Network Working Group
Request for Comments: 2258
Category: Informational

J. Ordille
Bell Labs, Lucent Technologies

January 1998

Internet Nomenclator Project

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

The goal of the Internet Nomenclator Project is to integrate the hundreds of publicly available CCSO servers from around the world. Each CCSO server has a database schema that is tailored to the needs of the organization that owns it. The project is integrating the different database schema into one query service. The Internet Nomenclator Project will provide fast cross-server searches for locating people on the Internet. It augments existing CCSO services by supplying schema integration, more extensive indexing, and two kinds of caching -- all this in a system that scales as the number of CCSO servers grows. One of the best things about the system is that administrators can incorporate their CCSO servers into Nomenclator without changing the servers. All Nomenclator needs is basic information about the server.

This document provides an overview of the Nomenclator system, describes how to register a CCSO server in the Internet Nomenclator Project, and how to use the Nomenclator search engine to find people on the Internet.

1. Introduction

Hundreds of organizations provide directory information through the CCSO name service protocol [3]. Although the organizations provide a wealth of information about people, finding any one person can be difficult because each organization's server is independent. The different servers have different database schemas (attribute names and data formats). The 300+ CCSO servers have more than 900 different attributes to describe information about people. Very few common attributes exist. Only name and email occur in more than 90% of the servers [4]. No special support exists for cross-server searches, so searching can be slow and expensive.

The goal of the Internet Nomenclator Project is to provide fast, integrated access to the information in the CCSO servers. The project is the first large-scale use of the Nomenclator system. Nomenclator is a more general system than a white pages directory service. It is a scalable, extensible information system for the Internet.

Nomenclator answers descriptive (i.e. relational) queries. Users can locate information about people, organizations, hosts, services, publications, and other objects by describing their attributes. Nomenclator achieves fast descriptive query processing through an active catalog, and extensive meta-data and data caching. The active catalog constrains the search space for a query by returning a list of data repositories where the answer to the query is likely to be found. Meta-data and data caching keep frequently used query processing resources close to the user, thus reducing communication and processing costs.

Through the Internet Nomenclator Project, users can query any CCSO server, regardless of its attribute names or data formats, by specifying the query to Nomenclator (see Figure 1). Nomenclator provides a world view of the data in the different servers. Users express their queries in this world view. Nomenclator returns the answer immediately if it has been cached by a previous query. If not, Nomenclator uses its active catalog to constrain the query to the subset of relevant CCSO servers. The speed of the query is increased, because only relevant servers are contacted. Nomenclator translates the global query into local queries for each relevant CCSO server. It then translates the responses into the format of the world view.

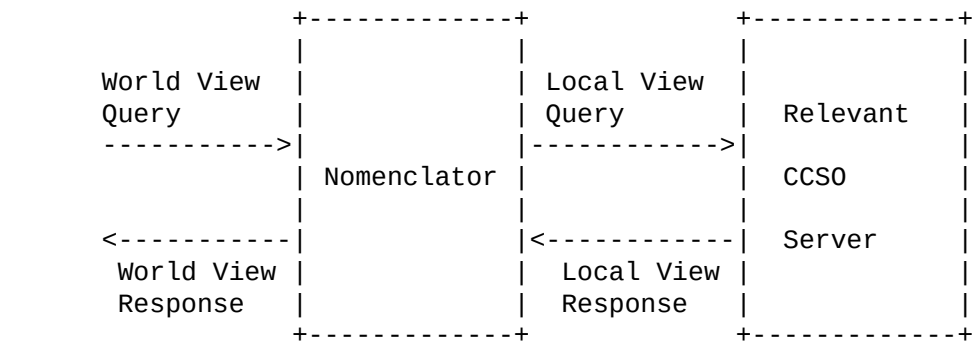

Figure 1: A Nomenclator Query

Nomenclator translates queries to and from
the language of the relevant CCSO servers.

The Internet Nomenclator Project makes it easier for users to find a particular CCSO server, but it does not send all queries to that server. When Nomenclator constrains the search for a query answer, it screens out irrelevant queries from ever reaching the server. When Nomenclator finds an answer in its cache, it screens out redundant queries from reaching the server. The server becomes easier to find and use without experiencing the high loads caused by exhaustive and redundant searches.

The Internet Nomenclator Project creates the foundation for a much broader heterogeneous directory service for the Internet. The current version of Nomenclator provides integrated access to CCSO and relational database services. The Nomenclator System Architecture supports fast, integrated searches of any collection of heterogeneous directories. The Internet Nomenclator Project can be enhanced to support additional name services, or provide intergated query services for other application domains. The project is starting with CCSO services, because the CCSO services are widely available and successful.

