Case Study: Creating a Dataweb of Financial Service Data Using Infomaster

Arthur M. Keller, Meikel Pöss, Michael R. Genesereth*
Kenneth J. Montoro and Denis Papathanasiou †

June 26, 1998

Abstract

We have created a "dataweb" of financial services data for Merrill Lynch using Infomaster. Merrill Lynch has about 20 separate systems containing data on commercial customer accounts. There are many commercial customers with multiple relationships with Merrill Lynch and therefore have accounts represented in multiple systems. There is common data in multiple customer records across the different systems that can be used to link them. However, the common data is entered manually and does not have referential integrity across the systems. Merrill Lynch is creating the Cross-Reference Utility (CRU) to link the customer data together. CRU is used both for inquiries by account representatives and for cleaning up the incorrect cross-system references. Stanford's Center for Information Technology is experimenting with Infomaster as a "shadow" project to CRU. The primary differences between the CRU approach and the Infomaster approach are: (1) CRU assigns CRU IDs to each customer, so there is a black and white decision as to whether two account records refer to the same customer. In contrast, Infomaster has multiple shades of gray of record matching since it creates multiple lists of matching records across systems based on which fields or combinations of fields match. (2) CRU uses an algorithmic approach that accesses the systems in a defined order. In contrast, Infomaster uses a rule based approach that allows for an arbitrary order of traversing the data from the systems. (3) CRU processes the input data all in memory and, after assigning CRU

*Stanford University, Computer Science Dept., Gates Building 2A, Palo Alto, CA 94305-9020, contact Author: Arthur M. Keller keller@cs.stanford.edu

†Merrill Lynch Operations, Services Technologies, 101 Hudson Street, Jersey City, NY 07302-3997

IDs and detecting diagnostics, writes the rules in an Oracle database; a separate application allows web clients to browse and search the database. In contrast, the same Infomaster materializes the various connections in memory and serves a rich interconnected dataweb for search and browsing to web clients. This paper describes the shadow Infomaster-based project.

1 Introduction

Merrill Lynch has approximately 20 systems containing data on commercial customers. These systems were usually developed independently from each other. Sometimes the systems are separate because they are for different kinds of accounts, such as working capital versus retirement funds. Sometimes the systems came with a investment company acquired by Merrill Lynch.

Account managers working with a commercial customer would like to be able to go to a single place to get all the account data on that customer in all of the systems. While such a capability is useful in enhancing customer service, this capability can also satisfy legal requirements. When a company declares bankruptcy, such as Barings Bank a few years ago, Merrill Lynch must freeze trading on all the accounts associated with the bankrupt company. Finding all of those accounts is a challenge.

Over time, cross references were added from one system to another. But the values of these cross references were entered manually and without enforcing referential integrity. Yet these cross references represent shared values that can be used to link the records of the various systems.

The integration of these commercial customer systems serves two communities. First, account managers look up their clients and get linked records across the integrated systems. Second, system maintainers use the diagnostics reported to manually clean up the cross references.

Merrill Lynch created the Cross-Reference Utility (CRU) to provide an integrated view of these accounts. In the first iteration, the CRU system integrated three of these systems: IPS, TESS, and CTL. In the second iteration, the CRU system was then enhanced to include XTAS-MERL.

In a shadow project, Stanford used Infomaster first to integrate IPS, TESS, and CTL, and then included XTAS-MERL in the integration.

2 The Problem

l A simplified ER-Diagram of the four systems being integrated is given in Figure 1. The systems *XTAS*, *TESS* and *IPS* consist of 2 relations each. *Client* contains all the client information such as Client Name and Type, while *Account* describes client accounts. Each client can own multiple accounts. The architecture of the *CTL* system differs in that it contains solely client information.

Clients and accounts can be represented in each system. In XTAS, TESS and IPS clients have the same *symbol*, accounts have the same *AP-number* in each of the four systems. *CTL* clients are only linked to TESS using the *Broker Code*.

Table 1 gives an overview of the number of clients, accounts and the average number of accounts each client owns.

