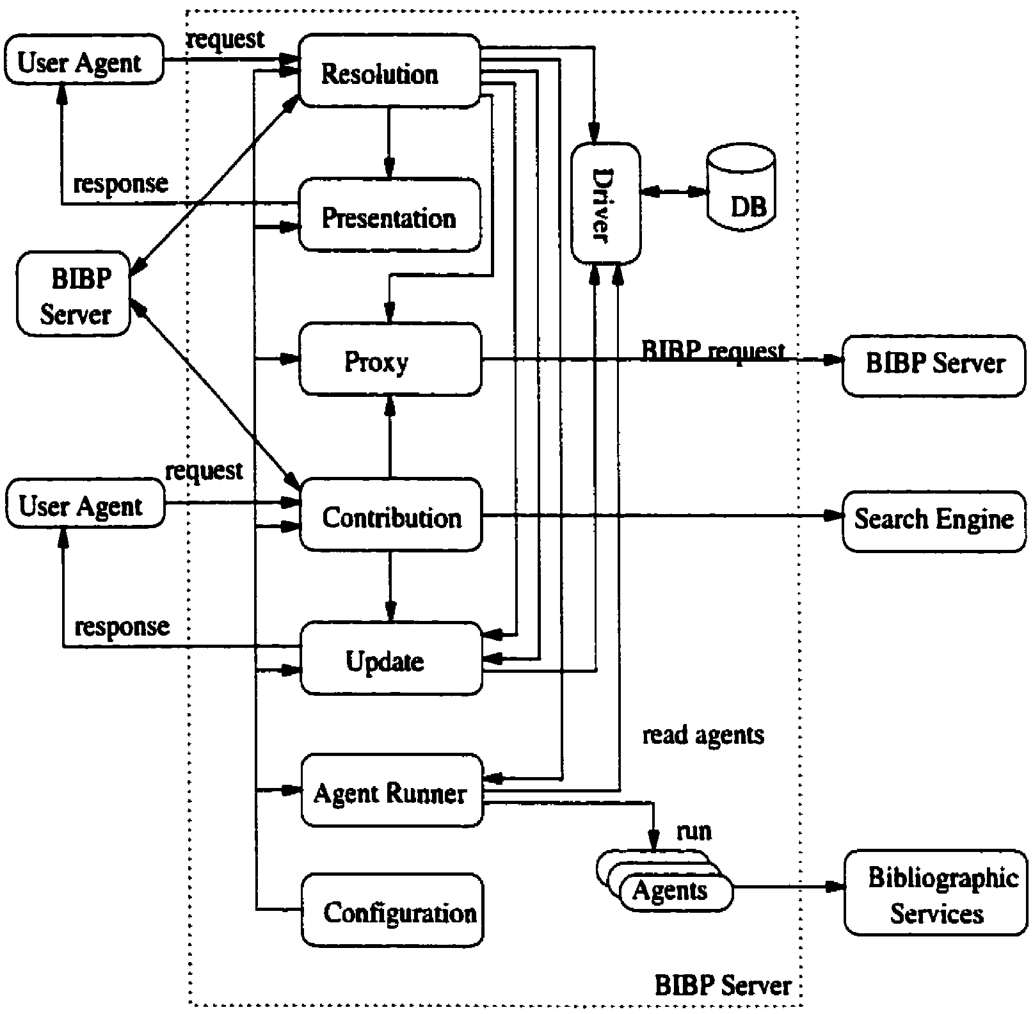


Figure 4.1: BIBP Server Modules

Figure 4.1 presents the core set of modules that can be put together to form a BIBP server. Depending on the server type (i.e. global, master or local), some modules may provide a different implementation or may be simply missing; for instance, a master server may not need to issue requests to any other BIBP server and therefore might not employ a proxy module.

The modules are glued together by a container responsible for request dispatching and response relaying. The implementation of the container is based upon Java servlet technology [21], which handles the details associated with the underlying HTTP protocol.

The interaction with the local database is done through the *Database Driver* module, a component designed to allow communication with various data storage implementations. The prototype system uses a flat file hierarchy to store bibliographic information, but a more sophisticated solution might deploy a relational database; the storage technology used does not affect the rest of the modules as they all use the driver, which operates with bibliographic records formatted in XML.

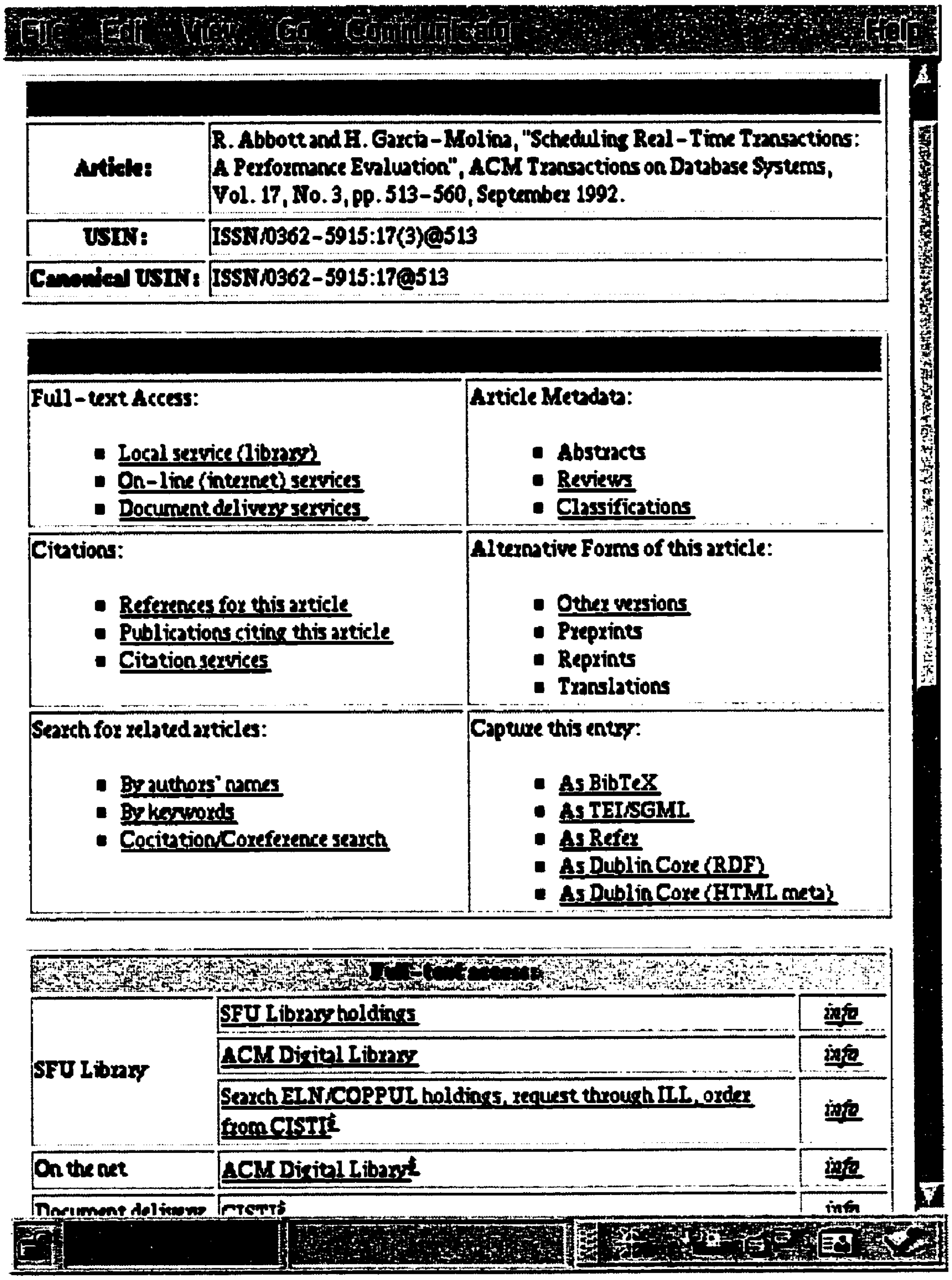
The *Configuration* module provides a user interface for the customization of the server. Parameters that specify how certain modules should run (e.g. whether information obtained via agents should be cached locally or not) or how the information is displayed (e.g. paths to stylesheets, icons, etc.) are set through this module. The configuration module exports an interface to the other modules, through which they can retrieve configuration options.

