Citation Linking in Federated Digital Libraries

Eike Schallehn Martin Endig Kai-Uwe Sattler

Department of Computer Science, University of Magdeburg
P.O. Box 4120, D-39016 Magdeburg, Germany
{eike|endig|kus}@iti.cs.uni-magdeburg.de

Abstract

Today, bibliographical information is kept in a variety of data sources world wide, some of them publically available, and some of them also offering information about citations made in publications. But as most of those sources cover only certain areas of publications it is currently not possible to follow citation links when information about the referenced publication is not stored locally. This is known as the citation linking problem. In this paper we present an approach to integrating bibliographical information including citation information from various sources providing a platform for the solution of the citation linking problem. The approach is based on a loosely coupled federation using the FRAQL language and its conflict resolution mechanisms. Furthermore, we describe an adapter allowing a seamless integration of various sources in an heterogeneous and highly distributed environment.

1 Motivation

Over the last years the way information is accessed has changed significantly because of the rise of the Internet and a move toward electronic publishing. New techniques had to be developed to efficiently look up the information of interest for a given user among the huge amount of information available. While good progress has been made on this issue using new approaches for information retrieval there is still work to be done on discovering complementary and related information that is contained in various sources. Though a skilled user with knowledge about relevant sources can easily follow implicit relationships using interfaces from the various information providers this does not hold true for the majority of potential users. Furthermore, these implicit relationships cannot be used to derive some higher level knowledge. Hence, making implicit knowledge on the web explicit and integrating complementary information is one of the main tasks for the next years.

Publishers, libraries and computer scientists engage in developing digital libraries and electronic catalogues to make the publications themselves (digital contents) and their accompanying metadata (bibliographical data about the publications) available and to offer various services based on the provided information. Those libraries and catalogues currently serve certain areas of research, for instance the ACM and IEEE digital libraries providing metadata and full text for computer science in general and the German DBLP service providing metadata for the field of database research. The services these different libraries and catalogues provide vary in their quality, for instance full text versus bibliographical data, quantity and the associated cost model.

Usually citations are used within scientific publications to link the contents to related work or publications containing basic or more detailed descriptions of mentioned concepts. These citations must be considered part of the metadata for a given document. In most systems offering full texts the citations are included in the contents and cannot easily be used. Some systems providing metadata also offer citation metadata as part of it. This is still very problematic, because a system that only covers a certain area or amount of publications cannot guarantee a meaningful resolution of such a citation. For example a publication on the Human Genome Project is very likely to cite publications from the biological as well

as from the computer science area. A researcher looking up such a paper in the DBLP catalogue might be interested in papers from both areas, but in the current situation it is not possible to look up papers that are not in the local system. Therefore, their location cannot be resolved.

This problem is known as citation or reference linking and potential uses of a universal view of citations among scientific publications have been discussed since the early 1950's. More than just making these links "browsable" or (more general) "actionable" towards referenced and (!) to referencing publications the vision right from the beginning was much broader. It can bring additional value in making scientific progress more visible and understandable. Using knowldege discovery techniques can help to classify and cluster work and thus discover development trends, anomalies and dependencies among workgroups and research topics [Gar79].

Though this vision has been around for quite some time now only during the last few years first steps toward the realization have been done. The major problems are integration of metadata from the various available sources and the identification of objects in this global, integrated scope. For the latter problem current developments in the WWW technology and probabilistic approaches for identifying object seem to be promising. New techniques to tackle the problem of metadata integration were developed in the area of integration of information systems, such as federated database systems and mediator based architectures. The integration of digital libraries and electronic catalogues using these approaches is considered one of the main tasks in the digital library community [SC99].

2 Related Work

For certain research areas citation indexes have been published in paper form practically since the beginning of scientific publishing. Well known indices like those provided by the Institute for Scientific Information (ISI) are available as databases and in some cases are also available online. In [Cam97] Cameron proposes a universal database dedicated to keeping citation indexes. The main problem with both above mentioned approaches is that citation information has to be added and maintained manually and that there is no concept of using existing information from a variety of sources already providing this service.

In other projects like CiteSeer and the Open Journal project attempts have been made to automatically discover citation information. The CiteSeer system as described in [BLG98, LGB99] provides a universal database that is build from extracted metadata from scientific publications found on the World Wide Web using standard search engines and specific heuristics. During the CiteSeer project various approaches to identifying same objects in this scenario were investigated.