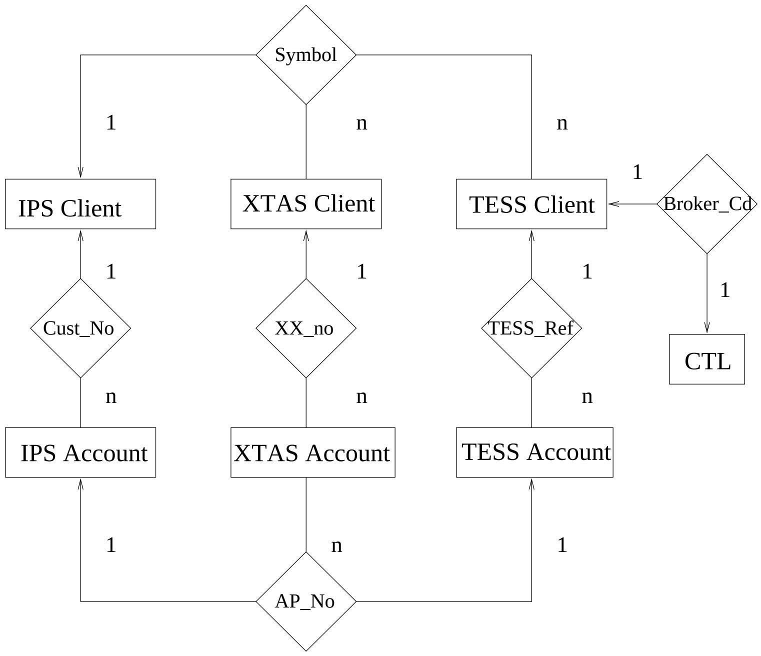

Figure 1: *ER-Diagram of the 4 systems*

3 The CRU Approach

The goal of the Merrill Lynch Cross-Reference Utility is to provide a mechanism that will take daily client data feed files from various systems, and to cross-reference the data in such a way that will provide a singular view of a client across all systems. A view of the data should be easily accessible and the implementation should allow for viewing through the Merrill Lynch intranet by our internal business community. This goal is achieved through the CRU Update Process and the CRU Web Interface.

Relation	**number of Records**	**avg. account per client**
XTAS Client	77,182	1.55
XTAS Account	119.626	
TESS Client	30,308	3.01
TESS Account	91,238	
IPS Client	6,932	26.86
IPS Account	168,247	
CTL	38,320	

Table 1: *Statistics about the relations*

As the project has evolved, so has the requirements and implementation. Initially, the CRU development team cross-referenced three systems, IPS, TESS, and CTL, and addressed the requirement to provide warnings about data integrity problems. The second phase saw the addition of an additional system, XTAS-MERL, as well as the requirement to able to search for common internal and external identifiers that may not have been cross-referenced. In addition, based on the findings from the first phase, the cross-reference rules were re-worked to generate only client-level cross-references.

Future phases see the addition of a data filter that will allow the ability to limit the views based on geographic region and business type. The complexity of the problem and the volume of data that the CRU processes is constantly increasing, and the underlying design of the CRU takes the audience and scope of future phases into consideration.

3.1 The CRU Process

The CRU Process has as its tasks:

1. To provide the ability to add additional systems with fields common to existing cross referenced data in an efficient manner that does not require a major re-write of the system.

2. To be adaptable to changing business requirements and a constantly increasing volume of data and business users from around the globe.

3. To accommodate daily changes to the data. Update files for each system arrive at the end of every business day. The data, impacted

cross-references, and data integrity statistics are updated based on the daily feed.

4. To flag potential error conditions and provide a mechanism to have the warnings accessed through the interface.

3.1.1 Adding a System to the CRU

In order to add a system to the CRU, the following procedure is followed:

1. Analysis:

   (a) Obtain a data file extract from the system to be cross-referenced that contains all common and unique client and account identifiers. The format of a feed file varies depending on the systemwe cannot depend on receiving data in a homogenous form for all systems. The feed is analyzed to determine valid fields to use in the CRU data model and the method by which it will be loaded into the database.

