December 2000

An Approach to Intelligent Query and Component Retrieval for Web-Based Repositories

Vijayan Sugumaran
Oakland University

Veda Storey
Georgia State University

AN APPROACH TO INTELLIGENT QUERY AND COMPONENT RETRIEVAL FOR WEB-BASED RESPOSITORIES

Vijayan Sugumaran
School of Business Administration
Oakland University
U.S.A.

Veda C. Storey
J. Mack Robinson School of Business Administration
Georgia State University
U.S.A.

Extended Abstract

1. INTRODUCTION

With the increasing amount of commerce performed over the Internet, there has been an expansion in the creation and use of web databases. From this, two trends have emerged. First, databases are developed "from scratch," even though it is well known that the development and employment of reusable artifacts is the most efficient way to approach the development process. Second, after a database has been developed, retrieval problems often exist, because there might be related information the user does not know about, or the user cannot express his or her requirements in natural language. The objectives for this research are (1) to develop an approach to obtaining intelligent results from a query to a web database and (2) to develop a procedure for defining and reusing domain models to assist in the development of web applications.

2. ARCHITECTURE FOR INTELLIGENT QUERIES

An architecture for a system is shown in Figure 1. The client side consists of a web browser. The server side is comprised of a query interface module, query refinement module, query execution module, domain model, ontology, repository of reusable domain objects (components), and the application database.

The client component gathers the user's queries and displays the results. The query interface on the server side captures the user's requirements, generates the preliminary database query, and displays the results. Natural language processing translates the user's query into a structured query language. The query refinement module enhances the initial query by using the domain specific information in the domain model and ontology. The query execution module uses the repository to execute the query and retrieve required information.

The reuse repository contains predefined objects (components) of an application domain. For each object it includes its purpose, objects upon which it depends, and its capabilities (methods) (Table 1). Part of the domain model is shown in Table 2 and organized into objectives, processes, actions, actors, and components (Sugumaran et al. 2000). The ontology provides additional information that is used in conjunction with the domain model and reuse repository to retrieve related information. It facilitates a more "intelligent" response because it enables the system to search for synonyms and related terms. The application database stores local data relevant to the application.

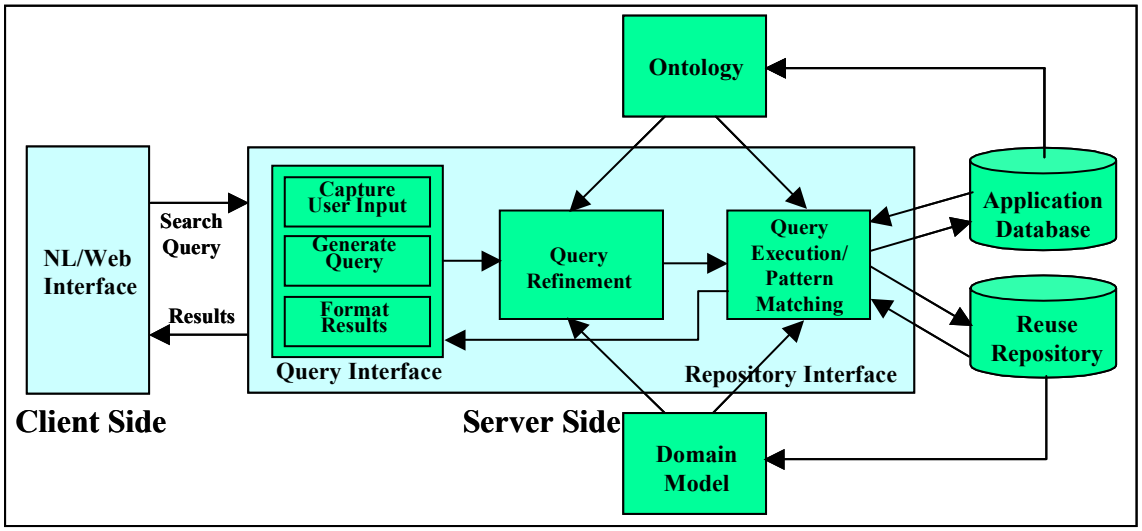

Figure 1. System Architecture

Table 1. Sample Component from Reuse Repository (Auction Domain)