As part of the Open Journal project a citation service was developed that based on existing citation indexes from the ISI and a citation agent extracting citations information from newly uploaded documents similar to the CiteSeer approach [HCH+97, HCH+98]. Another focus was on making the references "actionable" based on technology from the Distributed Link Service (DLS) [Car97].

While on the one side universal databases for citation indexes and ways to efficiently fill these databases with useful data are proposed, on the other side the problem is being addressed by developing a universal digital identifier for documents and according resolution mechanisms. The currently most relevant initiative is the Digital Object Identifier (DOI) [Pas99a, Pas99c]. Basically its intention is to provide persistent, globally unique names for documents. Those names are independent of their location. The DOI resolution mechanism is based on the Handle System [PC96] developed by the CNRI. A DOi (lower case indicates a single instance) can be used as an Universal Resource Name (URN) according to the guidelines proposed to the World Wide Web Consortium [The96, LPD98]. If the DOI concept becomes widely accepted and used, resolving the citation linking problem becomes easy. First approaches were described in [Pas99b] and [LR99].

As integration of various sources for bibliographical information is one of the major tasks in the development of digital libraries, concepts from the area of information system integration are adopted

to this usage scenario. A federated database system is a distributed system consisting of different heterogeneous database systems or local file clusters [HM85]. These cooperating but *autonomous* local systems are coupled by the federated database management system. The FDBS provides a uniform and transparent interface to the heterogeneous and distributed data sources of independently developed local systems. The FDBS communicates with the local systems using their own interfaces. Therefore, the local database systems and application programs can remain unchanged.

The currently most applicable solutions are based on loosely coupled federated database systems in connection with a multidatabase language and mediator based-systems. Query languages supporting the integration of heterogeneous sources are for instance multidatabase languages like MSQL [GLRS93], SQL/M [KGK+95] and SchemaSQL [LSS96]. Examples of system implementations are federated database systems like IRO-DB [GGF+96], Pegasus [ASD+91] or IBM DataJoiner [VZ98] as well as mediator-based systems like TSIMMIS [GPQ+97], Garlic [CHS+95] or Information Manifold [LRO96].

3 Citation Linking: Fundamentals

In this section we present our approach of citation linking as part of a federation service for digital libraries. The approach is based on FRAQL, a query language for loosely-coupled database federations. In the following we first introduce the basic idea of our citation linking approach. In the next section we will discuss the implementation in FRAQL.

Linking citation references from different bibliographical sources requires global unique keys which identify the document entries. An ideal solution would be the usage of standardized identifiers, like the DOI approach discussed in section 2. However, most of the existing online libraries use their own proprietary format for identifying documents. Therefore, if we want to integrate these sources, we have to deal with these formats. In this context two main problems have to be solved:

(a) guaranteeing the global uniqueness of the identifiers,

(b) resolving non-local references, which refer to documents from another source.

Our approach of a federation service bases on a global integrated schema for all document sources. So, we can define a federated bibliographical library $BibDB$ as a set of named relations containing sets of bibliographical items $BibItem$, whereas $SName$ denotes the set of all source names:

$$BibDB \triangleq SName \times \mathbb{P}BibItem$$

Each $BibItem$ consists of a document identifier (key) $BibKey$, several bibliographical attributes $A_1, \ldots A_n$ like title, authors etc. and a set of references to other documents (citations). Here, a reference is also a document identifier and forms a link:

$$BibItem \triangleq BibKey \times A_1 \times \cdots \times A_n \times \mathbb{P}BibKey$$

Finally, a document identifier $BibKey$ is constructed by combining the source name and the local document identifier denoted by the set OID:

$$BibKey \triangleq SName \times OID$$

A *valid* global document identifier is a tuple $(src, oid) \in BibKey$ with:

(a) src denotes an existing source: $src \in SName$

(b) oid is an existing document identifier in the source src i.e.,
$\exists (src, items) \in BibDB, \exists (id, a_1 \ldots a_n, cites) \in items : oid = id$

With this simple model we are able to build a federated bibliographical library, if we ensure that all document identifiers (document keys and references) are valid. But because each source uses its own identification scheme, a mapping between the global document identifiers and the local identifiers has to be established. This mapping is required for the document keys as well as for the references. Because each existing key uniquely identifies a document, we can create a global document identifier by simply adding the source name. Therefore, the global identifier of the document, denoted by the local identifier id in the source src, is the tuple (src, id).

