INTELLIGENT RETRIEVAL
Open Journal

Review

Distributed Visual Reasoning for Intelligent Information Retrieval on the Web

C. Lee[1];Y-T. Chen[2]

[1]*Department of Computer Science and Engineering, National Sun Yat-Sen University, Kaohsiung, Taiwan*
[2]*Department of Information Management, National Sun Yat-Sen University, Kaohsiung, Taiwan*

[*]***Corresponding author***
C. Lee
Department of Computer Science and Engineering, National Sun Yat-Sen University, Kaohsiung, Taiwan

Article information
Received: February 6th, 2024; **Revised:** March 2nd, 2024; **Accepted:** March 31st, 2024; **Published:** April 7th, 2024

Cite this article
C. Leea, Y-T Chen. Distributed visual reasoning for intelligent information retrieval on the web. 2024; 2(1). doi: https://doi.org/10.70705/ppp.ir.2024.v02.i01.pp24-29

ABSTRACT

For smart Web information retrieval, we provide a distributed visual reasoning system in this work. Graphical programming, the World Wide Web, Java applets, an inference engine, and servers for databases are all part of the system. In order to facilitate the retrieval of information, it mediates messages going between the Web clients, inference engine, and database servers and provides users with a visual programming interface. A web server stores the necessary web page and provides the necessary system setup services. Java applet functionality may be activated by downloading the necessary classes, information, and the visual programming interface agent over the Web client and the HTTP protocol. In the visual programming interface, the user may construct visual statements and input parameters to get the necessary data. In order to provide faster and more reliable communication, a distributed communication paradigm is suggested. As an example of one of many possible uses for our technology, we provide a financial diagnostic. All rights are held by Elsevier Science B.V., copyright 2000.

Keywords
World wide web; Information retrieval; Visual programming; Java; Financial diagnosis; Distributed system

INTRODUCTION

With the exponential growth of database sizes comes the pressing need to glean useful insights from seemingly mundane sources, including stock indices and financial reports, in order to better understand macroeconomic patterns, commodities, and markets [1-6]. When it comes to market research, the Spotlight system was an early adopter of information retrieval techniques [2]. With this solution, consumers can finally make sense of the deluge of marketing data that's flooding the industry. being scanned at grocery stores. The user's interests determine the kind and depth of the information that is supplied to them in a timely manner [1]. Embedded knowledge-based systems for understanding data in massive point-of-sale databases have never been seen in a product before. According to Ref. [5], a user's interest profile informs the search for text-based content using an Intelligent Agent (IA) product. Multiple agents may operate together in DIEIS [6], a framework that facilitates sophisticated information processing. There, a coordinator facilitates a dispersed set of agents' collaborative efforts to find a solution to a complicated issue. Although the concept is brilliant, the method has not been put into action just yet.

Additionally, there has been a meteoric rise in the need for programming systems that are easy to use for both professionals and those without coding experience. The incorporation of visual data into the man-machine interface is one method that has proven effective in meeting this need [7–12]. Any language that allows for the processing of visual information or the expression of programming via visual methods is considered a visual programming language. One emerging trend in human-computer interactions is the use of facial expressions as a means of communication. Usually, a person will draw an image, a diagram, or a visual example, and the computer will decipher the user's purpose based on the visual expression.

Given the widespread usage of the World Wide Web, connecting to a database server over the Internet is only logical. The systems mentioned in Refs. [13–15] are many. However, this is only the beginning of the process of extracting information from the WWW's low-level data. A user may contact a database server on the web by sending a query to an HTTP server using a web browser. After the HTTP server receives a request from a user, it starts the Common Gate Interface program to process the request and sends the results of the query back to the browser. A static HTML-based interface is used for this basic database search. Despite the Web's easy and consistent user interface, the majority of requests are still typed. Incorporating visual programming into a web browser is certainly a good way to

improve the browser's capabilities.

