Network Documentation: A Web-based Relational Database Approach

Wade Warner
Georgia State university
wade@cs.gsu.edu
Rajshekhar Sunderraman
Georgia State University
raj@cs.gsu.edu

Abstract

Every organization managing a network of computers has a need to organize, maintain, and access information related to the network. Users at various levels of the organization need quick and convenient access to this critical information at all times. We propose a methodology which is Web-based and which uses relational databases at the back end to store and organize this information. We envision that this Web-based system will be used in an intranet environment and will provide several levels of access to different user types.

1 Introduction

In today's complex networks, it is more important than ever that network documentation be clear, concise, and up to date. Simple tasks such as looking up the serial number of a workstation before calling tech support are often repeated endlessly because the required information is not recorded anywhere. Technicians unfamiliar with the setup of a particular workstation must first take the time to analyze the configuration of the station before any useful work can be done. Considerable time could be saved if all of the pertinent details are readily available.

Paper records are one possible solution, but there are many problems associated with them. Keeping paper records up to date is time consuming at best, and it can be difficult to search for the needed data. A better solution is to keep the documentation on the network itself.

We propose a methodology which is based on relational database technology. The information about the network is stored in a relational database (Oracle8) and is accessed via a Web browser in an intranet environment. Easy to use Web forms are used to access and update the data.

The solution presented in this paper is specific to our organization, but the system can be easily modified for any environment due to our use of Query By Example (QBE) technology for dynamic query generation. For organizations that do not have Oracle, we propose to also implement our system using a freely available database engine, mSQL.

Both, the Oracle and the mSQL implementation will be available freely to interested users.

2 Database Design

The database was first designed using the Entity-Relationship modeling technique and then converted into relational schema in SQL. The ER schema is shown in Figure 1.

The following three entity types are identified for the database:

- **Computers**: This entity type consists of all the PCs, workstations, and servers in the organization. The hostname attribute is chosen as the primary key. Three unique keys have also been identified. These are: gsu_sn the serial number assigned by the organization, ip_address, and mac_address. In addition to these attributes, several other attributes have been identified and the detailed description is shown later in the paper.

- **Software**: This entity set consists of all the software packages and programs available on the network. The title and version attributes form the primary key for this entity set. An interesting attribute identified for this entity set is the doc_url attribute which contains the URL for the documentation related to the software.
- **Printers**: This entity set consists of all the printer devices available on the network. The gsu_sn organizational serial number has been identified as the primary key for this entity set. Some of the more interesting attributes of this entity set are the model, the model number, queue_name, the queue name assigned to the printer, location, the physical location of the printer, and availability, whether the printer is available locally or on the network.

Two relationships between these entity sets have been identified. These are:

- **Installed_sw**: This is a many-to-many relationship between computers and software that indicates which software is installed (or available) on which computer. This relationship type has an attribute: availability, which indicates if the software is locally installed or installed on the network.
- **Installed_pr**: This is a many-to-many relationship between computers and printers that indicates which printer is accessible to which computer in the network.

Based on these entity and relationship types, the following SQL definition of the database tables has been developed for our organization. The relational tables that correspond to these entity and relationship sets are created in the Oracle 8 database.

```
drop table computers cascade constraints;
create table computers (
  hostname      varchar2(20) not null,
  domain        varchar2(10),
  manufacturer  varchar2(20),
  model         varchar2(20),
  architecture  varchar2(20),
  memory_size   number(5),
  hdd_size      number(7,2),
  sound_card    varchar2(20),
  video_card    varchar2(20),
  os            varchar2(20),
  vendor_sn     varchar2(50) not null,
  gsu_sn        number(6) not null,
  location      varchar2(20),
  ip_address    varchar2(15) not null,
  mac_address   varchar2(30),
  primary key (hostname),
  unique (gsu_sn),
  unique (ip_address),
  unique (mac_address)
);

drop table software cascade constraints;
create table software (
  title     varchar2(20)  not null,
  version   varchar2(10)  not null,
  category  varchar2(20),
  doc_url   varchar2(100),
  primary key (title,version)
);

drop table installed_sw cascade constraints;
create table installed_sw (
  hostname      varchar2(20) not null,
  title         varchar2(20) not null,
  version       varchar2(10) not null,
  availability  varchar2(10)
                check(availability in
                  ('local','network')),
  primary key (hostname,title,version),
  foreign key (hostname) references computers,
  foreign key (title,version) references software
);

drop table printers cascade constraints;
create table printers (
  manufacturer  varchar2(20),
  model         varchar2(20),
  queue_name    varchar2(20),
```

```
  location      varchar2(20),
  vendor_sn     varchar2(50) not null,
  gsu_sn        number(6)    not null,
  availability  varchar2(10)
                check(availability in
                  ('local','network')),
  color         varchar2(10)
                check(color in
                  ('color','greyscale')),
  type          varchar2(20),
  address       varchar2(30),
  primary key(gsu_sn)
);

drop table installed_pr cascade constraints;
create table installed_pr (
  hostname      varchar2(20) not null,
  gsu_sn        number(6)    not null,
  primary key (hostname,gsu_sn),
  foreign key (hostname) references computers,
  foreign key (gsu_sn) references printers
);
```