However, the mapping of references is more complex. Here we have to distinguish between local resolvable references and inter-source references. A *local reference* is a citation of a document from the same source. Here, we can construct the global identifier again by simply adding the source name. An *inter-source reference* is a citation of a document, which is stored in another source. In this case, we have to provide a mapping table, which relates the reference identifier with the global identifier of the referenced document i.e., this table implements a function $f : OID \mapsto BibKey$. Normally, the mapping table has to be built and maintained manually, but data analysis tools may provide support for this task.

4 Citation Linking: FRAQL Implementation

As stated above, our implementation approach is based on FRAQL. This language is a query language for object-relational database federations. It extends SQL by features for defining federations, accessing metadata in queries, restructuring query results, and resolving integration conflicts. This is comparable with other multidatabase languages like MSQL or SchemaSQL, but in contrast to these, FRAQL is extensible by user-defined data types and functions and it supports dynamic integration of new sources.

In FRAQL, a federation is a set of databases consisting of relations. A database can be provided by a full-featured DBMS or even by a Web source encapsulated by a wrapper ([RS97]). This wrapper has to implement the query mechanisms which are not supported directly by the source. FRAQL is based on an object-relational data model: it supports the definition of object types and object tables derived from types. Using object-relational features simplifies the integration of post-relational data sources (e.g. ODBMS-based sources or XML datastores) and provides more advanced modeling concepts for schema definition.

Object types describe the structure of objects as sets of attributes and their domains. Types can be organized in a specialization hierarchy. *Object tables* represent global relations of the federation. Here we distinguish between import and integration relations. An *import relation* is a projection of a local relation of a data source. The import relation is defined by specifying the origin (the identifier of the source) and, if required, a mapping between local and global attributes.

```
create table global_name of type_name as import from source.local_name
  [ mapping_definitions ];
```

In the above table definition, the attribute mapping can be described in the following forms:

- Without an explicit mapping, a local attribute corresponding to an attribute defined by the global object type in terms of identifier and type becomes an attribute of the global relation.
- The notation *g_name* **is** *l_name* means renaming the local attribute to *g_name*. This requires type compatibility.
- The notation *g_name* **is** *func(l_name)* defines that the global attribute value is calculated by using the user-defined function *func* on the local attribute value.
- The definition *g_name* **is** *@tbl (l_name, src, dest, default)* means that the database table *tbl* is used for mapping the values from the local attribute *L name*. This value of the global attribute is obtained by looking for the value of attribute *L name* in column *src* and retrieving the corresponding value of

column *dest*. The field *default* denotes a default value, either as literal or as local attribute, which is assigned to the global attribute, if the value of *L_name* is not found in the table. In fact, this kind of attribute mapping is evaluated by a left outer join, whereas the `NULL` value is replaced by the default value *default*.

An *integration relation* is a view on other global relations combined by using operators like union, θ-join and outer join. In addition, the standard SQL operations selection and projection are provided, too. An integration relation is defined as follows, where the term `table_expression` denotes a SQL view definition with extensions described elsewhere.

```
create table global_name of type_name as table_expression;
```

As an extension to standard SQL, attributes of tuple variables in queries can be obtained during evaluation. This means, while in SQL names of attributes and relations are constants, in FRAQL they can be constructed from current values of other tuple attributes. This *variable substitution* is written in the notation `$var` and can appear anywhere in a query, where names of attributes or relations are expected. For example, the expression `tbl1.$(tbl2.col)` means the attribute of the current tuple of relation `tbl1` whose name is obtained from the current value of `tbl2.col`. In the same way, a relation in the `FROM` clause or a query could be dynamically determined. The following query selects ISBN and title information from all relations implementing the object type `book`. So it is equivalent to a union of all these relations.

```
select t2.isbn, t2.title
from catalog.tables t1, $(t1.name) t2
where t1.type_name = 'book';
```