4.3.1 The Resolution Process

The *Resolution* module implements the `RESOLVE` primitive; the algorithm performed by its implementation depends on the type of server deployed, as presented in chapter 3. The resolution can be carried out locally by querying the database; alternatively, a remote BIBP server or site may be contacted to provide the information. Once obtained, the bibliographic metadata is then handed off to the *Presentation* module, which formats it according to the user's wishes; figure 4.2 shows an example of an HTML page returned by this module. In the case of metadata obtained from a remote site, the resolution module passes the data to the *Update* module, which may decide to store it locally.

The communication with a remote BIBP server is realized through the *Proxy* module, which encapsulates the remote communication details; this module creates and manages one proxy for each of the remote BIBP servers that need to be contacted. The references for the remote BIBP servers are retrieved by the resolution module directly from the database. The decision not to use a naming and discovery service was taken in order to keep the system simple; if the BIBP network evolves to the point where such services become necessary, changes can be made to the resolver module locally, without affecting the rest of the design.

The resolution module may contact the *Agent Runner* module in order to perform on-demand resolution from a non-BIBP server (typically a Web server, although servers based on other protocols, such as Z39.50, may be contacted). The Agent Runner module provides an execution environment for agents – programs customized to contact remote sites, fetch and parse bibliographic metadata. For instance, the PubMed system stores over 10 million



Article:	R. Abbott and H. Garcia-Molina, "Scheduling Real-Time Transactions: A Performance Evaluation", ACM Transactions on Database Systems, Vol. 17, No. 3, pp. 513-560, September 1992.
USIN:	ISSN/0362-5915:17(3)@513
Canonical USIN:	ISSN/0362-5915:17@513

Full-text Access: ■ Local service (library) ■ On-line (internet) services ■ Document delivery services	Article Metadata: ■ Abstracts ■ Reviews ■ Classifications
Citations: ■ References for this article ■ Publications citing this article ■ Citation services	Alternative Forms of this article: ■ Other versions ■ Preprints ■ Reprints ■ Translations
Search for related articles: ■ By authors' names ■ By keywords ■ Cocitation/Coreference search	Capture this entry: ■ As BibTeX ■ As TEI/SGML ■ As Refer ■ As Dublin Core (RDF) ■ As Dublin Core (HTML meta)

Full-text access:		
SFU Library	SFU Library holdings	info
	ACM Digital Library	info
	Search ELN/COPPUL holdings, request through ILL, order from CISTI	info
On the net	ACM Digital Library	info
Document delivery	CISTI	info

Figure 4.2: A fragment of a HTML page returned by the BIBP server

citations and abstracts from more than 4000[1] medical journals. Instead of acquiring and storing all this information in the BIBP database, PubMed was registered as a service with the BIBP framework. Since the system does not employ a BIBP server, an agent has been written that contacts the PubMed site and uses the bibliographic identifiers obtained from a USIN to retrieve an article citation. When a request for a journal served by the PubMed system arrives at the BIBP server, the resolution module contacts the agent runner module, which dynamically loads the appropriate agent and calls its resolution routine. The agent contacts the PubMed system and parses the returned information (in case of PubMed, the citation is encoded using the Abstract Syntax Notation One - ASN.1 [3]). Finally, the agent prepares an XML record to be returned to the Resolution module.

4.3.2 The Contribution Process

The *Contribution* module implements the `SUBMIT` primitive; depending on the type of server and type of submission metadata, the module may store the information locally or forward it to another BIBP server via the proxy module.

The contribution module needs to perform checks on the metadata before submitting it to the database via the update module. For the prototype server, these checks are very basic, as the data is not automatically entered in the database. Instead, the contribution module stores the data in a temporary location, where it is manually inspected; the updater module is also manually called, after the data is deemed correct. The reason for the insertion of a manual process is that since the system is not live yet, the only contributions are done by the developers of the system, who do not need to use the Web to post the data; in addition, research is being conducted to establish the algorithms for data checking and unification.

The contribution module is responsible for the conversion from various formats to the canonical XML format used throughout the BIBP framework. The update module can interpret this data and perform merges and unifications with the information stored in the database: whenever article information is entered, records have to be updated for tables of contents, reference lists and other structures; thus, the update module presents a higher level of intelligence than the database driver, which is merely responsible for reads and writes at the record level.

[1]Source: PubMed overview, currently at URL:
http://www.ncbi.nlm.nih.gov/entrez/query/static/overview.html

Bibliographic information is submitted at the article level, using XML or BIBTeX (other formats will be supported in the future). The XML format is similar to the DOI-X format used in the CrossRef project, but it is more comprehensive in that it allows the basic article information to be followed by a list of citations which, although optional, is highly preferred since it allows the system to perform citation searches. A sample submission file is given in Appendix B for the article presented in section 2.1.

Service description information is submitted using an XML format only. The description contains basic service information such as a textual description, home page URL, access descriptions, followed by a set of link specifications and a list of holdings for the journals covered. If the service is registered as a provider of basic bibliographic information via agents, then a list of agent descriptions is included. Appendix C provides examples of service submissions for the SFU library and the PubMed system.

4.4 The BIBP Database

The prototype implementation of the BIBP database stores the bibliographic information in XML files, organized in a hierarchical directory tree which is isomorphic to the USIN structure. The USINs are parsed into tuples whose elements are then mapped to directory and file names and then joined to yield a path to the location of a particular record. The size of the tuples can be varied, thus accommodating serial items identified by any kind of hierarchical numbering scheme.

The above is not a prescriptive design, as the choice of storage technology is left to the implementors, but for the purpose of prototyping, using the file system as an indexing engine and XML files as pseudo-database tables and records yields sufficient speed and scalability, while allowing for a short development time. A discussion of storage scalability given in chapter 5 illustrates some of the limitations encountered by this approach.

The database organization has been designed around the particular topic of journal article metadata, but the usage of XML represents a flexible approach, since specialized XML document types can be designed for any type of article.

Articles are stored together in issue files; the article record contains the basic bibliographic information: title, authors, publication facts, identifiers. Citation information is stored as separate records in citation files. Volumes are represented by table of content files, which store basic information about the issues, such as dates and page ranges.

As an example, the table of contents for the 17th volume of ACM Transactions on Database systems is stored in the file `$DBROOT/ISSN/03/62/51/95/17/toc.xml` and has the following content:

```
<?xml version="1.0"?>
<Serial title="ACM Transactions on Database Systems" issn="0362-5915">
<Vol number="17">
   <Issue number="1" year="1992" month="March"
          firstpage="1" lastpage="199"/>
   <Issue number="2" year="1992" month="June"
          firstpage="201" lastpage="383"/>
   <Issue number="3" year="1992" month="September"
          firstpage="385" lastpage="560"/>
   <Issue number="4" year="1992" month="December"
          firstpage="563" lastpage="745"/>
</Vol>
</Serial>
```

The metadata for the articles in the 3rd issue of this volume is stored in the file `$DBROOT/ISSN/ISSN/03/62/51/95/17/3/PAGE.xml`. An excerpt of this file is presented below:

```
<?xml version="1.0"?>

<Serial title="ACM Transactions on Database Systems" issn="0362-5915">
<Vol number="17">
<Issue number="3" year="1992" month="September">
   <!-- ... (other articles) --->
   <Article>
      <From>bibtex</From>
      <ArticleTitle>
         Scheduling Real-Time Transactions: {A} Performance Evaluation
      </ArticleTitle>
      <AuthorList>
         <Author>Robert K. Abbott</Author>
         <Author>H{\'e}ctor Garc{\'\i}a-Molina</Author>
      </AuthorList>
      <Page begin="513" end="560"/>
   </Article>
</Issue>
</Vol>
</Serial>
```

Journal descriptions are stored in journal files, one for each journal. These files contain bibliographic information about the journal – names, abbreviations, publisher information, identifiers and publication pattern.