Here we introduce a web-based intelligent information retrieval system that uses distributed visual reasoning. By retrieving the relevant functionality, implemented in Java applets, from a web server, the system provides the web client the ability to visually program. After turning on the agent for the visual programming interface, a user may make use of the system's pre-existing active icons to construct his own visual program by linking the icons, and then enter the necessary parameters. By entering the SQL command in the pop-up parameter menu, a user who is acquainted with SQL may immediately access database information. In order for the inference agent to make inferences and extract information, a parser that is incorporated in the visual programming interface agent must first transform the visual representation into the relevant inference rules. Here is how the rest of the article is structured: Section 2 lays out the groundwork for the rest of the document by outlining the fundamental theory, building blocks, and design principles; Section 3 provides an example of a financial diagnosis to illustrate how the system interacts; Section 4 provides a high-level overview of the system's architecture; and Section 5 presents the results.

2. System design concepts

The fundamental ideas, components, and philosophy of the system will be laid forth in this part.

Principles of design (2.1)

Improved human-computer interaction during data retrieval is one of our system's primary objectives, along with facilitating easy system access and requiring little modification to the Web client. In developing the system, we adhered to the following four guidelines.

2.1.1. A built-in component

The present Web clients and database servers must not be altered. This article describes an embedded module that may be used in an existing client/server architecture. The visual interface applet is a module that runs in the client browser, and the system itself is a mediator. Users are able to extract the necessary information using the system's functions, which operate as a bridge between the Web client and servers.

2.1.2. Reasoning through visuals

As the majority of web clients still transmit queries to web or database servers via text or template inputs, one of the main concepts of the system is to provide them the capacity to reason visually. Users are provided with an additional way to study the network's information using pictorial and symbol icons in a visual reasoning interface.

2.1.3. Orientation for users

In order to increase the likelihood that users would actually utilize the system, it is important to keep them in mind when designing it. A user-friendly interface that is consistent across platforms, has good visualization, is simple to use, and is quick to understand is the most critical factor in user orientation. Application development on top of the Web client has become commonplace due to the browser's ubiquitous nature as an interface for accessing data and information online. Graphical inputs and outputs made possible by visualization enable two-way pictorial communication between machines and people. Many cutting-edge software programs follow this pattern.

2.1.4. Communication over a network

A Common Gate Interface bottleneck on the HTTP server [15] may be avoided using Java technology via the socket connection. One possible source of bottlenecks is when several web clients try to access database servers over the same gateway. Using a distributed communication approach, the system is able to avoid crowded communication traffic, which in turn increases speed and dependability.

Section 2.2: Blocks

There are primarily two sorts of parts to the system, and these are agents and icons.

Model based on agents (2.2.1)

Cooperation issue solving [6] and distributed problem solving [16] are other names for agent-based models. Agents in this paradigm coordinate their efforts to accomplish a common goal or carry out a complicated task. The system's coordinator, who distributes duties on behalf of other agents, is called a message handling agent (MHA). For instance, in order to resolve the overall inference issue, it delegated the reasoning agent's responsibility to infer and the database server's method to retrieve facts. An inference engine and reasoning process make up the reasoning agent. The rules for production serve as the reasoning process. Based on a commercially available inference engine product known as M.4 [17], the inference engine is built.

Section 2.2.2: Icons that are currently active

An active icon serves as the visual component's primary in a visual programming interface. An active icon has a visual representation, data, actions, limitations, body of knowledge, and logic. Active icons not only serve the same purpose as traditional icons, but they also enable users to self-correct, self-learn, and conduct consistent checks. An item with a visual representation, textual description, and interactive capabilities is called an active icon. On the screen, the picture stands in for an icon. There are multimedia files, properties, constraints, and more in the data. The icon's behavior explains how the symbol may accomplish its goals, present its information, get facts from an item, and use its reasoning methods. Reasoning, decision-making, calculation, and fundamental concept data are the several kinds of active icons used for information retrieval. Characteristics shared among icons in the same category are very consistent.