Section 2 describes the Nomenclator system in more detail. Section 3 explains how to register a CCSO server as part of the project. Section 4 briefly describes how to use Nomenclator. Section 5 provides a summary.

2. Nomenclator System

Nomenclator is a scalable, extensible information system for the Internet. It supports descriptive (i.e. relational) queries. Users locate information about people, organizations, hosts, services, publications, and other objects by describing their attributes. Nomenclator achieves fast descriptive query processing through an active catalog, and extensive meta-data and data caching.

The active catalog constrains the search space for a query by returning a list of data repositories where the answer to the query is likely to be found. Components of the catalog are distributed indices that isolate queries to parts of the network, and smart algorithms for limiting the search space by using semantic, syntactic, or structural constraints. Meta-data caching improves performance by keeping frequently used characterizations of the search space close to the user, thus reducing active catalog communication and processing costs. When searching for query responses, these techniques improve query performance by contacting only the data repositories likely to have actual responses, resulting in acceptable search times.

Administrators make their data available in Nomenclator by supplying information about the location, format, contents, and protocols of their data repositories. Experience with Nomenclator shows that gathering a small amount of information from data owners can have a substantial positive impact on the ability of users to retrieve information. For example, each CCSO administrator provides a mapping from the local view of data (i.e. the local schema) at the CCSO server to Nomenclator’s world view. The administrator also supplies possible values for any attributes with small domains at the data repository (such as the "city" or "state_or_province" attributes). With this information, Nomenclator can isolate queries to a small percentage of the CCSO data repositories, and provide an integrated view of their data. Nomenclator provides tools that minimize the effort that administrators expend in characterizing their data repositories. Nomenclator does not require administrators to change the format of their data or the access protocol for their database.

2.1 Components of a Nomenclator System

A Nomenclator system is comprised of a distributed catalog service and a query resolver (see Figure 2). The distributed catalog service gathers meta-data about data repositories and makes it available to the query resolver. Meta-data includes constraints on attribute

values at a data repository, known patterns of data distribution across several data repositories, search and navigation techniques, schema and protocol translation techniques, and the differing schema at data repositories.

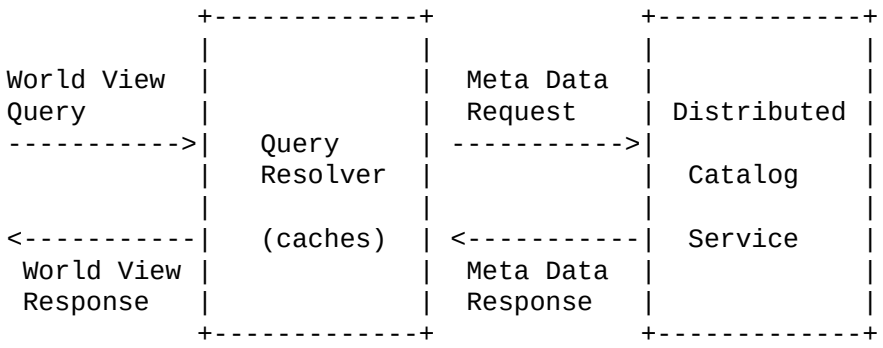

Figure 2: Components of a Nomenclator System

Query resolvers at the user sites retrieve, use, cache, and re-use this meta-data in answering user queries. The catalog is "active" in two ways. First, some meta-data moves from the distributed catalog service to each query resolver during query processing. Second, the query resolver uses the initial meta-data, in particular the search and navigation techniques, to generate additional meta-data that guides query processing. Typically, one resolver process serves a few hundred users in an organization, so users can benefit from larger resolver caches.

Query resolvers cache techniques for constraining the search space and the results of previously constrained searches (meta-data), and past query answers (data) to speed future query processing. Meta-data and data caching tailor the query resolver to the specific needs of the users at the query site. They also increase the scale of a Nomenclator system by reducing the load from repeated searches or queries on the distributed catalog service, data repositories, and communications network.

The distributed catalog service is logically one network service, but it can be divided into pieces that are distributed and/or replicated. Query resolvers access this distributed, replicated service using the same techniques that work for multiple data repositories.

A Nomenclator system naturally includes many query resolvers. Resolvers are independent, but renewable, query agents that can be as powerful as the resources available at the user site. Caching decreases the dependence of the resolver on the distributed catalog service for frequently used meta-data, and on data repositories for frequently used data. Caching thus improves the number of users that can be supported and the local availability of the query service.