With the help of these features the global schema as well as the mapping to the local schemata can be defined. In the following we assume for simplification that each local source schema contains only one relation of the form:

$$Publications(Key, Attr_1, \ldots, Attr_n, \{Ref_1, \ldots, Ref_m\})$$

Here Key denotes the document identifier, $Attr_i$ a bibliographical attribute (author, title, journal etc.) and Ref_j the citation as a document reference. Because of the currently limited capabilities of the FRAQL data model—it lacks support for multi-valued attributes—this relation has to be transformed into normal form. For this purpose, two object types `PublType` for the "publication" entities and `RefType` for the "references" relationship are defined:

```
create type PublType (
  id varchar(20),
  source varchar(20),
  title varchar(100),
  authors varchar(100));
```

```
create type RefType (
  id varchar(20),
  source varchar(20),
  ref_id varchar(20),
  ref_source varchar(20));
```

In both type definitions the attribute pairs (`id`, `source`) and (`ref_id`, `ref_source`) form the document identifier according to the definition in section 3. Furthermore, the source relation `Publications` is transformed into first normal form by the FRAQL source adapter. So, an example relation exported by the adapter may look like shown in Fig. 1.

Based on these types the FRAQL import relations are specified. First of all, for each source *N* the relation `Publications` is partitioned into a relation `Publ_`*N* of type `PublType` and a relation `Ref_`*N* of type `RefType`. Furthermore, the global attribute `source` of relation `Publ_`*N* is initialized with a literal name identifying the source (in our case the name of the relation `Publ_`*N*):

Key	*Title*	*Authors*	*References*
Ada98	Databases	Adams	Sco98
Ada98	Databases	Adams	Mey92
Sco98	Dig. Libraries	Scott	Moo94

(a) Local Relation *Publications*

id	*source*	*title*	*authors*
Ada98	Publ_1	Databases	Adams
Sco98	Publ_1	Dig. Libraries	Scott

(b) Import Relation *Publ_1*

Figure 1: Publications Relations

```
create table Publ_1 of PublType as
  import from src_1.Publications (
    id is Key,
    source is "Publ_1",
    title is Title,
    authors is Authors);
```

For the example relation from Fig. 1(a) the resulting contents of the import relation is shown in Fig. 1(b).

In order to map the inter-source references we have to provide the mapping table mentioned in section 3, which relates the local reference identifiers to the global identifiers. This mapping table is stored in a local component database of the federation and has to be built by hand. For the example in Fig. 1(a) the table contains the values shown in Fig. 2(a). It can be used for mapping the reference attributes (`ref_id` and `ref_source`) of the import relation `Ref_1` (Fig. 2(b)):

```
create table Ref_1 of RefType as
  import from src_1.Publications (
    id is Key,
    source is "Publ_1",
    ref_id is @tbl1 (References, key, id, References),
    ref_source is @tbl1 (References, key, source, "Publ_1"));
```

After importing the source relations into a homogeneous model we are able to integrate them by applying the union operator. This can be done explicitly in the form `Publ_1 union Publ_2 union ... union Publ_n` or by using the advanced metadata facilities provided by FRAQL:

```
create table Publications of PublType as
  select t2.*
  from catalog.tables t1, $(t1.table_name) t2
  where t1.type_name = "PublType";

create table References of RefType as
  select t2.*
  from catalog.tables t1, $(t1.table_name) t2
  where t1.type_name = "RefType";
```

With these both relations the retrieval of documents referenced by a given document with identifier `"Ada98"` is straightforward:

```
select t2.id, t2.title, t2.authors
from References t1, Publications t2
where t1.id = "Ada98" and
      t1.source = "Pub_1" and
      t1.ref_id = t2.id and
      t1.ref_source = t2.source;
```

key	*id*	*source*
Moo94	Moo94	Publ_2
Mey92	Mey92a	Publ_3

(a) Mapping Relation *tbl1*

id	*source*	*ref_id*	*ref_source*
Ada98	Publ_1	Sco98	Publ_1
Ada98	Publ_1	Mey92a	Publ_3
Sco98	Publ_1	Moo94	Publ_2

(b) Import Relation *Ref_1*

Figure 2: References Relations

However, the evaluation of the above query requires access to all sources, because it cannot be determined which source relation contains the document. But this information is available in attribute `source`. Therefore, we can formulate a more efficient query by using the FRAQL mechanism for variable substitution:

```
select t2.id, t2.title, t2.authors
from References t1, $(t1.ref_source) t2
where t1.id = "Ada98" and
      t1.source = "Pub_1" and
      t1.ref_id = t2.id;
```

This query is evaluated in the following steps:

(1) In relation `References` the tuples with value ”Ada98” for attribute `id` are selected as result set `t1`.