Auction Item Component		
Attributes	**Methods**	**JavaBeans Implementation**

I_ID
Title
Description
PaymentMethod
MinimumBid
ReservePrice
SubmitTime
StartTime
CurrentPrice
• • • • •
	Access methods
	Update methods
	Review item price
	Review item description
	Review item condition
	Review item listing
	Analyze bidding activity
	• • • • •
		//Auction Item Class - with Bound property
		import java.beans.*; import java.util.*:
		public class Acution_Item extends Object {
		//instance variables
		protected int Item_ID;
		protected String Item_Title;
		protected String Item_Description;
		. . . //additional instance variables
		PropertyChangeSupport changeSupport;
		//constructor
		public Auction_Item() {
		changeSupport = new PropertyChangeSupport(this);}
		• • • • •

3. IMPLEMENTATION

A prototype of the system is under development using Jess, JavaBeans, and Java Server Pages.

Step 1 (User requirements): As shown in Figure 2, the user submits his or her requirements, for example: “find me opal ring to buy” and “restrict the search to three web auction sites.” The system takes the key phrases “opal ring” and “buy,” and searches the auction ontology to discover that “buy” has related terms “customer,” “buyer,” “seller,” and “item.” The system asks the user to select relevant terms, and assuming that the user selects only “item,” it searches the attributes of “item” to identify those that have something to do with “opal ring.”

The mechanism used to accept a natural language description and generate a corresponding SQL query consist of (1) a set of terms or phrases that indicate the initiation of a query; (2) a set of pronouns (ignored during query processing); (3) articles that can be parsed out; and (4) heuristics for creating SQL statements.

Table 2. Partial Auction Domain Model

Objectives	Processes	Actions
Facilitate Bidding/commerce Monitor Bidding process/bids Facilitate Transaction Facilitate Financing Market/trend analysis Provide Saving • • • • •	Initiate bid, Post bid Evaluate bid, Update bid Close bid Process transactions Customer Service Feedback • • • • •	Visit web site Search for wanted item(s) Open an account/registration Authentication Create product description Submit bid • • • • •

Actors	Components
Buyer, Seller Distributor, Auction Site Financial Institution • • • • •	Bid, Auction Item, Buyer, Seller, Account Auction schedule, Transaction Processing Payment processing, Database Interface • • • • •