2.2 Meta-Data Techniques

The active catalog structures the information space into a collection of relations about people, hosts, organizations, services and other objects. It collects meta-data for each relation and structures it into "access functions" for locating and retrieving data. Access functions respond to the question: "Where is data to answer this query?" There are two types of responses corresponding to the two types of access functions. The first type of response is: "Look over there." "Catalog functions" return this response; they constrain the query search by limiting the data repositories contacted to those having data relevant to the query. Catalog functions return a referral to data access functions that will answer the query or to additional catalog functions to contact for more detailed information. The second response to "Where?" is: "Here it is!" "Data access functions" return this response; they understand how to obtain query answers from specific data repositories. They return tuples that answer the query. Nomenclator supplies access functions for common name services, such as the CCSO service, and organizations can write and supply access functions for data in their repositories.

Access functions are implemented as remote or local services. Remote access functions are services that are available through a standard remote procedure call interface. Local access functions are functions that are supplied with the query resolver. Local access functions can be applied to a variety of indexing and data retrieval tasks by loading them with meta-data stored in distributed catalog service. Remote access functions are preferred over local ones when the resources of the query resolver are inadequate to support the access function. The owners of data may also choose to supply remote access functions for privacy reasons if their access functions use proprietary information or algorithms. Local functions are preferred whenever possible, because they are highly replicated in resolver caches. They can reduce system and network load by bringing the resources of the active catalog directly to the users.

Remote access functions are simple to add to Nomenclator and local access functions are simple to apply to new data repositories, because the active catalog provides "referrals" that describe the conditions for using access functions. For simplicity, this document describes referral techniques for exact matching of query strings. Extensions to these techniques in Nomenclator support matching query strings that contain wildcards or word-based matching of query strings in the style of the CCSO services.

Each referral contains a template and a list of references to access functions. The template is a conjunctive selection predicate that describes the scope of the access functions. Conjunctive queries that are within the scope of the template can be answered with the referral. When a template contains a wildcard value ("*") for an attribute, the attribute must be present in any queries that are processed by the referral. The system follows the following rule:

> Query Coverage Rule:
>
> If the set of tuples satisfying the selection predicate in a query is covered by (is a subset of) the set of tuples satisfying the template, then the query can be answered by the access functions in the reference list of the referral.

For example, the query below:

```
select * from People where country = "US" and surname = "Ordille";
```

is covered by the following templates in Lines (1) through (3), but not by the templates in Lines (4) and (5):

(1) country = "US" and surname = "*"

(2) country = "US" and surname = "Ordille"

(3) country = "US"

(4) organization = "*"

(5) country = "US" and surname = "Elliott"

Referrals form a generalization/specialization graph for a relation called a "referral graph." Referral graphs are a conceptual tool that guides the integration of different catalog functions into our system and that supplies a basis for catalog function construction and query processing. A "referral graph" is a partial ordering of

the referrals for a relation. It is constructed using the subset/superset relationship: "S is a subset of G." A referral S is a subset of referral G if the set of queries covered by the template of S is a subset of the set of queries covered by the template of G. S is considered a more specific referral than G; G is considered a more general referral than S. For example, the subset relationship exists between the pairs of referrals with the templates listed below:

(1) country = "US" and surname = "Ordille"
    is a subset of
    country = "US"

(2) country = "US" and surname = "Ordille"
    is a subset of
    country = "US" and surname = "*"

(3) country = "US" and surname = "*"
    is a subset of
    country ="US"

(4) country = "US"
    is a subset
    "empty template"

but it does not exist between the pairs of referrals with the following templates:

(5) country = "US"
    is not a subset of
    department = "CS"

(6) country = "US" and name = "Ordille"
    is not a subset of
    country = "US" and name = "Elliott"

In Lines (1) and (2), the more general referral covers more queries, because it covers queries that list different values for surname. In Line (3), the more general referral covers more queries, because it covers queries that do not constrain surname to a value. In Line (4), the specific referral covers only those queries that constrain the country to "US" while the empty template covers all queries.

During query processing, wildcards in a template are replaced with the value of the corresponding attribute in the query. For any query covered by two referrals S and G such that S is a subset of G, the set of tuples satisfying the template in S is covered by the set of

tuples satisfying the template in G. S is used to process the query, because it provides the more constrained (and faster) search space. The referral S has a more constrained logical search space than G, because the set of tuples in the scope of S is no larger, and often smaller, than the set in the scope of G. Moreover, S has a more constrained physical search space than G, because the data repositories that must contacted for answers to S must also be contacted for answers to G, but additional data repositories may need to be contacted to answer G.