(2) From the resulting tuples the names of the relations containing the referenced entries are determined.

(3) In these relations the tuples with `id` equals to `ref_id` are selected as result set `t2`.

(4) The cross product of both result sets is computed followed by the projection according the `select` clause.

However, this query reformulation should be done automatically by the query processor. Currently, we are investigating appropriated query optimization techniques.

5 Prototype implementation

The previous sections presented concepts and methods to support the integration of metadata in the field of digital libraries. The subject of this section is the description of a prototype providing metadata of citations in a heterogenous environment. For a single point of access to metadata via a uniform interface an appropriate federated system is required. In principle the architecture of such a system comprises four layers: an application layer, a federation layer, an adapter layer and a data layer. On the *data layer*, all metadata of citations in this scenario are provided preserving the autonomy of data sources. For the representation of metadata in a uniform data format special adapters are introduced on the *adapter layer*. With an adapter the data source specific data models are transformed. The query processing and the integration of metadata from several data sources takes place on the *federation layer*. All relevant functionality is provided to the users in form of an object-relational interface on the *application layer*. In this application scenario the users have a uniform access interface for citation metadata from several data sources.

The adapter layer has a special significance in this scenario, because the adapters implement the access to heterogenous data sources. In principle several adapters are conceivable in this area and the

generation of adapters is subject of current research work. For example an appropriate adapter for accessing HTML pages is presented in [SH99]. In the following, a *generic adapter* for the access of metadata in XML [BPSM98] format is discussed. The usage of XML is an approach to exchange metadata about citations in the above scenario. All XML documents have their own provider-specific DTD.

The approach for the XML adapter is based on the assumption that all metadata are stored in form of objects with appropriate attributes. The attributes contain all relevant information. Additionally, the attributes are subdivided into simple and complex attributes, the latter containing other attributes. During the exchange of metadata about citations different objects like books, articles, or proceedings are considered. Therefore the XML adapter is implemented on the basis of meta object types according to the object-relational model of the federation layer.

For the usage of object types the introduction of a mapping between the provider specific DTD and the adapter specific DTD is required. XML data are transformed to corresponding object types by an appropriate parser. Due to the provider specific DTD the necessary mapping information are specific for each provider. For the realization of this mapping we propose the application of XSLT [Cla99] in the adapter. XSLT is a stylesheet language for transforming XML documents into other XML documents. It is used in the adapter for transforming the provider specific XML data into XML data suitable for processing by the adapter. Hence, not only the special XML files but also an appropriate XSLT file is required as an input for the adapter.

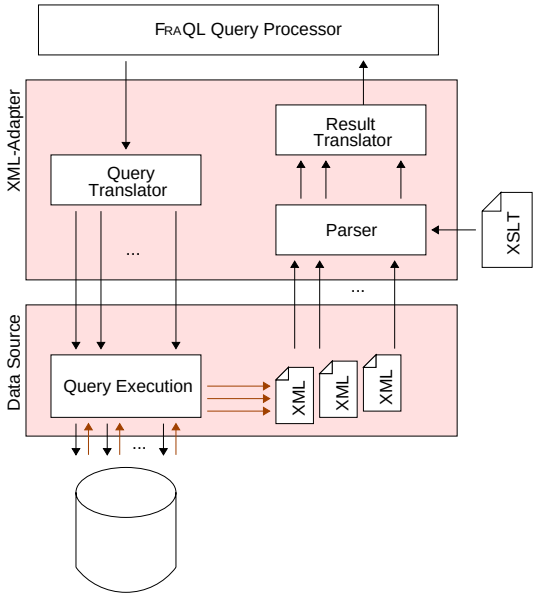

Figure 3: Architecture for XML Adapters

Finally, in Fig. 3 the architecture of our generic XML adapter is specified. In principle, the architecture consists of four parts: a query translator, an XML parser and a result translator. By using a *query translator*, requests to the adapter are transformed into specific requests for special data sources and those requests are transmitted. In the data source the requests are executed. The execution results are made available for the adapter in XML format. By means of a generic XML parser and the usage of appropriate XSLT information the special XML files are transformed into an internal representation. In this representation generic object types and according attributes are used. Based on this the results are transformed in compliance with the request in the *result translator*. In general, the query expression generated by the federation layer is parsed in the adapter. A source description specifying the query capabilities of the source is part of the adapter configuration and is currently stored in a separate XML file. Based on this, a query statement for the data source is generated in form of a simple CGI request.