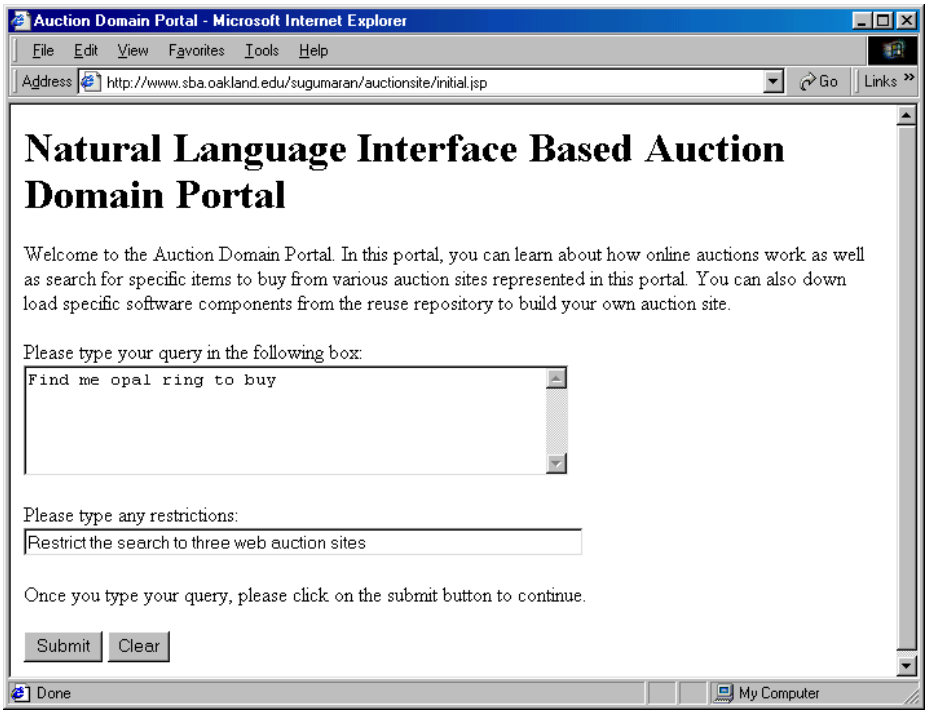

Figure 2. Initial Panel for Query Specification

Step 2 (Search query): The preliminary SQL query is generated.

```
Select *
From item
Where description like '%opal ring%'
```

From the results, the user can select one or more items to bid on (Figure 3). Once the bids are posted, the system attempts to provide additional information. From the ontology, the system discovers that an item is listed in a category; "opal ring" is classified under the jewelry category. The jewelry category can be related to other categories such as antiques.

Step 3 (Execute additional queries): The user can select related categories and SQL code is generated to obtain information on other items.

Step 4 (Refine query): A query may be refined or additional information retrieved based upon information in the domain model. For example, "search for wanted item" might be linked to the "open an account."

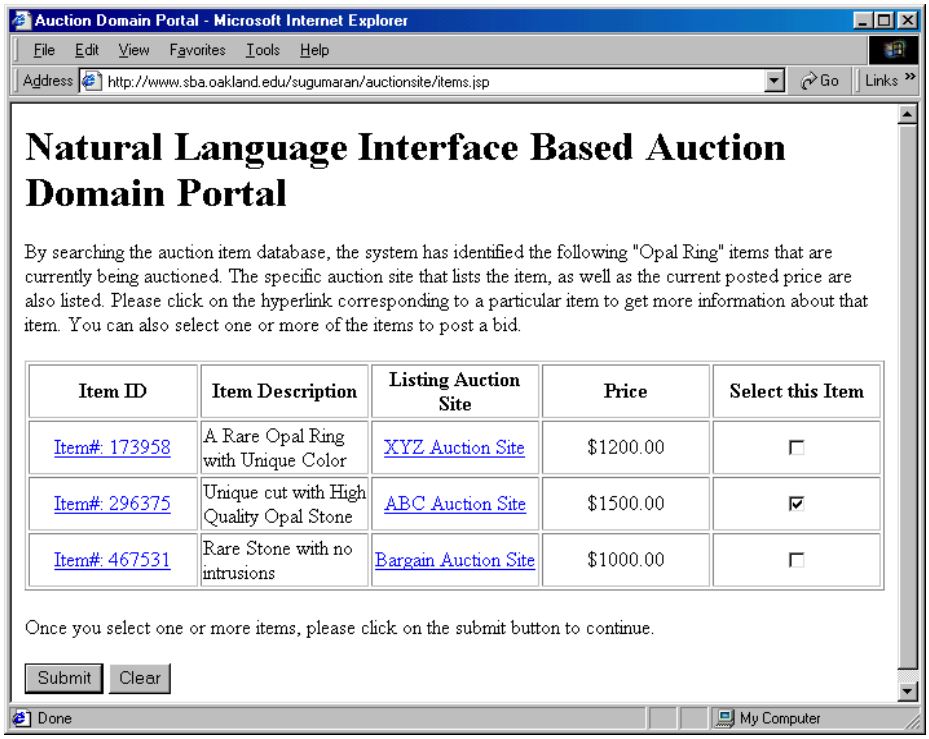

Figure 3. Results of Query Execution

4. CONCLUSION

An approach to the use of domain-dependent knowledge along with natural language parsing techniques has been presented that will enable a user to execute more intelligent queries. The approach is being implemented in a prototype system. In addition to facilitating the execution of intelligent queries, the system will provide a mechanism for effective reuse of application development components.

Reference

Sugumaran, V., Tanniru, M., and Storey, V. C. "A Domain Model for Supporting Reuse in Systems Analysis," *Communications of the ACM*, 2000.