2.3. Visual languages and parser

In the context-free visual grammar for information retrieval, G = {N, ō, P, S}, the four-tuple symbolizes the following: N is the set of non-terminal symbols, ξ is the set of terminal symbols, P is the set of production rules, and S is the set of start symbol. The intersection of N and ξ is …. An icon is connected to each grammatical symbol. The function of non-independent variables represents the product rule P, which is the connection between grammar symbols. rel (N, *o*) represents the terminal and terminal symbols. The syntax for the production rules is Sen → rel (N, ξ), where Sen may be either a terminal or non-terminal symbol, a start symbol, or a combination of the two. The linkages may take many forms, including connections, calculations, choices, parallel processing of computations and decisions, etc. The visual grammar is shown with a specific example in Appendix A. A In order to deduce rules for producing inferences

from visual sentences, a parser is used. This parser is based on the context-free grammar that was previously discussed. An icon dictionary, rules for iconic inference, a visual program, and a non-terminal icon dictionary are all parts of the interpreter.

3. The application model

Visual reasoning is an appealing technique for information retrieval applications that have an inference structure, such as financial diagnosis and technical analysis. Fig. 1 provides a conceptual framework for the development of information retrieval applica- tions on the Web. The model consists of a presentation layer, a communication layer, and a data and service layer. The presentation layer consists of understanding and design levels. In the understanding level a user should know what applications are available on the current system, understand about the problem to be reasoned and the inference structure of the applications. The current system supports a financial diagnosis application and a technical analysis application. In the design level the user can just drag and connect the visual icon to make information retrieval, and inference rule generation. The parser takes the visual sentence and their parameters as inputs from the visual programming interface, then uses the decision tree to represent the inference structure. Then it transforms the decision tree to if–then rule format directly. Similarly, the decision table is used to describe the decision strategy for each node in the decision tree. The communication

layer composes of the Web server agent, and communication protocol. In the data and service the domain application database servers, and reasoning processes are provided. The system aims for financial experts, who are not familiar with using the SQL language, to retrieve information from the database and the rules of M.4 expert shell to make an inference. Of course, it is also useful for an advanced user who knows the SQL language, but not the rules of M.4. Then he/she can design the inference structure by connecting the visual icons and then assign the parameters to each active icon. To illustrate this system we used a financial diagnosis application as an example. We will focus on the design of active icons, inference rules, visual reasoning grammar, and fuzzy functions.

3.1. Defining the application domain

3.1.1. A case study of a financial diagnostic

For a company to succeed, it is crucial that its employees make sound judgments when the moment is right. Because making sound financial choices is of paramount importance to organizations in today's dynamic and technologically advanced business world, financial management is both a fascinating and demanding field. The use of expert systems technology in the financial sector improves decision quality and increases agreement across various decision-making processes [18].

Among the many important subfields of financial management, financial analysis stands out. Expert financial analysts have a hard time finding the right words to express their knowledge and skills when it comes to the decision process, a complex topic in financial analysis [19]. Finding the right way to acquire and convey information in financial analysis is an issue that has been the subject of much study [3,4,20]. In the knowledge acquisition stage, decision trees and decision tables are often used in the literature as visualization aids. In the knowledge representation stage, they are transformed into text-based production rules. In Appendix B, you may find a straightforward statement on financial diagnosis that is based on references [3, 4].

Figure 2 is a section of Figure 14's finance inference framework. It takes a lot of basic facts like current assets and short-term liabilities, as well as a lot of inference and reasoning, to determine the firm's key performance characteristics like financial leverage, liquidity, financial structure, and the final inference goal-financing
status.