Using the programming language Java and the ORACLE XML Parser classes, a prototype of the

generic XML adapter was implemented. In this prototype, the approach based on object types and attributes for the generic XML adapter was verified. The prototype is currently being tested on various XML citation metadata in a heterogenous environment.

6 Application

In this section, we briefly describe some possible usage scenarios for the described system. An application can use citation information through interfaces provided by the federation layer, for instance by executing SQL statements using the JDBC interface. Examples for basic usage scenarios could be the following:

Publications citing a given publication: Once we have found a publication we can easily retrieve information from various sources by simply executing the following standard SQL statement.

```
select p.source, p.id, p.title, p.authors
from References r, Publications p
where p.id = r.id and p.source = r.source
    and r.ref_id = "90140218" and r.ref_source = "SpringerLink";
```

Using a computationally complete language, a closure of a certain depth can easily be derived from an originating publication that can be used as a starting point for studies in a certain area.

Most often cited publications: Statistical information derived from citation information can be used as a very simple measure for the usefulness of a publication.

```
select count(*) as num, p.title, p.authors
from References r, Publications p
where r.ref_id = p.id and r.ref_source = p.source
group by r.ref_source, r.ref_id
having count(*) > 10
order by num desc;
```

This query returns all publications that are referenced more than ten times.

We present here only very few use cases, but a large class of useful information can be extracted in this simple way.

By integrating a number of sources we gain access to a large amount of data that can again be used to apply *Knowledge Discovery in Database* (KDD) techniques and thus extract higher level information that might be useful. A simple scenario could be the following: based on the stored citation data we can build clusters [CHY96, FPSM92] by finding medoids that represent central objects. Those medoids are representative for a certain cluster of objects by having a minimal overall distance (citation steps) from objects within this cluster and on the other hand are most distinct from all other medoids found. Those clusters can be used in different ways:

- An alerting service for instance can inform a user about publications that are most likely interesting for him, because the new publication cites papers within a cluster the user marked as interesting for him. The user profile, i.e. “what is interesting” for the user, can as well be gathered automatically from previous query results or traversals.
- The clusters themselves can be considered useful information, as they represent certain research fields and/or working groups. Within these clusters we can easily apply measures as to which publications are “essential”, i.e., are cited often from within the cluster, or “basic”, i.e., cited often but do not cite many other papers within the cluster.

We are planning to investigate how well known techniques from the area of KDD can be applied in the area of database federations [SS99] and especially in the application area of citation linking.

7 Conclusion

We presented an approach to integrate citation linking information available in various sources based on concepts from the area of federated database systems. For this purpose, we used the FRAQL language which provides mechanisms for resolving conflicts arising from the process of the integration of heterogeneous data sources. Furthermore, we described an architecture based on the FRAQL system that allows seamless integration of a great variety of different sources.

The proposed approach can be considered a platform for the representation of citation information that can be used to build an integrated citation index. It is not yet a fully functional solution that satisfies all real life requirements that would exist for such a system. We consider the building process of the described mapping table as the most critical issue. Building such a table manually is not a feasible solution in most cases, but we believe that our approach provides a general basis for the automatic generation. On the one hand, we can use the provided integration platform to import and combine various existing citation indexes. On the other hand, we can provide additional functionality and a locally replicated subset of metadata that can help to identify objects and assign them a unique global identifier of the described format. Whenever publications from a source of bibliographical data are accessed, the following process is can be triggered: the source name, the local identifier and a candidate key sufficient to identify the abstract representation of a work in a global scope (in most cases probably *title* and *authors*) is extracted and matched against an existing table of these cached information. This can be done using techniques from statistics and data mining, like described in [BLG98] and [Nei98]. If we find a match, we can add the key and link the newly found local identifier to the match found as another manifestation, thereby establishing a structure similar to DOI nesting. In this case, we can discard the now duplicate data required to identify the object. If we fail to find such a match, the full record of the global key plus its identifying metadata is cached.

The proposed system would not conflict with the DOI infrastructure, which will provide a general solution for the citation linking problem, but instead could complement and support it in the early stages by providing DOi's for systems that do not yet support DOI and help to look up DOi's based on bibliographical data.