Services information is stored in files containing the following information: general facts about the service (descriptions, geographical location, access terms, etc.), agent descriptions, and holdings information. Appendix C offers two examples of service specifications, as they are submitted to the framework: the SFU Library Service (section C.1) and the PubMed service of the U.S. National Library of Medicine (section C.2).

The current organization of the database represents an initial prototyping effort; future iterations of the design are planned, that will improve on its efficiency, scalability and performance. The design proves that an XML-based approach offers a high degree of flexibility, in that serials with various enumeration hierarchies and organizations can be accommodated with a minimal effort. In contrast, a relational database design would need to consider a very general schema that can support any possible variation of bibliographic record, in addition to coping with inefficiency issues stemming from the storage of text in database records.

Chapter 5

Evaluation

This chapter evaluates the design of the BIBP framework, as well as some aspects of the prototype implementation of a BIBP server. The evaluation concentrates first on how the BIBP framework resolves the reference linking problem and how it compares to other citation linking systems. The second part of the chapter evaluates some of the systemic properties of the BIBP architecture, namely scalability and usability, while the third part discusses several outstanding technical issues.

5.1 Resolution of the Reference Linking Problem

In evaluating the BIBP framework as a reference linking solution, one inevitably has to consider how well the system resolves the reference linking problem - that is, how well it helps its users create and resolve citation links. Note that this is an enlargement of the reference linking problem as defined in [19], since the creation of links is also considered; moreover, in the BIBP system the resolution of a citation link does not only help one get to the cited item, but also to services relevant to the item in cause.

The reference linking problem can be seen from many different angles; some of these perspectives and the way the BIBP framework reflects them are presented below.

5.1.1 Link Construction

The BIBP model for the creation of citation links is extremely simple; finding the unique identifier for the serial is the only task where the author might have to consult an external reference database. In most of the cases, document authors already have the information needed to construct a USIN available in the paper they cite (this is especially true in the case of printed journals which bear the ISSN numbers); moreover, it is envisioned that in the future, serials will be assigned names that are easy to remember by members of a particular scientific community. Once a USIN is found, the creation of a hypertext link resolvable via the BIBP protocol is trivial, as demonstrated throughout the previous chapters.

In contrast, constructing a SICI is a far more complex endeavour, as seen in section 2.2.3. Obtaining a DOI or a PII may involve a reference database lookup if the DOI or PII are not published within the article content, or it may not be possible at all if such an identifier has not been assigned to the article.

5.1.2 Coverage

Since the BIBP system is based on USINs, its coverage is theoretically that of the USIN identifier system, which is the realm of serially published items. The effective coverage however depends on the availability of bibliographic metadata in the system.

The BIBP system is an open system and therefore neither its links nor its metadata are generated by a single authority or within a single article collection; instead, contributions are accepted from any interested party, regardless of its field of activity or institutional affiliation. This is in marked contrast with closed linking systems that concentrate on a discipline (e.g. PubMed), set of disciplines (e.g. Web of Science), a collection (e.g. Open-Journal), or with systems developed within boundaries defined by institutions or consortia (e.g. DOI). Furthermore, with the use of agents, the BIBP framework is able to expand its linking coverage to include other systems, as demonstrated by the prototype system which provides access to PubMed via BIBP links.

USIN identifiers can be assigned to documents regardless of their format of publication or general availability. Thus, the BIBP framework can provide links to both paper and electronic documents. Since the system provides links to services, paper documents are linked via library catalogues, document delivery or other services. Estimations have been cited [23] that allow 20 years until 90% of the references in a journal article will be to electronic articles only; therefore, links to holdings are arguably an important service for scholars.

5.1.3 Available Information

The information made available at the end of the resolution process is another aspect where the BIBP system improves on the existing practice. In addition to basic bibliographic information, the framework is able to link to a wide range of services, not limited to those traditionally offered by publishers, such as abstracting and reviews. As new services for serial items are established, their descriptions may be contributed to the framework, in order to enlarge the selection already offered to the users.

Links to services are built from link specifications which are similar to those in the S-Link-S and PubMed systems; this has the advantage that, unlike the Web of Science system, the BIBP system does not need to store links for each of the items covered by a service. The drawback is that building link specifications may not be possible when the items are not organized at the service site in a pattern that can be captured by a template.

Similar to the SFX system, the resolution process in the BIBP framework takes into account the environment in which links are consumed. As a consequence, the information presented to the user may be enhanced by a local service by incorporating data which is relevant only to users located within institutional boundaries (e.g. links to services covered by a site license).

5.1.4 Participation and Access

It can be argued that the BIBP framework contributes to more open access to bibliographic information by potentially concentrating all known information about an item under a single link and customizing it to the environment where the links are consumed. Open participation to the framework allows all the services relevant to an item to submit their descriptions and link specification. Users are free to choose the service they need based on the value it offers; thus, the system promotes competition among bibliographic services, hopefully leading to a better overall experience for the user community. Of the current linking systems reviewed, only S-Link-S and SFX allow such a feature, with the mention that S-Link-S constitutes the basis of a commercial system, while the SFX system starts with bibliographic metadata instead of citation identifiers [25].

Another aspect that contributes to the open spirit of the framework is its low cost of maintenance. The BIBP model is based on voluntary participation of the interested parties and on division of tasks among institutions willing to deploy BIBP servers. Moreover, it will be seen in section 5.2.1 that a BIBP server can be deployed on low-cost hardware.

An important trait of the BIBP framework is that it does not provide a unilateral service: the ability to offer a complete citation linking solution not only depends on the implementors and the maintainers of the framework, but also on the collaboration of its users. Without article metadata contributions from the users, the system is limited to the bibliographic information acquired by its maintainers and by the system itself via agents. This is a matter of social engineering, but there are arguments [15] that the cost of entering bibliographic and citation data at the source can be made very low. With proper support from the institutions that originate the items, and with adequate bibliographic software, authors can submit metadata for the papers they write with a minimum of effort. Publishers, libraries and other organizations involved in the storage and dissemination of published literature need to be encouraged and helped to submit their article information, in order for the BIBP framework to acquire metadata about literature published prior to its deployment.

5.1.5 Link Resolution

The BIBP protocol allows for a staged resolution process that gives the local user environment the first opportunity to resolve a link. This approach gives the user enhanced access to information, as mentioned in section 5.1.3. The multiple resolution (Harvard) problem is resolved by not linking directly to copies. Once a USIN is resolved and the metapage is presented, choosing a link to a copy of the item or to a service is left to the users. This is not considered a drawback, since the information presented to the user is always relevant to the item and the users have the opportunity to select whatever they want, as opposed to whatever the system considers "appropriate".

5.1.6 Integration With Other Services

The BIBP framework can be integrated with other linking systems due to its openness and generality of concept. First, any bibliographic service can be linked by submitting a service specification to the framework, or via agents. This allows services that were only accessed via a proprietary interface (typically a Web page with a search form) to be linked directly from HTML documents; the linking of PubMed proves the feasibility of such a task. Second, linking via USINs is available to any service, provided it can map the components of a USIN to the appropriate article record in its own database. An institution that provides a Level 1 BIBP implementation would write a simple BIBP server to access its own bibliographic data store and respond with an HTML page, possibly with the content of the document, but preferably with a metapage; the implementation of a Level 2 BIBP server would make the service available to the BIBP network, as a local server, or even as a master server, depending on institutional commitment.