   (b) Determine the set of rules that would indicate a client level cross-reference with another system. A representation of the client-level cross reference is held by the CRU_ID identifier. The CRU_ID is a transient numerical representation of the client cross-reference. Clients with the same CRU_ID number are considered to be the "same" client, however, the specific mapping and numerical value of the CRU_ID may change from day to day based on daily updates. For example, a client record may map to one set of records on one particular day with a particular CRU_ID. Based on a data correction the following day, it may no longer meet the matching criteria and would be assigned another CRU_ID, based on a different rule.

   There has been discussion that the CRU could potentially maintain persistent CRU_IDs. This would require the process to override certain cross-reference rules and make decisions about CRU_ID mapping that ignore data anomalies. The current CRU matching rules are described in Figure 2.

   The matching rules currently propagate from IPS. Since AP and Symbol are native IPS identifiers, IPS is processed first, followed

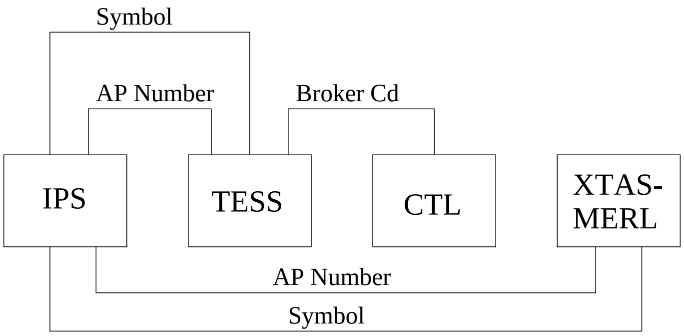

Figure 2: *Current CRU matching rules used for CRU_ID determination*

by TESS, XTAS-MERL, and CTL. The fields chosen for matching are considered to be the most reliable. AP number is a secondary matching field to symbol, since the data in this field in non-IPS systems is less reliable. In non-IPS systems, the AP field is manually updated and maintained.

While there are other external identifiers such as BIC and SIB, they are not used in the cross- reference. Integrating them into the rules would produce ambiguous cross-references, since matches on BIC and SIB often conflict with matches on internal identifiers. In the interest of providing a clear internal cross-reference, the external identifiers are not used as matching fields.

(c) Determine a list of what error conditions should be flagged. Conditions that are determined by the business community to be useful to troubleshoot data integrity problems are stored in a separate table in the database. The update process also generates the static HTML formatted pages on a nightly basis that are part of the CRU interface. Thus, the diagnostic data can be retrieved more quickly through the interface, rather than generating the output at the time of the user request.

2. Implementation:

   (a) Create the tables necessary for the system in the Oracle database. A given system feed may hold any combination of client and account data in any form, however, the table structure it is loaded into must be as consistent as possible across systems.

(b) Load a full feed of the system data into the database and map it to the existing systems using the cross-reference rules.

(c) Create the necessary configuration files and classes/subclasses necessary to integrate the new system into the nightly update process.

3.1.2 Nightly Feed Processing

Update processing for the CRU is done in a batch manner. An automated process delivers the nightly extracts to the CRU server. The CRU update process must adapt itself to the various feed formats, and implement the desired set of matching rules for each system. Some systems may send updates only, whereas other systems may send all of its data, thus requiring the CRU update process to determine whether or not it is actually an update. Future phases may require the ability to filter data from a single feed based on geographic region and/or business type.

The design of the update process is object-oriented, and it takes advantage of configuration files that define the cross-referencing and parsing rules for each system. The data feed handler validates the extract and makes it available to the process for processing.

3.2 The CRU Web Interface

The CRU interface is comprised of two parts:

1. Showroom:
   The showroom provides two different types of queries to access data held by the CRU:

   (a) Client Level cross-reference queries to search for clients common across all systems. This search uses the CRU ID generated by the update process as the linkage between clients across all systems.

   (b) A global search across all systems included in the CRU database using common identifiers. (Symbol, Client Name, AP Number, AE Number). The Search Facility does not concern itself with cross-references, rather, it is a straight query of the identifiers held by the CRU.