In constraining a query, a catalog function always produces a referral that is more specific than the referral containing the catalog function. Wildcards ("*") in a template indicate which attribute values are used by the associated catalog function to generate a more specific referral. In other words, catalog functions always follow the rule:

> Catalog Function Constrained Search Rule:
>
> Given a referral R with a template t and a catalog function cf, and a query q covered by t, the result of using cf to process q, cf(q), is a referral R' with template t' such that q is covered by t' and R' is more specific than R.

Catalog functions make it possible to import a portion of the indices for the information space into the query resolver. Since they generate referrals, the resolver can cache the most useful referrals for a relation and call the catalog function as needed to generate new referrals.

The resolver query processing algorithm obtains an initial set of referrals from the distributed catalog service. It then navigates the referral graph, calling catalog functions as necessary to obtain additional referrals that narrow the search space. Sometimes, two referrals that cover the query have the relationship of general to specific to each other. The resolver eliminates unnecessary access function processing by using only the most specific referral along each path of the referral graph.

The search space for the query is initially set to all the data repositories in the relation. As the resolver obtains referrals to sets of relevant data repositories (and their associated data access functions) it forms the intersection of the referrals to constrain the search space further. The intersection of the referrals includes only those data repositories listed in all the referrals. Intersection combines independent paths through the referral graph to derive benefit from indices on different attributes.

2.3 Meta-Data and Data Caching

A Nomenclator query resolver caches the meta-data that result from calling catalog functions. It also caches the responses for queries. If the predicate of a new query is covered by the predicate of a previous query, Nomenclator calculates the response for the new query from the cached response of the old query. Nomenclator timestamps its cache entries to provide measures of the currentness of query responses and selective cache refresh. The timestamps are used to calculate a t-bound on query responses [5][1]. A t-bound is the time after which changes may have occurred to the data that are not reflected in the query response. It is the time of the oldest cache entry used to calculate the response. Nomenclator returns a t-bound with each query response. Users can request more current data by asking for responses that are more recent than this t-bound. Making such a request flushes older items from the cache if more recent items are available. Query resolvers calculate a minimum t-bound that is some refresh interval earlier than the current time. Resolvers keep themselves current by replacing items in the cache that are earlier than the minimum t-bound.

2.4 Scale and Performance

Three performance studies of active catalog and meta-data caching techniques are available [5]. The first study shows that the active catalog and meta-data caching can constrain the search effectively in a real environment, the X.500 name space. The second study examined the performance of an active catalog and meta-data caching for single users on a local area network. The experiments showed that the techniques to eliminate data repositories from the search space can dramatically improve response time. Response times improve, because latency is reduced. The reduction of latency in communications and processing is critical to large-scale descriptive query optimization. The experiments also showed that an active catalog is the most significant contributor to better response time in a system with low load, and that meta-data caching functions to reduce the load on the system. The third study used an analytical model to evaluate the performance and scaling of these techniques for a large Internet environment. It showed that meta-data caching plays an essential role in scaling the distributed catalog service to millions of users. It also showed that constraining the search space with an active catalog contributes significantly to scaling data repositories to millions of users. Replication and data caching also contribute to the scale of the system in a large Internet environment.

3. Registering a CCSO Server

The Internet Nomenclator Project supports the following home page:

http://cm.bell-labs.com/cs/what/nomenclator

The home page provides a variety of information and services.

Administrators can register their CCSO servers through services on this home page. The registration service collects CCSO server location information, contact information for the administrator of the CCSO server, implicit and explicit constraints on entries in the server's database, and a mapping from the local schema of the CCSO server to the schema of the world view.

The implicit and explicit constraints on the server's database are the fuel for Nomenclator's catalog functions. The registration center currently collects constraints on organization name, department, city, state or province name, country, phone number, postal code, and email address. These constraints are automatically incorporated into Nomenclator's distributed catalog service. They are used by catalog functions in query resolvers to constrain searches to relevant CCSO servers. For example, a database only contains information about the computer science and electrical engineering departments at a French university. The department, organization and country attributes are constrained. Nomenclator uses these constraints to prevent queries about other departments, organizations or countries from being sent to this CCSO server.