Thus, most of the citation linking services mentioned so far are amenable to integration with BIBP. The OpenJournal project could apply post-hoc BIBP links to electronic journal pages, thus becoming an open system and solving the problem of unavailable information. Linking via DOIs could be accomplished if the BIBP system received a DOI prefix - USIN-based DOIs would thus be possible, although the mnemonic syntax would be compromised; a second alternative could be introduced by the contribution of article metadata that contains DOI identifiers and subscription of the DOI resolution system as a service, thus creating a bridge to the DOI framework. Similarly, the SFX system can be directly linked via a USIN to SFX-URL translation mechanism, or as a service link provided by a local BIBP server.

5.2 System Properties

5.2.1 Scalability

Scalability is a built-in feature of the BIBP system, both in terms of bandwidth and processing loads put on its servers, as well as the amount of storage required to accommodate current and future literature metadata. The BIBP framework was designed so that as the number of requests increase, a satisfactory level of performance can be maintained by the addition of new master and local servers to the network. Master servers partition the resolution space by resolving requests for a relatively small number of serials; in addition, they resolve the storage scalability problem, since they store the bulk of the metadata for the journals they serve, while the global and local servers need store only mappings from serial identifiers to master server references.

The BIBP protocol requires the building of scalability features directly into the client software: clients must direct their queries to the local servers first, thus taking the load off the global server. If local servers become overwhelmed, the load can be further spread by the deployment of additional BIBP servers on the local subdomains (e.g. if the host `bibhost.sfu.ca` does not offer enough performance, separate servers can be introduced for `sfu.ca` subnetworks, at `bibhost.cs.sfu.ca`, `bibhost.bus.sfu.ca` and so on).

Since the BIBP protocol is based on HTTP requests and responses, the issues related to Web server performance apply to the BIBP framework, in particular those scenarios involving active page generation in three-tier systems, as is the case of the BIBP server, where the content is generated by Java servlets based on data extracted from a database. As such, the problems are typical to this technology: aside from the implementation of the servlet engine and that of the servlet itself, the performance of a single BIBP server largely depends upon the operating system chosen (I/O and multithreading performance in particular), Web server technology and ultimately the hardware used. It is envisioned that the number of requests generated to a BIBP server (which may consist of a single machine, or a server farm) would be much smaller than those experienced by popular news sites for instance, but there is nothing inherent to the BIBP technology that prevents a BIBP site from achieving similar performance (i.e. thousands of requests per second).

The BIBP network is envisioned to grow from an initial global server with the addition of master and local servers. The initial global server should be able to serve enough article metadata to be useful at least for one branch of science; the question is then how much data is enough? If the initial global BIBP server stored no article metadata and only agents for the major bibliographic sites, such as PubMed or the IETF RFC site, it would still be useful since it links those sites via USINs. Therefore, "enough" metadata means very little: several service descriptions and Java classes or scripts that implement the BIBP Agent API. However, the global server should be able to store more than agents, in that it can be initially populated with bibliographic metadata readily available on the Internet in the form of BIBTEX files or other formats; contributions from authors and publishers and the deployment of master servers would enlarge this base to eventually cover the most significant serially published literature in the world.

A goal of this project was to develop an architecture that allows the deployment of a local BIBP server on a low cost machine, for instance a PC machine equipped with modest storage facilities, e.g. a 20GB harddisk. Calculations made using the prototype implementation for the BIBP database revealed that a long running journal such as the Journal of the ACM, for which the database holds article descriptions since 1954, requires 615 files (this includes

XML files and directory files); this large figure is due to the inefficient space utilization of the experimental database, which uses one XML file for each issue; if instead all the issues in a volume are stored into one XML file, the number of XML files decreases by a factor of 4. The amount of storage space taken by a single article is 1KB, while the average size of a file is 1.7KB – another deficiency of the current implementation, since it induces heavy disk fragmentation. The numbers above yield a total of $20 * 10^9/(615 * 1700) = 19,129$ journals that could be stored on a 20GB hard disk; these journals would be covered for a long time in the future, given that half of the publications in the ISSN Register have been registered in the last 10 years. Another constraint imposed on the problem is that the number of i-nodes available in the file system is a limited resource: a Linux filesystem such as extfs2 [20] would make around 20 million i-nodes available on a 20GB hard disk, and therefore it should accomodate files for $20 * 10^6/615 = 32,520$ journals, which insures that the 19,129 journals calculated based on the space constraints only indeed fit on the disk.

The above results indicate that even if a local BIBP server was hosted on a low-cost machine (or two machines, for fault tolerance) and with the current implementation of the database (which will be re-written shortly), the server could still store information about a significant number of journals, much more than the 4,000 journals indexed in the PubMed system. Thus, it can be argued that a useful BIBP system does indeed scale and does not require a costly investment from its implementors.

If past and current computer trends are to be considered, then according to Moore's Law[1], the storage capabilities per dollar of the BIBP network will double every year and a half. It is unlikely however that the production of scholarly literature will grow at the same accelerated pace (a more modest growth of 2.5% per year for STM literature is indicated in [39, 42]). On this basis, one could assert that the architecture of the BIBP framework offers a growth model that is sustainable at any point in time. The capability growth afforded by Moore's Law may be used to support an expansion in coverage due to the application of USIN identifiers to finer-grained items, such as minutes of governmental and institutional committees.

[1]Moore's Law is based on an observation made in 1965 by Gordon Moore, co-founder of Intel, that the number of transistors per square inch on integrated circuits doubles every year. The current definition of the law is that data density doubles approximately every 18 months – an opinion shared by most experts, including Moore himself (source: Webopedia, currently at URL: `http://www.webopedia.com/`).

5.2.2 Usability

One of the major technical hurdles in the implementation of clients for the BIBP framework is the lack of support for URL handling in the current browsers. If the major browsers will not provide hooks that allow for plugin-type software to handle URIs written for unsupported protocols, or if they will not directly support BIBP, then Web linking with USINs is restricted to the scripting solution described throughout this document. Scripting provides an acceptable solution, but it also introduces a usability problem for authors, since the script has to be either included or referenced from HTML files in order for the BIBP links to work.

However, from the usability point of view, scripting is arguably a better solution than browser plugins or other protocol handling mechanisms since it requires no effort from the user side. On-line resolution of DOIs for instance is based on the Handle protocol [1]; implementation of DOI resolvers for Web browsers poses practical problems related to platform availability (the DOI plugin is only available for Windows and Macintosh platform) and the ability to traverse firewalls (the handle protocol is not recognized out of the box by most of the current firewalls).

Another area of usability where technical difficulties may exist is that of creating BIBP links. While the USIN syntax is designed to be as mnemonic as possible, authors may find themselves in difficulty with respect to obtaining proper serial identifiers, assembling syntactically correct USINs and forming URIs (which may require escaping for those characters considered special under the URI syntax [11]). To address these problems, the BIBP framework will provide tools for creating links from metadata, including serials dictionaries and USIN and URI generators.

The usability of the interface offered to the users – namely, the metapages offered in response to resolution requests – is not a direct responsibility of the BIBP system, since service providers will compete in this arena to offer increasingly usable solutions.

5.3 Technical Issues

5.3.1 Arbitrary Enumeration Hierarchies

The current implementation uses separate XML document types and software for each of the serial types supported. This organization allows for optimized processing, but does not scale well since new document types have to be created every time a serial with a new enumeration hierarchy is covered. The decision to use separate XML document types was taken in order to experiment with various formats that were deemed useful, and which covered a large mass of literature. Once the experimentation phase is done, a more general XML schema will replace the current one and provide support for arbitrary enumeration hierarchies.