   The screens are designed to be clearly readable and usable by the business user community. It selects fields from the Oracle database based on user input. The fields chosen for input and display are defined by business requirements decided in the analysis phase. The showroom design takes advantage of configuration files and scripts, which allow for adaptability as future systems are added.

The showroom delivers information about client cross-references and diagnostics in a standard format so it is identical across systems. The user can choose to view either client or account level results. If the results are client level, a link is provided to view the accounts for that particular client.

The user can also choose whether to filter data that has a diagnostic associated with it. If the user wishes to view data with a diagnostic associated with it, a link is provided to view details about the diagnostics.

2. Diagnostics:
   These are data inconsistency reports and summary information that is updated daily and displayed through the interface. HTML pages are produced the end of the nightly update process and are linked to the web interface. The process uses data stored in the Diagnostics repository and the system tables to produce information about data anomalies, and formats the data in a uniform manner.

3.2.1 Future Considerations for CRU

A future consideration for the interface is the need to filter the presentation of data based on geographic region and business type. The CRU will expand its current login profile scheme to allow each user or group view access based on their specific permissions.

4 The Infomaster Approach

4.1 Infomaster

Infomaster is a general purpose heterogeneous database integration system that includes a main memory database and a rule-based reasoning engine. Infomaster includes both relations and classes. The relations are n-ary. Classes have a symbolic object ID and binary relations provide the mappings between the symbolic object ID and the (potentially multiple) values for the slots of the class. Any of these relations may be stored (e.g., in a database, a file, or in Infomaster's memory) or computed as needed. Rules are used to define a relation in terms of others. A relation may be materialized based on its definitions and the current values used by these definitions. A class may be defined that use data from relations, and a relation may be defined that use data from classes.

The developer using Infomaster uses an iterative process to write rules. First, one or more rules are written to the rule library. Then a query is given to test whether the new rules are correct. The rules are changed and then tested until they produce the correct results. If desired, the results of the rule can be materialized for processing by subsequent rules.

Currently, we are not using the distributed processing features of Infomaster. However, as we integrate more customer account systems, we plan to experiment with distributing this data over multiple servers.

A key feature of Infomaster we are using is the presentation of all the data as a dataweb. A *dataweb* is a richly interconnected hyperlinked collection of concepts supported by data derived from one or more sources. We have arranged a variety of cross references across the various records that makes it easy to explore the commercial customer data through integrated queries and navigation.

4.2 How Infomaster Is Used

Each of the nine base relations is represented as its own relation in memory. In addition to the fields in the base relations, we add a key for each relation while loading. This will make navigation through the system easier. In addition, from each relation we create a class. Within a class each field can have multiple values. This is very useful to express 1:n relationships.

After the base relations are loaded into memory, various views to inspect the data are materialized. Views are defined using rules.

A rule consists of a head and a body. The head is represented by a relation and variables, while the body can consists of multiple relations in a conjunctive or disjunctive way. Each variable occurring in the head must also occur in one of the body relations. An example of such a rule is:

```
(<= (TESS_CL_AC ?TC ?TA)
      (TC_TESS_REF_NO ?TC ?TESS_REF_NO)
      (TA_TESS_REF_NO ?TA ?TESS_REF_NO))
```

This rule expresses the relation between *TESS clients* and *TESS accounts*. The TC_TESS_REF relation consists of all pairs of *TESS client* *key* and *TESS Ref Number*, which is the foreign key for *TESS account*. The TA_TESS_REF relation consists of all pairs of *TESS account key* and *TESS Ref Number*, which is the primary key in *TESS account*. Hence, *TESS_CL_AC* contains all pairs of keys to *TESS clients* and *TESS accounts* who have the same *Reference Number*.