The next important step in the process of making this database available is to populate it with real data. For an existing network, this process may involve extracting data from existing electronic files or spreadsheets using scripts written in Perl for example, or manually entering new data.

3 Web Application Development

In this section, we sketch the Web application that will access the Oracle database defined earlier. Our Web application provides for three different levels of access. At the highest level of access, users such as the network administrator will have access to querying pages, database update pages, and report generation pages. At the next level of access, users such as managers will have access to querying and report generation pages. At the lowest level of access, casual users will have access only to limited querying pages.

Every user is assigned an userid and password to access the system. They are also assigned one of three access levels: *Privileged*, *Normal*, and *Casual*. To keep track of the user information, we have designed a table with the following schema:

```
create table users (
  name        varchar2(25) not null,
  username    varchar2(10) not null,
  password    varchar2(15) not null,
  usertype    varchar2(10) not null
              check(username in('Privileged',
                      'Normal','Casual')),
  primary key (username)
);
```

Depending on the user access level, a different set of options will be available in response to a successful login.

We now present some querying options that are available at all levels of access. Figure 2 shows three queries that the users can pose: query `computers` on host name, query `printers` on printer type, and query `software` on title.

Sample results from the query on printer types is shown in Figure 3.

4 Generalizing the Methodology

The methodology and system presented in this paper has several features that make it easy to use it in any organization.

The querying aspect of the system is divided into two categories: static or canned queries as described in the previous section and dynamic or on the fly queries.

The static queries are the ones that would apply to any organization. Typically these queries are related to getting details based on serial numbers of computers or titles of software or categories of printers etc. Almost every organization would see a need for such queries and our system provides easy interfaces to answer these queries as shown in the previous section.

In addition to such fixed queries, our system provides an interface which will allow dynamic queries to be generated and executed by the user. A visual querying interface on the Web implementing the

Query By Example (QBE) query language is used to allow the users to pose arbitrary queries. Another Web interface for dynamic queries is based on SQL. Users who have a knowledge of SQL can easily formulate dynamic queries using this interface and have their results displayed on the Web browser. Both these dynamic querying interfaces have already been built in [6] and is easily added to the system described in this paper.

Even though the methodology presented in this paper is specific to our organization, we believe that adapting it to any other organization should be an easy task. We have tried to make the database design general enough so that many organizations can use the database tables as is. Building new functionality is then a matter of writing new code. If the database design has to be altered, some of the functions have to be redesigned and some new ones may have to be written. The dynamic querying aspects of the system is easily portable to any other organization.

The current implementation uses Oracle 8.0 database server and the Oracle Application Server 4.0 system. Any organization using Oracle can easily port our system with minimal changes. We propose to implement the system using mSQL, a publicly available popular database system. This implementation can be used by almost any organization.

5 Conclusions

We have presented a framework for documenting a local area network. A relational database (Oracle 8) is used at the back end to store the relevant information. Applications that run on the Web are developed to provide ready access to this information at various levels of security levels. Users are able to query the information using static or canned queries as well as dynamic or on the fly queries. Other users are able to print statistical information and reports from the database. Privileged users have the capability of changing the information using easy to use Web forms.

These applications are being developed using Oracle Application Server 4.0. We propose to provide an implementation using mSQL, a freely available database engine, so that organizations who do not have Oracle database server can use the mSQL system. Finally, the solution presented in this paper is easily adaptable to any organization with little effort.

6 Acknowledgments

We acknowledge the implementation effort of our graduate student, Shahnaz Ahmed.

References

[1] B.D. Brown, R.J. Niemiec, and J.C. Trezzo. Oracle Application Server Web Toolkit Reference. Osborne McGraw-Hill, 1998.

[2] R. El-Masri and S. Navathe. Fundamentals of Database Systems. Addison-Wesley Longman, 1994.

[3] B. Johnson. Oracle Web Application Server Handbook. Osborne McGraw-Hill, 1997.