5.3.2 Serial Title Matching

One of the technical challenges in the area of contributions is establishing the correct identity for an article citation. Since serial titles support many variations, the submission may contain typos or misspellings. Algorithms that were developed to detect identical citations (based on word and phrase matching), or string distance algorithms can be used [12, 14], but since none of them offer 100% accuracy, the method needs to be backed by interaction with the contributors. The system may be able to detect a title in a citation as potentially belonging to several journals, but it is still the contributor who has to identify which one the citation is referring to. The `usin.org` website will offer support for finding the correct journal title by collecting, storing, indexing and publishing webpages for journal titles.

5.3.3 Internationalization

The bibliographic metadata is currently stored using the US-ASCII character set (the ANSI X3.4-1986 standard). The advantage of using this character set lies in its simplicity; moreover, it is supported by any browser. However, the BIBP framework should in the future provide support for international languages and character sets; this support has to be reflected in the BIBP protocol and the storage schemes for the database.

Chapter 6

Conclusion and Future Work

6.1 Summary

This thesis demonstrates a viable solution to the Web-based reference linking problem, by adhering to an approach that focuses solely on citation links to serially published items, without ever having the intention to encompass any other classes of items, or offering any other services than resolution to comprehensive bibliographic information.

The thesis brings a number of contributions to the field of reference linking on the Web. The system proposed herein introduces for the first time support for USIN identifiers, thus guaranteeing a precisely delimited scope and eliminating several problems encountered by other identifier systems; deployment of USINs on the Web revealed a number of practical issues and suggested improvements on the USIN syntax.

The BIBP framework effectively constitutes a working URN resolution system, as USINs satisfy all the URN requirements. Its design demonstrates that viable URN systems can be built when they are restricted to a judiciously chosen domain. A unique feature of the system is the manner in which localized resolution is dealt with, by offering a very flexible solution that capitalizes on current Internet technologies, namely the Domain Name System and the scripting capabilities of today's Internet browsers.

Introduction of the BIBP protocol as a support technology for USINs and the BIBP reference linking framework adds to the collective Web and Internet experience by establishing a new way to handle protocols not supported by current browsers (via the JavaScript user agent) and demonstrates a simple, yet powerful way to design distributed systems with only several HTTP primitives.

The approach taken towards a structured description of Web bibliographic services adds a new facet to metadata development, in that it does not confine itself to describing links to services, but instead describes services as whole systems that have outstanding properties such as access method, geographical locations and differentiated subservices.

Finally, the system proposed by this thesis is an open system - anyone is free to participate and experiment with it. This characteristic places it in a unique position in a field where strong institutional influences and closed approaches based on particular science domains are the norm.

6.2 Future Research

The BIBP framework is currently well defined at the Level 1 [17]. The Level 2 of the BIBP needs to be developed into a complete specification that will allow an active promotion of the BIBP network; reference implementations at this level will provide software and guidance to other parties wishing to deploy BIBP servers. The Level 3 of the protocol is envisioned to provide a framework for implementation of a Universal Citation Database, as described in [15].

The system will keep in step with future USIN research and accommodate the new syntax specifications that may appear for a wider variety of serial and serial item types. It is expected that the USIN specification will establish and consolidate a scheme for the unique and persistent identification of series in the general case (beyond the current ISSN, ISBN and RDNS domains), and that syntax specifications for the expression of numbering hierarchies of any depth will be recommended.

On the metadata front, further research is needed to develop more generalized XML representation of serial items. The current implementation of the XML-based BIBP database needs to be optimized for better scalability. Another important aspect of metadata research is the development of schemes for the representation of serial publication patterns, which will allow BIBP processing entities to perform semantic validation of submitted records, and information unification (construction of complete data structures by combining subscriptions of partial or partially incorrect information).

Research in the area of user interfaces may aid in the adoption of BIBP technology. In particular, BIBP clients may be implemented as bibliographic plug-ins for the most popular document preparation packages, to aid authors in the construction of USINs and reference lists.

Appendix A

Example Usage of BIBP Links In HTML Documents

This section illustrates the steps that need to be taken when authoring an on-line HTML document containing BIBP links.

A.1 Specification of Resolution Parameters

In order for the BIBP links in an HTML document to be resolvable, the authors need to specify a resolver script and an optional parameter denoting the preferred server. These specifications have to occur in the HEAD element of the HTML document so that browsers can load the resolution code before any of the BIBP links (which occur in the document body). An example is provided below, without including the pasted resolver script, which is presented in section A.3:

```
<HTML>
<HEAD>
   <TITLE>Paper Title</TITLE>

   <!-- the following declaration specifies the preferred host for browsers
        that provide BIBP support -->
   <LINK REL="citehost" href="http://www.preferredhost.com/">

   <!-- the folllowing declaration specifies the preferred host for
```

```
        current browsers (which do not support BIBP) -->
   <SCRIPT TYPE="text/javascript>
      <!-- hide from old browsers
      bibp_citehost="http://www.preferredhost.com/" ;
      -->
   <SCRIPT>

   <SCRIPT TYPE="text/javascript">
   <!-- here goes the script code -->
   </SCRIPT>
```

A.2 HTML Content with BIBP Links

BIBP links may be specified in the list of references for an on-line document, as follows:

```
<HTML>
<HEAD>
   <TITLE>Paper Title</TITLE>
   <!-- here goes the specification of resolution parameters and the
        script (or a link to it) -->
</HEAD>
<BODY>

   <!-- ... here goes the content of the paper -->

   <!-- declaration of references: -->
   [Abbot-Molina92]
   R. Abbott and H. Garcia-Molina. Scheduling Real-Time
   Transactions: A Performance Evaluation. ACM Transactions on Database
   Systems, Vol. 17, No. 3, pp. 513-560, September1992.
   <A HREF="bibp:ISSN/0362-5195:17@513"
      TITLE="links to bibliographic information"
      REL="meta"
      REV="cite">here goes the anchor name</A>

   <!-- ... other references -->
</BODY>
</HTML>
```

In accordance with the HTML specification [5], the REL attribute for the BIBP link specifies that the document at the end of the link represents metadata for the current document, while the `REV` attribute specifies that the current document provides a citation of the referenced entity.

A.3 A JavaScript Resolver

The following script is taken from the BIBP Level 1 specification document [17]. The script code is unaltered, but formatting operations have been applied to the original compact form, for better readability.

```
// (c) Robert D. Cameron and Serban Tatu, June 2000, GNU Public License.
var BibP_BaseURL ;
function BibP_SetBaseURL (server) {
   BibP_BaseURL = "http://" + server + "/bibp1.0/resolve?"
                  + (BibP_citehost
                     ? "citehost=" + BibP_citehost + "&usin="
                     : "usin=") ;
}
BibP_SetBaseURL(BibP_citehost ? BibP_citehost : "usin.org") ;
var BibP_Icon = new Image ();   // To test for local bibhost icon.
function BibP_onIcon () {
   if (BibP_Icon.height!=0)
      BibP_SetBaseURL("bibhost") ;
}
BibP_Icon.onload = BibP_onIcon ;
BibP_Icon.src = "http://bibhost/bibp1.0/bibpicon.jpg";
function BibP_ShowResolved (theURL) {
   if (theURL.indexOf(BibP_BaseURL) == 0) {      //  A resolved bibp: URL?
      window.status = "bibp:" + theURL.slice(BibP_BaseURL.length) ;
      return true ;
   } else
      return false ;
}
function BibP_TryResolve (elem, theURL) {
   var bibpSpot = theURL.indexOf("bibp:") ;
   if (bibpSpot != -1) {
      window.status = theURL.slice(bibpSpot) ;
      elem.href = BibP_BaseURL + theURL.slice(bibpSpot+5) ;
   }
   return true ;
}
function BibP_NN_onMouseOver (event) {
   return BibP_ShowResolved(event.target.href) ||
          BibP_TryResolve(event.target, event.target.href) ;
}
function BibP_IE_onMouseOver () {
   var URL = new String(event.srcElement) ;
   return BibP_ShowResolved(URL) || BibP_TryResolve(event.srcElement, URL) ;
}
if (navigator.appName.indexOf('Netscape') != -1) {
   if (parseInt(navigator.appVersion) >= 4) {
      document.captureEvents(Event.MOUSEOVER | Event.MOUSEOUT) ;
   }
   document.onMouseOver = BibP_NN_onMouseOver ;
} else if (navigator.appName.indexOf('Microsoft') != -1) {
   document.onmouseover = BibP_IE_onMouseOver ;
}
function BibP_onMouseOut () {
   window.status='';
   return true ;
}
document.onmouseout = BibP_onMouseOut;
// END SCRIPT
```