References

[ASD+91] R. Ahmed, P. De Smedt, W. Du, W. Kent, M.A. Ketabchi, W. Litwin, A. Rafii, and M.-C. Shan. The Pegasus Heterogeneous Multidatabase System. *IEEE Computer*, 24(12):19–27, December 1991.

[BLG98] K. D. Bollacker, S. Lawrence, and C. Lee Giles. CiteSeer: An Autonomous Web Agent for Automatic Retrieval and Identification of Interesting Publications. In K. P. Sycara and M. Wooldridge, editors, *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, pages 116–123, New York, May 9–13, 1998. ACM Press.

[BPSM98] T. Bray, J. Paoli, and C.M. Sperberg-McQueen. Extensible Markup Language (XML) 1.0. Technical report, World Wide Web Consortium, February 1998.

[Cam97] R. D. Cameron. A Universal Citation Database as a Catalyst for Reform in Scholarly Communication. *First Monday*, 2(4), 1997.

[Car97] L. Carr. Using a Link Service for Citation Linking. In *Proceedings of the Eighth ACM Conference on Hypertext – Posters and Demonstrations*, Demonstrations, 1997.

[CHS+95] M.J. Carey, L.M. Haas, P.M. Schwarz, M. Arya, W.F. Cody, R. Fagin, M. Flickner, A. Luniewski, W. Niblack, D. Petkovic, J. Thomas II, J.H. Williams, and E.L. Wimmers. Towards Heterogeneous Multimedia Information Systems: The Garlic Approach. In Omran A. Bukhres, M. Tamer Özsu, and Ming-Chien Shan, editors, *Proc. RIDE-DOM '95, 5th Int. Workshop on Research Issues in Data Engineering - Distributed Object Management, Taipei, Taiwan*, pages 124–131. IEEE-CS, 1995.

[CHY96] M. Chen, J. Han, and P. S. Yu. Data mining: an overview from a database perspective. *IEEE Trans. On Knowledge And Data Engineering*, 8:866–883, December 1996.

[Cla99] J. Clark. XSL Transformations (XSLT) – Version 1.0. Technical report, World Wide Web Consortium, November 1999.

[FPSM92] W. J. Frawley, G. Piatetsky-Shapiro, and C. J. Matheus. Knowledge discovery in databases - an overview. *Ai Magazine*, 13:57–70, 1992.

[Gar79] E. Garfield. *Citation Indexing: Its Theory and Application in Science, Technology, and Humanities*. Wiley, New York, 1979.

[GGF+96] G. Gardarin, S. Gannouni, B. Finance, P. Fankhauser, W. Klas, D. Pastre, R. Legoff, and A. Ramfos. IRO-DB — A Distributed System Federating Object and Relational Databases. In O. A. Bukhres and A. K. Elmagarmid, editors, *Object-Oriented Multidatabase Systems — A Solution for Advanced Applications*, chapter 20, pages 684–712. Prentice Hall, Eaglewoods Cliffs, NJ, 1996.

[GLRS93] J. Grant, W. Litwin, N. Roussopoulos, and T. Sellis. Query Languages for Relational Multidatabases. *VLDB Journal*, 2(2):153–171, 1993.

[GPQ+97] H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J. D. Ullman, V. Vassalos, and J. Widom. The TSIMMIS Approach to Mediation: Data Models and Languages. *Journal of Intelligent Information Systems*, 8(2):117–132, March/April 1997.

[HCH+97] S. Hitchcock, L. Carr, S. Harris, J. M. N. Hey, and W. Hall. Citation Linking: Improving Access to Online Journals. In *DL'97: Proceedings of the 2nd ACM International Conference on Digital Libraries*, Digital Scholarship, pages 115–122, 1997.

[HCH+98] S. Hitchcock, L. Carr, W. Hall, S. Harris, S. Probets, D. Evans, and D. Brailsford. Linking electronic journals: Lessons from the Open Journal project. Technical report, D-Lib Magazine, December 15, 1998.

[HM85] D. Heimbigner and D. McLeod. A Federated Architecture for Information Management. *ACM Transactions on Office Information Systems*, 3(3):253–278, July 1985.