A more complicated rule is the following:

```
(<= (TC_Diagnostics ?tc ?diagn_id)
    (OR (IB_TESS_CLIENT1 ?diagn_id ?tc)
        (Inc_Sym_AP_XT_TC ?diagn_id ?tc)))
```

This rule maps all diagnostics that are relevant for *TESS Client* to the *TESS Client* class. Notice that this is a disjunctive rule of two relations *IB_TESS_CLIENT1* and *Inc_Sym_AP_XT_TC* . *IB_TESS_CLIENT1* contains all clients that have inconsistent bic numbers within *TESS* and *Inc_Sym_AP_XT_TC* points to all clients who have an inconsistent symbol for a matching ap number in one of its accounts. After execution of this rule the relation *TC_Diagnostics* contains the union of both diagnostics.

The four systems are maintained separately and therefore client and account information can be inconsistent across the different systems. To find those clients several diagnostics are introduced. Diagnostic "*Inconsistent Symbol for Matching AP: System X to System Y*" links clients from System X to clients from System Y based on their accounts in those systems. Clients who have different values for the field *symbol* are then entered into the diagnostic. This diagnostic exists for all pairs of the systems TESS, XTAS and IPS.

```
(<= (inconsistent_symbol_for_matching_ap_tess_ips
        ?AP ?TA ?TC ?TC_SYMBOL ?IC_SYMBOL ?IC ?IA ?AP)
    (TA_IA_AP ?TA ?AP ?IA)
    (TESS_CL_AC ?TC ?TA)
    (IPS_CL_AC ?IC ?IA)
    (TC_IPS_CLIENT_SY ?TC ?TC_SYMBOL)
    (IC_CUST_SYMBOL ?IC ?IC_SYMBOL)
    (DISTINCT ?TC_SYMBOL ?IC_SYMBOL))
```

This rule starts with a list of TESS Account records and IPS Account records that match in AP Number. The rule then maps to corresponding TESS Client and IPS Client records, and then looks up the IPS Symbol in each of these records. If they do not match (i.e., if they are DISTINCT), then a result tuple is assembled for this combination.

The class *Symbol_Record* links client records from IPS, TESS, and XTAS based on the field *IPS Symbol*. Since the *CTL system* does not have the attribute *IPS Symbol*, a *CTL* client is linked to the *TESS* client with the matching value for *Broker Code*. The *AP Number* relation links account records from IPS, TESS, and XTAS using the field *AP Number*.

The following definitions describe the *Symbol_Record* class.

```
(ISA SYMBOL_RECORD CLASS)
(SLOT SYMBOL_RECORD SY_SYMBOL)
(ISA SY_SYMBOL RELATION)
(ARITY SY_SYMBOL 2)
(SLOT SYMBOL_RECORD SY_IPS_CLIENT)
(ISA SY_IPS_CLIENT RELATION)
(ARITY SY_IPS_CLIENT 2)
(SLOT SYMBOL_RECORD SY_TESS_CLIENT)
(ISA SY_TESS_CLIENT RELATION)
(ARITY SY_TESS_CLIENT 2)
(SLOT SYMBOL_RECORD SY_XTAS_CLIENT)
(ISA SY_XTAS_CLIENT RELATION)
(ARITY SY_XTAS_CLIENT 2)
(SLOT SYMBOL_RECORD SY_CTL)
(ISA SY_CTL RELATION)
(ARITY SY_CTL 2)
```