The mapping from the local schema of the CCSO server to the schema of the world view allows Nomenclator to translate queries and responses for the CCSO server. The registration center currently collects this mapping by requesting an example of how to translate a typical entry in the CCSO server into the world view schema and, optionally, an example of how to translate a canonical entry in the world view schema into the local schema of the CCSO server [4]. These examples are then used to generate a mapping program that is stored in the distributed catalog service. The CCSO data access function in the query resolver interprets these programs to translate queries and responses communicated with that CCSO server. We plan to release the mapping language to CCSO server administrators, so administrators can write and maintain the mapping for their servers. We have experimented with more than 20 mapping programs. They are seldom more than 50 lines, and are often shorter. It typically takes one or two lines to map an attribute.

4. Using Nomenclator

The Internet Nomenclator Project currently provides a centralized query service on the Internet. The project runs a Nomenclator query resolver that is accessible through its Web page (see the URL in Section 3) and the Simple Nomenclator Query Protocol (SNQP) [2].

The service answers queries that are a conjunction of string values for attributes. A variety of matching techniques are supported including exact string matching, matching with wildcards, and word-based matching in the style of the CCSO service. Our web interface uses the Simple Nomenclator Query Protocol (SNQP) [2]. Programmers can create their own interfaces by using this protocol to communicate with the Nomenclator query resolver. They will require the host name and port number for the query resolver which they can obtain from the Nomenclator home page. SNQP, and hence the web interface, are defined for US-ASCII. Support for other character sets will require further work.

Subsequent phases of the project will provide enhanced services such as providing advice about the cost of queries and ways to constrain queries further to produce faster response times, and allowing users to request more current data. We also plan to distribute query resolvers, so users can benefit from running query resolvers locally. Local query resolvers reduce latency for the user, and distribute query processing load throughout the network.

5. Summary

The Internet Nomenclator Project augments existing CCSO services by supplying schema integration and fast cross-server searches. The key to speed in descriptive query processing is an active catalog, and extensive meta-data and data caching. The Nomenclator system is the result of research in distributed systems [5][6][7][4]. It can be extended to incorporate other name servers, besides the CCSO servers, and to address distributed search and retrieval challenges in other application domains. In addition to providing a white pages service, the Internet Nomenclator Project will evaluate how an active catalog, meta-data caching and data caching perform in very large global information system. The ultimate goal of the project is to refine these techniques to provide the best possible global information systems.

6. Security Considerations

In the Internet Nomenclator Project, the participants' data are openly available and read-only. Since the risk of tampering with queries and responses is considered low, this version of Nomenclator does not define procedures for protecting the information in its queries and responses.

7. References

[1] H. Garcia-Molina, G. Wiederhold. "Read-Only Transactions in a Distributed Database," ACM Transactions on Database Systems 7(2), pp. 209-234. June 1982.

[2] Elliott, J., and J. Ordille, "The Simple Nomenclator Query Protocol (SNQP)," RFC 2259, January 1998.

[3] S. Dorner, P. Pomes. "The CCSO Nameserver: A Description," Computer and Communications Services Office Technical Report, University of Illinois, Urbana, USA. 1992. Avaialble in the current "qi" distribution from <URL:ftp://uiarchive.cso.uiuc.edu/local/packages/ph>

[4] A. Levy, J. Ordille. "An Experiment in Integrating Internet Information Sources," AAAI Fall Symposium on AI Applications in Knowledge Navigation and Retrieval, November 1995. <URL:http://cm.bell-labs.com/cm/cs/doc/95/11-01.ps.gz>

[5] J. Ordille. "Descriptive Name Services for Large Internets," Ph. D. Dissertation. University of Wisconsin. 1993. <URL:http://cm.bell-labs.com/cm/cs/doc/93/12-01.ps.gz>

[6] J. Ordille, B. Miller. "Distributed Active Catalogs and Meta-Data Caching in Descriptive Name Services," Thirteenth International IEEE Conference on Distributed Computing Systems, pp. 120-129. May 1993. <URL:http://cm.bell-labs.com/cm/cs/doc/93/5-01.ps.gz>

[7] J. Ordille, B. Miller. "Nomenclator Descriptive Query Optimization in Large X.500 Environments," ACM SIGCOMM Symposium on Communications Architectures and Protocols, pp. 185-196, September 1991. <URL:http://cm.bell-labs.com/cm/cs/doc/91/9-01.ps.gz>

8. Author's Address

Joann J. Ordille
Bell Labs, Lucent Technologies
Computing Sciences Research Center
700 Mountain Avenue, Rm 2C-301
Murray Hill, NJ 07974 USA

EMail: joann@bell-labs.com

9. Full Copyright Statement