Appendix B

Example Article Metadata

Illustrated below is the use of XML to format a submission of bibliographic information for the following article:

> R. Abbott and H. Garcia-Molina. *Scheduling Real-Time Transactions: A Performance Evaluation*. ACM Transactions on Database Systems, Vol. 17, No. 3, pp. 513–560, September 1992.

The submission uses the BIBP XML format for articles. The example is deliberately incomplete, given its illustrative purpose; only two cited articles are presented here.

```
<?xml version="1.0"?>
  <!DOCTYPE ArticleList SYSTEM "http://usin.org/dtd/article_submit.dtd">

  <ArticleList>
     <Submitter>
        <Name>John Smith</Name>
        <Email>jsmith@university.edu</Email>
     </Submitter>
     <Article>
        <Title>
           Scheduling Real-Time Transactions: A Performance Evaluation
        </Title>
        <AuthorList>
           <Author>R. Abbott</Author>
           <Author>H. Garcia-Molina</Author>
        </AuthorList>
```

```
    <Journal issn="0362-5915">
        ACM Transactions on Database Systems
    </Journal>
    <Volume>17</Volume>
    <Issue>3</Issue>
    <Year>1992</Year>
    <Page begin="513" end="560"/>
    <IdentifierList>
       <Identifier type="USIN">ISSN/0362-5915:17@513</Identifier>
       <Identifier type="DOI">some_doi</Identifier>
       <!-- etc. -->
    </IdentifierList>

    <!-- this is optional, although highly preferred! -->
    <CitationList>
       <!-- provide a usin for the cited article and make another
            article record for it somewhere in this file -->
       <Cite usin="ISSN/0163-5808:17(1)@71"/>
       <Cite usin="ISSN/0001-0782:19(11)@624"/>
       <Cite usin="ISBN/0-934613-75-3@1"/>
       <!-- etc. -->
    </CitationList>
 </Article>

 <!-- an article cited by the article above; it has no citation list -->
 <Article>
    <Title>
      The Notions of Consistency and Predicate Locks
      in a Database System
    </Title>
    <AuthorList>
       <Author>Kapali P. Eswaran</Author>
       <Author>Jim Gray</Author>
       <Author>Raymond A. Lorie</Author>
       <Author>Irving L. Traiger</Author>
    </AuthorList>
    <Journal issn="0001-0782" abbr="CACM">
       Communications of the ACM
    </Journal>
    <Volume>19</Volume>
    <Issue>11</Issue>
    <Year>1976</Year>
```

```
      <Page begin="624" end="633"/>
   </Article>

   <!-- a conference proceedings citation -->
   <InProceedings>
      <Title>
         Scheduling Real-time Transactions: a Performance Evaluation
      </Title>
      <AuthorList>
         <Author>Robert K. Abbott</Author>
         <Author>Hector Garcia-Molina</Author>
      </AuthorList>
      <Book isbn="0-934613-75-3">VLDB</Book>
      <Year>1988</Year>
      <Page begin="1" end="12"/>
   </InProceedings>

   <!-- etc. -->

</ArticleList>
```

Appendix C

Example Service Metadata

This section illustrates the description of bibliographic services using XML specifications. Two examples are provided, one for the description of a library service for the Simon Fraser University Library; the other one illustrates the description of a service capable of providing basic bibliographic information via agents (section 4.3.1).

C.1 The SFU Library Service

The SFU Library offers (among others) an on-line based catalog service, and a document requesting service called GODOT (Generalized Online Documents, Ordering and Texts) which provides a unified interface for the retrieval of fulltext from various sources with which the library has special agreements.

The example shows the specification for Web links to the catalog and the GODOT service. In addition, a link specification is offered for the ACM Digital Library, whose texts are available freely to the SFU community via a site license.

```
<?xml version="1.0"?>
<!DOCTYPE ServiceList SYSTEM  "service_submit.dtd">

<ServiceList provider="SFU Library" uid="lib.sfu.ca" type="&local;">

   <!-- the description for the SFU library service (catalog and GODOT) -->
   <Service id="lib">
      <Info>
   <URI>http://www.lib.sfu.ca/</URI>
   <Location>Vancouver, British Columbia</Location>
   <Description>The mission of  the SFU Library is to support
   the teaching, learning and research goals of the Simon Fraser
   University community</Description>
   <Access>Restricted to the members of the SFU community</Access>
</Info>

<LinkTemplateList>
   <!-- a link template for the SFU catalog -->
   <URI returns="&journal.holdings;">
      <CategoryList>
         <Category id="&fulltext;">Journal Holdings</Category>
      </CategoryList>
      <HTTP_GET>
         http://troy.lib.sfu.ca/search/i?SEARCH=&journal.issn;
      </HTTP_GET>
   </URI>

   <!-- a link template for the GODOT system -->
   <URI returns="&journal.holdings;">
      <CategoryList>
        <Category id="&fulltext;">
           Holdings/Request Item via GODOT
        </Category>
      </CategoryList>
      <HTTP_POST>
         <Action>http://delos.lib.sfu.ca/perl/hold_tab.cgi</Action>
         <Input name="hold_tab_dbase_local">ecdb</Input>
         <Input name="hold_tab_dbase_type">erl</Input>
         <Input name="hold_tab_userno">19V4vpv</Input>
         <Input name="hold_tab_ti">&article.title;</Input>
         <Input name="hold_tab_so">&journal.title;</Input>
         <Input name="hold_tab_ca">&article.authors.keywords;</Input>
         <Input name="hold_tab_issn">&journal.issn;</Input>
         <Input name="hold_tab_ft">0</Input>
      </HTTP_POST>
   </URI>
</LinkTemplateList>

<!--
Note: no holdings are specified here: in this case, the BIBP server
```

```
      will offer a link to this server for every article
      -->
   </Service>

   <!-- the description for the ACM service, provided by the SFU library
    via a site license -->
   <Service id="acm">
      <Info>
         <URI>http://www.lib.sfu.ca/cgi-bin/trust1.pl?acm_journals</URI>
         <Access>Access Type: Use of ACM (Association for Computing Machinery)
         Journals is restricted to members of the SFU community. You may access
         ACM (Association for Computing Machinery) Journals freely from on
         campus, or by dialing into the campus network. You may also gain
         access using your SFU computing services CCN (Unix) account id and
         password--that is, the same account that you use to access email and
         other computing services.</Access>
      </Info>

      <LinkTemplateList>
         <URI returns="&article.metadata;">
            <CategoryList>
               <Category id="&fulltext;">
                  ACM Digital Library: Full Text Access
               </Category>
               <Category id="&abstract;">
                  ACM Digital Library: Abstract
               </Category>
               <Category id="&review;">
                  ACM Digital Library: Reviews
               </Category>
               <Category id="&classify;">
                  ACM Digital Library: Index Terms
               </Category>
            </CategoryList>
            <HTTP_POST>
               <Action>http://www.acm.org/ows-bin/dl/owa/dl_srch.new</Action>
               <Input name="fields_expr">&article.title.keywords;</Input>
               <Input name="fields_expr_type">ALL</Input>
               <Input name="f_title">y</Input>
               <Input name="author_expr">&article.authors.keywords;</Input>
               <Input name="author_expr_type">ALL</Input>
               <Input name="isbn_issn">&journal.issn;</Input>
               <Input name="since_year">&issue.year;</Input>
            </HTTP_POST>
         </URI>
      </LinkTemplateList>

      <HoldingsList>
         <Holdings>
            <USIN>ISSN/0164-0925:7--ff</USIN> <!-- toplas -->
            <Media>&online;</Media>
            <Type>&fulltext;</Type>
            <Type>&metadata;</Type>
         </Holdings>
         <Holdings>
            <USIN>ISSN/00045411:32--ff</USIN> <!-- jacm -->
            <Media>&online;</Media>
            <Type>&fulltext;</Type>
            <Type>&metadata;</Type>
         </Holdings>
         <Holdings>
            <USIN>ISSN/03625915:1--ff</USIN> <!-- tods -->
            <Media>&online;</Media>
            <Type>&fulltext;</Type>
            <Type>&metadata;</Type>
         </Holdings>
      </HoldingsList>

   </Service>
</ServiceList>
```