The following rules populate the *Symbol_Record* class.

```
;;;;;Symbol-Record: general
;;;;;Define linking relations
(<= (REF_IC_SY ?IC ?Id)
    (IC_CUST_SYMBOL ?IC ?SY)
    (not_null ?sy)
    (SY-SY_ID ?SY ?ID))
(<= (REF_TC_SY ?TC ?Id)
    (TC_IPS_CLIENT_SY ?TC ?SY)
    (not_null ?sy)
    (SY-SY_ID ?SY ?ID))
(<= (REF_XTC_SY ?XTC ?Id)
    (XTC_IPS_CLIENT_SY ?XTC ?SY)
```

```
    (not_null ?sy)
    (SY-SY_ID ?SY ?ID))
;;;;;Declare type of object as Symbol_Record
(<= (ISA ?OID SYMBOL_RECORD)
    (ISA_SYMBOL ?OID))
(<= (ISA_SYMBOL ?OID)
    (REF_IC_SY ?IA ?OID))
(<= (ISA_SYMBOL ?OID)
    (REF_TC_SY ?TC ?OID))
(<= (ISA_SYMBOL ?OID)
    (REF_XTC_SY ?XTC ?OID))
;;;;;Define mapping from IPS Symbol as a string
;;;;;to the symbolic object ID of Symbol_Record
(<= (SY-SY_ID ?ID ?X)
    (STRAPPEND "SYMBOL_" ?ID ?XS) (EXPRESSION ?XS ?X))
;;;;;Symbol_Record: Binary Relations
;;;;;Mappings to IPS, TESS, XTAS, and CTL
(<= (SY_IPS_CLIENT ?SY ?IC)
    (REF_IC_SY ?IC ?SY))
(<= (SY_TESS_CLIENT ?SY ?TC)
    (REF_TC_SY ?TC ?SY))
(<= (SY_XTAS_CLIENT ?SY ?XTC)
    (REF_XTC_SY ?XTC ?SY))
(<= (SY_CTL ?SY ?CT)
    (REF_TC_SY ?TC ?SY)
    (REF_CTL_TESS ?CT ?TC))
;;;;;Filling in IPS Symbol from whichever system has it
;;;;;Redundant results are ignored
(<= (SY_SYMBOL ?ID ?SY)
    (IC_CUST_SYMBOL ?IC ?SY)
    (SY-SY_ID ?SY ?ID))
(<= (SY_SYMBOL ?ID ?SY)
    (TC_IPS_CLIENT_SY ?TC ?SY)
    (SY-SY_ID ?SY ?ID))
(<= (SY_SYMBOL ?ID ?SY)
    (XTC_IPS_CLIENT_SY ?XTC ?SY)
    (SY-SY_ID ?SY ?ID))
```

There are currently 1160 rules and 287 relations defined. It takes almost

3 hours minutes to load the data and materialize 221 relations.

5 Future Work

There are several steps ahead for Stanford's work on this project. There are additional commercial customer systems to integrate. For the first four systems integrated, Stanford's work followed that of Merrill Lynch. It will be interesting for Stanford to integrate in a commercial customer system before Merrill Lynch does. Measurements of performance and evaluations of scalability need to be performed on Infomaster as the number of commercial customer systems integrated grows. These factors apply to both the load/materialize time and the query/browse time. Additional design work must be done on how this project presents its data through Infomaster. The interface must be simplified in order to be usable by the account managers' assistants and clarified in order to be usable by the commercial customer system maintainers.

Infomaster has an update capability. However, materializing data interferes with this update capability, as the materialized relations do not change automatically when their source data changes. Merrill Lynch is interested in experimenting with this update functionality through Infomaster. The CRU system has a nightly batch update incorporate the incremental changes that have occurred to each of the commercial customer systems. A subsequent experiment is to support online update feed from the commercial customer systems that supports near real time maintenance of the integrated data.

There are a variety of improvements that can be made to Infomaster. Increased customizability of the user interface could increase the usability of the system. Support for more complex class structures would allow Infomaster to replicate more closely the displays of the CRU system. A Java-based interface could support dynamic change of the query screens in response to certain specifications. Automatic maintenance of materialized relations could better support update processing.

6 Acknowledgements

We thank Rada Chirkova, Winton Davies, Elham Ghassemzadeh, and Rob Hoskins of Stanford for their assistance and advice. We also thank Erna Adelson, Ray Bonker, and others of Merrill Lynch for providing the data and timely feedback on our efforts. This work was supported in large part by

Merrill Lynch. We thank Epistemics Inc. for providing the latest commercial version of Infomaster for our work.

7 Conclusion

Stanford has integrated the client and accounts data of four commercial customer systems of Merrill Lynch as a shadow project of Merrill Lynch's own efforts in this area. These data integrated includes several hundred thousand records and over 70 megabytes of data as a raw text data feed. Stanford added the fourth commercial customer system, XTAS, to the integration of the first three over a two-week period that included modeling the data, modifying existing views and creating additional views of the data, implementing additional diagnostics, and determining which relations to materialize to ensure rapid query response.