[4] G. Koch and K. Loney. Oracle 8: The Complete Reference. Osborne McGraw-Hill, 1997.

[5] R. Sunderraman Oracle Programming: A Primer. Addison-Wesley Longman, 1999.

[6] R. Sunderraman ReqWeb: Relational Querying of Web Data. Unpublished manuscript, Georgia State University.

[7] L. Wall, T. Christiansen, and R.L. Schwarz. Programming Perl. O'Reilly and Associates, 1996.

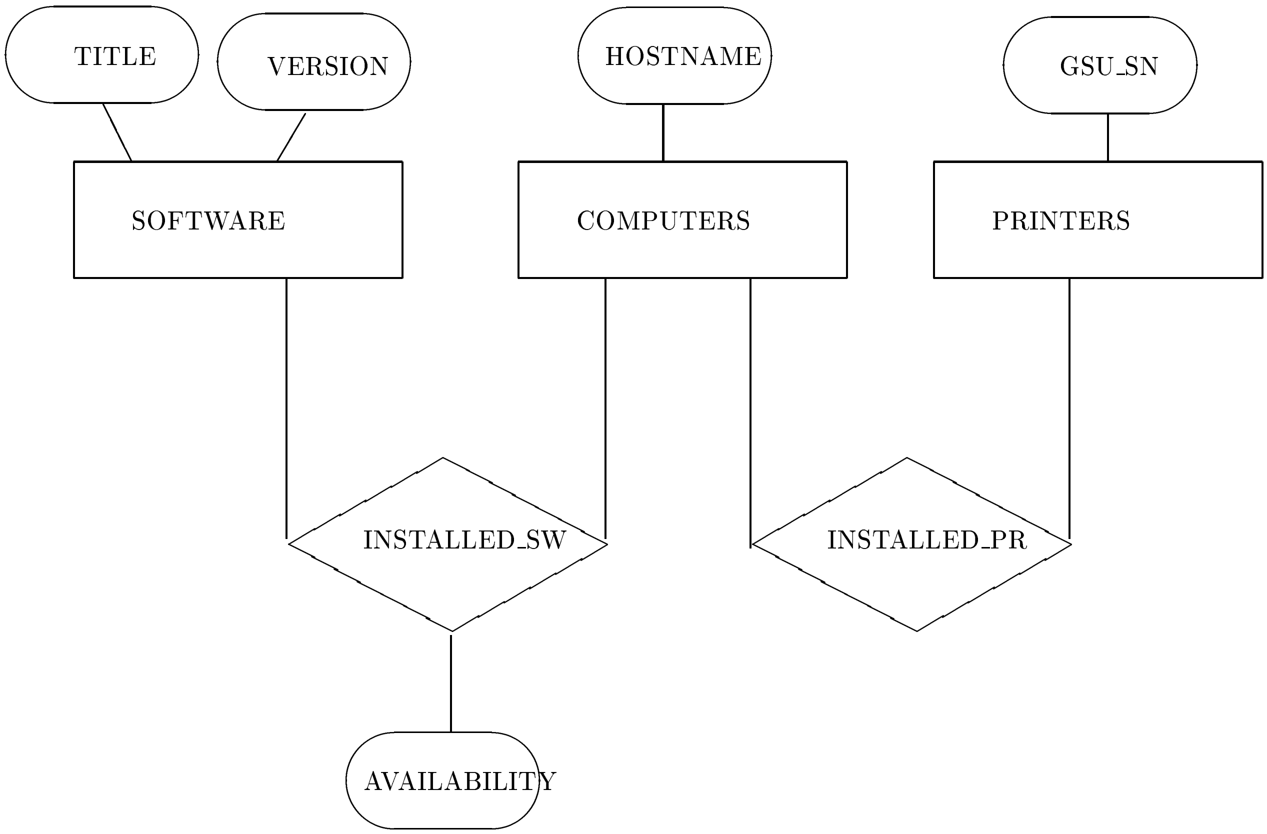

Figure 1: ER Diagram for Network Documentation Database

NETWORK ADMINISTRATION SYSTEM

GEORGIA STATE UNIVERSITY

Computer Details

Select Host Name: Hewlett Get Details

Printer Details By type

Select Printer Type: bubblejet Get Details

Software Details By title

Select Software Title: Microsoft Access Get Details

Figure 2: Query Menu

The following laser printers are available

Manufacturer	Model	Queue Name	Location	Color
Canon	C4000	cn1	745 COE	color
Xerox	X2000	xp12	744 COE	greyscale

Figure 3: Printer Query Results