C.2 The PubMed On-line Bibliographic Service

A sample service specification for the PubMed service is given below. This specification may be submitted to the BIBP framework, along with the agent implementation.

```
<?xml version="1.0"?>
<!DOCTYPE ServiceList SYSTEM "service_submit.dtd">

<ServiceList provider="National Library of Medicine: PubMed"
             uid="ncbi.nlm.nih.gov"
             type="&online;">
   <Service id="pubmed">
```

```
<Info>
   <URI>
      http://www.ncbi.nlm.nih.gov:80/entrez/query/static/overview.html
   </URI>
   <Location>
      U.S. National Library of Medicine,
      8600 Rockville Pike, Bethesda, MD 20894
   </Location>
   <Description>
      PubMed is the National Library of Medicine's search service
      that provides access to over 10 million citations
      in MEDLINE, PreMEDLINE, and other related databases,
      with links to participating online journals.
   </Description>
   <Access>
      unrestricted
   </Access>
</Info>

<!-- Agents are programs called by the BIBP server to perform a
query for an article; they are customized to the specifics of the
site and implement a BIBP API that allows the server to pass them
USIN's and to interpret the results returned; The agents can be
written in Java (fastest and cleanest way, if not the easiest),
or any other type of language. -->
<AgentList>
   <Agent returns="&article.metadata;">
      <Java classlib="pubmed.jar">
         org.usin.pubmed.PubmedAgentFactory</Java>
      <!-- alternatively, the agent could be a Perl script:
      <System>pubmed.pl</System>
      -->
   </Agent>
</AgentList>

<!-- in addition to supplying an agent for a list of journals, the
submitter may register value-added services to be presented to
the user. Thus, the PubMed specification may define an agent which
knows how to extract basic bibliographic information, but it may
also define a service by providing a link template to the PubMed
website -->
<LinkTemplateList>
```

```
        <URI returns="&article.metadata;">
           <CategoryList>
              <Category id="&fulltext;">
                 Fulltext Access through LinkOut
              </Category>
              <Category id="&abstract;">
                 Article Abstract
              </Category>
              <Category id="&deliver;">
                 Get Article through Loansome Doc
              </Category>
           </CategoryList>
           <HTTP_GET>
              <!-- note: the backslash (\) is used here to split the URL
              over multiple rows; its use is illegal in a real service
              specification -->
              http://www.ncbi.nlm.nih.gov/htbin-post/Entrez/query?db=m\
              &amp;form=4&amp;term=&journal.issn;[JOUR]+AND+\
              &volume.number;[VOL]+AND+&article.page;[PAGE]&amp;dispmax=1
           </HTTP_GET>
        </URI>
     </LinkTemplateList>

     <!--a long list of journal ISSN's that are covered by this service-->
     <HoldingsList>
        <Holdings>
           <!--INTERNATIONAL JOURNAL OF DEVELOPMENTAL NEUROSCIENCE-->
           <USIN>ISSN/0736-5748</USIN>
        </Holdings>
        <Holdings>
           <!-- JOURNAL OF THE AMERICAN BOARD OF FAMILY PRACTICE -->
           <USIN>ISSN/0893-8652</USIN>
        </Holdings>
        <Holdings>
           <!-- JOURNAL OF BIOLUMINESCENCE AND CHEMILUMINESCENCE -->
           <USIN>ISSN/0884-3996</USIN>
        </Holdings>
        <!-- etc. -->
      </HoldingsList>

   </Service>
</ServiceList>
```

Bibliography

[1] Handle System Introduction. Online document, currently at URL: http://www.handle.net/introduction.html.

[2] IFLA Universal Bibliographic Control and International MARC Core Programme, U-NIMARC: an introduction. Online document, currently at URL: http://www.ifla.org/VI/3/p1996-1/unimarc.htm.

[3] ISO 8824 Information Processing Systems - Open Systems Interconnection - Specifications for Abstract Syntax Notation One (ASN.1), 1987.

[4] ANSI Z39.50: Information Retrieval Service and Protocol, 1995.

[5] HTML 4.01 Specification. W3C Recommendation, December 24 1999. `USIN: RDNS(W3.ORG)/TR:REC-html401-19991224`.

[6] Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation, February 22 1999. `USIN: RDNS(W3.ORG)/TR:REC-rdf-syntax-19990222`.

[7] William Arms, Leslie Daigle, Ron Daniel, Dan LaLiberte, Michael Mealling, Keith Moore, and Stuart Weibel. Uniform Resource Names - A Progress Report. *D-Lib Magazine*, 2(2), February 1996. `USIN: ISSN/1082-9873:2(2)$02arms`.

[8] Helen Atkins. The ISI Web of Science - Links and Electronic Journals. *D-Lib Magazine*, 5(9), September 1999. `USIN: ISSN/1082-9873:5(9)$09atkins`.

[9] Helen Atkins, Catherine Lyons, Howard Ratner, Carol Risher, Chris Shillum, David Sidman, and Andrew Stevens. Reference Linking with DOIs: A Case Study. *D-Lib Magazine*, 6(2), February 2000. `USIN: ISSN/1082-9873:6(2)$02risher`.

[10] David Bearman, Eric Miller, Godfrey Rust, Jennifer Trant, and Stuart Weibel. A Common Metadata Model to Support Interoperable Metadata. *D-Lib Magazine*, 5(1), January 1999. `USIN: ISSN/1082-9873:5(1)$01bearman`.

[11] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax and Semantics. Request for Comments: 2396, Category: Standards Track, August 1998.
`USIN: RDNS(IETF.ORG)/RFC:2396.`

[12] Kurt D. Bollacker, Steve Lawrence, and C. Lee Giles. CiteSeer: An Autonomous Web Agent for Automatic Retrieval and Identification of Interesting Publications. In *2nd International ACM Conference on Autonomous Agents*, pages 116–123, May 10–13 1998.
`USIN: ISBN/0-89791-983-1@116.`

[13] D. Box et al. SOAP: Simple Object Access Protocol, April 2000. Online document, currently at URL:
http://msdn.microsoft.com/xml/general/soapspec.asp.