The collaboration between Stanford and Merrill Lynch has several key benefits. Stanford gets experience wth the applications of technology developed in the laboratory, and this feeds into additional research problems. Merrill Lynch gets early detailed exposure to new technology that can impact real business problems. A key goal is to provide lessons learned to Merrill Lynch, Stanford, and to the community at large, and to facilitiate the flow of technology from the laboratory as research ideas to its widespread adoption in the marketplace.

References

[KelGen97] Arthur M. Keller and Michael R. Genesereth "Using Infomaster to Create a Housewares Virtual Catalog,"; *Int. Journal of Electronic Markets*, University of St. Gallen, Switzerland, Vol. 7, No. 4, 1997, p. 41–45.

[KelGen96] Arthur M. Keller and Michael R. Genesereth, "Multivendor Catalogs: Smart Catalogs and Virtual Catalogs,"; *EDI Forum, The Journal of Electronic Commerce*, Vol. 9, No. 3, September 1996.

[GenKelDus97a] Michael R. Genesereth, Arthur M. Keller and Oliver M. Duschka, "Infomaster: An Information Integration System,"; *ACM SIGMOD*, May 1997.

[GedGenKelSin95] Donald F. Geddis, Michael R. Genesereth, Arthur M. Keller and Narinder P. Singh, "Infomaster: A Virtual Information System,"; *Intelligent Information Agents Workshop*, at CIKM '95, December 1995.

[Kel95a] Arthur M. Keller, "Smart Catalogs and Virtual Catalogs," (invited talk) in *An International Conference on Frontiers of Electronic Commerce, Sixth Organizational Computing, Coordination, and Collaboration Conference*, Austin, TX, October 1995.

[Kel95b] Arthur M. Keller, "Smart Catalogs and Virtual Catalogs,"; *Usenix Workshop on Electronic Commerce*, July 1995.

[KelGenSinSye94] Arthur M. Keller, Michael R. Genesereth, Narinder P. Singh and Mustafa A. Syed, "A Smart Catalog and Brokering Architecture for Electronic Commerce,"; *Workshop on Electronic Commerce*, December 1994.

[Dus97] Oliver M. Duschka, "Query Planning and Optimization in Information Integration,"; *Ph.D. Thesis* , Stanford University, Computer Science Technical Report STAN-CS-TR-97-1598, 1997.

[DusGen97a] Oliver M. Duschka and Michael R. Genesereth, "Query Planning in Infomaster,"; *Proceedings of the Twelfth Annual ACM Symposium on Applied Computing*, San Jose, February 1997.

[DusGen97b] Oliver M. Duschka and Michael R. Genesereth, "Answering Recursive Queries Using Views,"; *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS-97, Tucson, May 1997.

[GenKelDus97b] Michael R. Genesereth, Arthur M. Keller, and Oliver M. Duschka, "Infomaster: An Information Integration System,"; *Proceedings of the SIGMOD International Conference on Management of Data*, SIGMOD-97, Tucson, May 1997.

[Dus97b] Oliver M. Duschka, "Query Optimization Using Local Completeness,"; *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, AAAI-97, Providence, Rhode Island, July 1997.

[DusGen97c] Oliver M. Duschka and Michael R. Genesereth, "Infomaster - An Information Integration Tool,"; *Proceedings of the International Workshop "Intelligent Information Integration"*, during the 21st German Annual Conference on Artificial Intelligence, KI-97, Freiburg, Germany, September 1997.

[AbiDus98] Serge Abiteboul and Oliver M. Duschka, "Complexity of Answering Queries Using Materialized Views,"; *Proceedings of the Seventeenth SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS-98, Seattle, WA, June 1998.

[DusGen98] Oliver M. Duschka and Michael R. Genesereth, "Query Planning with Disjunctive Sources,"; *Proceedings of the Workshop on AI and Information Integration*, Madison, Wisconsin, July 1998.