[KGK+95] W. Kelley, S. Gala, W. Kim, T. Reyes, and B. Graham. Schema Architecture of the UniSQL/M Multidatabase System. In W. Kim, editor, *Modern Database Systems*, chapter 30, pages 621–648. ACM Press, New York, NJ, 1995.

[LGB99] S. Lawrence, C. Giles, and K. Bollacker. Digital Libraries and Autonomous Citation Indexing. *IEEE Computer*, 32(6):67–71, 1999.

[LPD98] C. Lynch, C. Preston, and R. Daniel. RFC 2288: Using Existing Bibliographic Identifiers as Uniform Resource Names, February 1998. Status: INFORMATIONAL.

[LR99] C. Lyons and H. Ratner. DOIs Used for Reference Linking: DOI-X. Technical report, International DOI Foundation, October 8, 1999.

[LRO96] A.Y. Levy, A. Rajaraman, and J.J. Ordille. Querying Heterogeneous Information Sources Using Source Descriptions. In T.M. Vijayaraman, A.P. Buchmann, C. Mohan, and N.L. Sarda, editors, *VLDB'96, Proc. of 22th Int. Conf. on Very Large Data Bases, 1996, Mumbai (Bombay), India*, pages 251–262. Morgan Kaufmann, 1996.

[LSS96] L.V.S. Lakshmanan, F. Sadri, and I.N. Subramanian. SchemaSQL - A Language for Interoperability in Relational Multi-Database Systems. In T. M. Vijayaraman, A.P. Buchmann, C. Mohan, and N.L. Sarda, editors, *VLDB'96, Proc. of 22th Int. Conf. on Very Large Data Bases, 1996, Mumbai (Bombay), India*, pages 239–250. Morgan Kaufmann, 1996.

[Nei98] M. Neiling. Data Fusion with Record Linkage. In I. Schmitt, C. Türker, E. Hildebrandt, and M. Höding, editors, *3. Workshop 'Foederierte Datenbanken'*, Aachen, 1998. Shaker Verlag.

[Pas99a] N. Paskin. DOI: Current Status and Outlook May 1999. Technical Report may99-paskin, D-Lib Magazine, May 17, 1999.

[Pas99b] N. Paskin. DOIs Used for Reference Linking. Technical report, International DOI Foundation, March 16, 1999.

[Pas99c] N. Paskin. Towards unique identifiers. *Proceedings of the IEEE (USA)*, 87(7):1108–1227, July 1999.

[PC96] V. Puvvada and R. H. Campbell. Inverse Mapping in the Handle Management System. In *DL'96: Proceedings of the 1st ACM International Conference on Digital Libraries*, Posters, page 180, 1996.

[RS97] M.T. Roth and P.M. Schwarz. Don't Scrap It, Wrap It! A Wrapper Architecture for Legacy Data Sources. In M. Jarke, M.J. Carey, K.R. Dittrich, F.H. Lochovsky, P. Loucopoulos, and M.A. Jeusfeld, editors, *VLDB'97, Proc. of 23rd Int. Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, pages 266–275. Morgan Kaufmann, 1997.

[SC99] B. Schatz and H. Chen. Digital Libraries: Technological Advances and Social Impacts. *Computer*, 32(2):45–50, February 1999.

[SH99] K.-U. Sattler and M. Höding. Adapter Generation for Extracting and Querying Data from Web. In S. Cluet and T. Milo, editors, *ACM SIGMOD Workshop on The Web and Databases (WebDB'99)*, pages 49–54, Philadelphia, Pennsylvania, USA, June 1999. Informal Proceedings.

[SS99] K. Sattler and G. Saake. Supporting Information Fusion with Federated Database Technologies. In S. Conrad, W. Hasselbring, and G. Saake, editors, *Proc. 2nd Int. Workshop on Engineering Federated Information Systems, EFIS'99, Kühlungsborn, Germany, May 5–7, 1999*, pages 179–184. infix-Verlag, Sankt Augustin, 1999.

[The96] The URN Implementors. Uniform Resource Names: A Progress Report. *D-Lib Magazine*, February 1996.

[VZ98] S. Venkataraman and T. Zhang. Heterogeneous Database Query Optimization in DB2 Universal DataJoiner. In A. Gupta, O. Shmueli, and J. Widom, editors, *VLDB'98, Proc. of 24rd Int. Conf. on Very Large Data Bases, 1998, New York City, New York, USA*, pages 685–689. Morgan Kaufmann, 1998.