[14] Steve Lawrence C. Lee Giles, Kurt D. Bollacker. CiteSeer: An Automatic Citation Indexing System. In *Digital Libraries '98: The Third ACM Conference on Digital Libraries*, pages 89–98, June 1998.
`USIN: ISBN/0-89791-965-3@89.`

[15] Robert D. Cameron. A Universal Citation Database as a Catalyst for Reform in Scholarly Communication. *First Monday*, 2(4), April 1997.
`USIN: ISSN/1396-0466:2(4)$cameron.`

[16] Robert D. Cameron. Towards Universal Serial Item Names. *Journal of Digital Information*, 1(3), October 1998.
`USIN: ISSN/1368-7506:1(3)$Cameron.`

[17] Robert D. Cameron and Serban Tatu. Bibliographic Protocol Level 1: Link Resolution and Metapage Retrieval. To appear as a CMPT Technical Report, Simon Fraser University.

[18] Priscilla Caplan. A model for reference linking. Report of the working group of the reference linking workshop, October 6 1999. Online document, currently at URL:
http://www.lib.uchicago.edu/Annex/pcaplan/reflink.html.

[19] Priscilla Caplan and William Y. Arms. Reference Linking for Journal Articles. *D-Lib Magazine*, 5(7/8), July 1999.
`USIN: ISSN/1082-9873:5(7/8)$07caplan.`

[20] Rémy Card, Theodore Ts'o, and Stephen Tweedie. Design and Implementation of the Second Extended Filesystem. In *Proceedings of the First Dutch International Symposium on Linux*, December 1994.

[21] James Duncan Davidson and Suzanne Ahmed. Java Servlet API Specification Version 2.1a. Online document, available from the Sun Microsystems Web site, November 1998.

[22] Lloyd A. Davidson and Kimberly Douglas. Digital Object Identifiers: Promise and Problems for Scholarly Publishing. *The Journal of Electronic Publishing*, 4(2), December 1998.
`USIN: ISSN/1080-2711:4(2)$davidson.`

[23] Herbert Van de Sompel and Patrick Hochstenbach. Reference Linking in a Hybrid Library Environment, Part 1: Frameworks for Linking. *D-Lib Magazine*, 5(4), apr 1999.
`USIN: ISSN/1082-9873:5(4)$04van_de_sompel-pt1.`

[24] Herbert Van de Sompel and Patrick Hochstenbach. Reference Linking in a Hybrid Library Environment, Part 2: SFX, a Generic Linking Solution. *D-Lib Magazine*, 5(4), apr 1999.
`USIN: ISSN/1082-9873:5(4)$04van_de_sompel-pt2.`

[25] Herbert Van de Sompel and Patrick Hochstenbach. Reference Linking in a Hybrid Library Environment, Part 3: Generalizing the SFX solution in the "SFX@Ghent & SFX@LANL" experiment. *D-Lib Magazine*, 5(10), oct 1999.
`USIN: ISSN/1082-9873:5(10)$10van_de_sompel.`

[26] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. Request for Comments 2616, Internet Engineering Task Force, June 1999.
`USIN: RDNS(IETF.ORG)/2616.`

[27] Brian Green and Mark Bide. Unique Identifiers: a brief introduction, 1998.
`USIN: ISBN/1-873671-18-0.`

[28] Eric S. Hellman. The S-Link-S^{TM} Framework for Reference Linking: Architecture and Implementation. Online document, currently at URL:
http://www.openly.com/SLinkS/ronneby.pdf.

[29] S. Hitchcock, L. Carr, S. Harris, J. M. N. Hey, and W. Hall. Citation linking: improving access to online journals. In *Proceedings of the 2nd ACM international conference on Digital libraries July 23 - 26, 1997, Philadelphia, PA USA*, July 1997.
`USIN: ISBN/0-89791-868-1@115.`

[30] Steve Hitchock et al. Linking electronic journals. Lessons Learned from the Open Journal project. *D-Lib Magazine*, 4(12), December 1998.
`USIN: ISSN/1082-9873:4(12)$12hitchcock.`

[31] Clifford Lynch, Cecilia Preston, and Ron Daniel. Using Existing Bibliographic Identifiers as Uniform Resource Names. Request For Comments 2288, Internet Engineering Task Force, February 1998.
`USIN: RDNS(IETF.ORG)/RFC:2288.`

[32] Eric Miller, Paul Miller, and Dan Brickley. Guidance on expressing the Dublin Core within the Resource Description Framework (RDF). Dublin Core Metadata Initiative, July 1 1999. Working Draft.

[33] R. Moats. URN Syntax. Request for Comments 2141, Internet Engineering Task Force, May 1997.
`USIN: RDNS(IETF.ORG)/RFC:2141.`

[34] Todd Mundle and Dave Binkley. GODOT: an Open, Highly Configurable Document Requesting System. A presentation at the 1999 Ontario Libraries Association Super Conference, January 22 1999.

[35] E. Nebel and L. Masinter. Form-based File Upload in HTML. Request for Comments 1867, Internet Engineering Task Force, November 1995.
`USIN: RDNS(IETF.ORG)/RFC:1867.`

[36] IFLA Study Group on the Functional Requirements for Bibliographic Records. Functional Requirements for Bibliographic Records. In UBCIM Publications - New Series, Vol 19, 1998. Final Report, K.G. Saur München.
`USIN: ISBN/3-598-11382-X.`

[37] National Information Standards Organization. *Serial Item and Contribution Identifier (SICI)*. NISO Press, August 14 1996.
`USIN: ISBN/1880124289.`

[38] F. Parmentier and A. Belaid. Logical Structure Recognition of Scientific Bibliographic References. In *Proceedings of the 4th International Conference on Document Analysis and Recognition (ICDAR '97)*, pages 1072–1076, August 1997.
`USIN: ISBN/0-8186-7898-4@1072.`

[39] Norman Paskin. Information Identifiers. *Learned Publishing*, 10(2):135–156, April 1997.
`USIN: ISSN/0953-1513:10@135.`

[40] Norman Paskin. DOI: Current Status and Outlook. *D-Lib Magazine*, 5(5), May 1999.
`USIN: ISSN/1082-9873:5(5)$05paskin.`

[41] Norman Paskin. E-Citations: actionable identifiers and scholarly referencing. DOI:10.1000/170, November 1999. Version 1.2.

[42] Norman Paskin. Toward Information Identifiers. *Proceedings of the IEEE*, 87(7):1208–1227, July 1999.
`USIN: ISSN/0018-9219:87(7)@1208.`

[43] Norman Paskin and Godfrey Rust. The Digital Object Identifier initiative: metadata implications. DOI Discussion paper Number 2, February 1999. Version 3.

[44] American Chemical Society, American Institute of Physics, American Physical Society, Elsevier Science, and IEEE. Publisher Item Identifier as a means of document identification, 1995. Online document, currently at URL:
http://www.elsevier.nl/inca/homepage/about/pii/.

[45] K. Sollins. Architectural Principles of Uniform Resource Name Resolution. Request for Comments 2276, Internet Engineering Task Force, January 1998.
`USIN: RDNS(IETF.ORG)/RFC:2276.`

[46] Karen Sollins and Larry Masinter. Functional Requirements for Uniform Resource Names. Request For Comments 1737, Internet Engineering Task Force, December 1994.
`USIN: RDNS(IETF.ORG)/RFC:1737.`

[47] S. Weibel, J. Kunze, C. Lagoze, and M. Wolf. Dublin Core Metadata for Resource Discovery. Request for Comments 2413, Internet Engineering Task Force, September 1998.
`USIN: RDNS(IETF.ORG)/2413.`

[48] James E. Whitehead, Jr. and Meredith Wiggins. WEBDAV: IETF Standard for Collaborative Authoring on the Web. *IEEE Internet Computing*, 2(5):34–40, September-October 1998.
`USIN: ISSN/1089-7801:2(5)@34.`