Section 3.2: Essential visual elements

3.2.1. Simple symbols

There are five distinct types of icons: basic, composite, fuzzy, decision, and goals icons. As you can see in Figure 3, there are five distinct kinds of icons. Wires link the icons together. There are a total of twenty-five possible permutations, since two icons may be linked by a wire. However, there are links that don't imply anything in a semantic sense. Therefore, in order to give the semantic validation, we used the restrictions of each active icon. Specifically, there are seven types of legitimate connections: BAsIc coNcepT to coMpUTA- TIoN, BAsIc coNcepT to fUzzY, coMpUTATIoN to coMpUTATIoN, coMpUTATIoN to fUzzY, fUzzY to decIsIoN, decIsIoN to goAL, and decIsIoN to coMpUTATIoN. The BAsIc concept incorporates the database queries using SQL strings. The capacity to verify the association between two active icons is a crucial feature of active icons.
The information that travels across the links in our system serves as a signal to initiate the inference rules, while the symbols on the blackboard stand in for the facts. The inference engine will be able to draw conclusions from the system-generated set of inference rules after the users have entered the visual sentence properly. One form of icon is a full inference rules set, which is a tree structure. What this means is that you only need one kind of icon to build a whole set of inference rules. In order to interact with other components, such a database, and generate an inference, a user may need to understand distinct features of each kind of icon.

3.2.2. Making symbols that can be clicked

The generation of active icons (3.2.2.1). The five built-in basic kinds of active icon classes are listed below; the primary interface to the system employs one of these classes:

First, the idea fundamental concept: this icon class symbolizes things like current assets, short-term obligations, net working capital, and so on. Each accounting idea may be reused by creating instances of the fact-retrieving algorithm that is encoded in the icon, which is a

pair of SQL statements provided by users.

2. The computation icon class calculates the associated fundamental concepts using user-defined operations.

3. The Fuzzy Icon Class: This class allows users to separate the core idea or calculation result into different intervals.

4. The decision icon class helps the inference process and the user construct the decision table by providing an intermediate objective.

5. The goal icon class emphasizes the end result, much as the choice symbol class.

By inheriting the active icon class, users may develop additional sorts of active icon classes. Appendix C shows the icon classes that are now active, along with their inheritance structure. Numerous BAsIc coNcepT icons, each with its own unique set of SQL strings representing various financial diagnostic and technical analytical inference structure attributes, have been supplied by the system. A user may easily include these pre-made icons into the design section of the financial diagnostic program without the need to submit any SQL instruction. In the case of individuals users who know the SQL commands can change the properties and then create new instances of BAsIc coNcepT icons for new applications.

3.2.2.1. Fuzzy function. For the fuzzy result icons, we built-in a fuzzy function. The purpose is to transform the continuous numbers into a fuzzy number with a certainty confidence factor.

3.2. Interaction with the system

In this subsection we show how to construct an application with the system.

3.2.1. Designing the visual inference structure

Based on the inference structure in Fig. 3. A user can design her/his own application in the Visual Programming Area by dragging the visual component from the Visual Compo- nent Box, and arranging and connecting them to create the inference structure of an application as shown in Fig. 4.

Fig. 4 shows the outlook of the visual programming interface agent and an example of the design of the reasoning inference structure by dragging the visual component from the Visual Component Box into the Visual Programming Area, and arranging and connecting them to create the inference structure of an application. The Visual Programming Area can be extended by scrollbars on both horizontal and vertical direction.

3.2.2. Filling in the parameters of each icon

After creating inference structure, a user must fill in parameters to generate his/her own inference rules. In Fig. 5, an advanced user can click the right button of the mouse on the Basic Concept Icon representing "Current assets" to pop a sub-window and fill in the "icon name", "icon caption" and "SQLString" fields. The SQL string text area is for those users familiar with the standard SQL-92 language to directly retrieve data. In Fig. 6, a user can click the right button of the mouse on the Computation Icon representing "net-working capital" to pop a sub-window and fill in the "icon name" and "icon caption" fields. The grayed out textfields provides the operators and operands for selection only. The available operators are plus, minus, multiplication and division, and the available operands list is generated by the connected-from icons. Users can not only select but edit the two fields. In the Java AWT, the gray color means that the column cannot be edited. In Fig. 7, a user can click the right button of the mouse on the Fuzzy Icon representing "long-term inventory financing status" to pop a sub-window and fill in the "icon name", "icon caption" and "no. of conclusion" fields first. Then the interval section will show up. Based on the "no. of conclusion" field a user can fill in each boundary for the interval in the "if" part and conclusions in the "then" part. In Fig. 7 the boundary values are 0 and 5, this means, when the input is 0, the status of the long term inventory financing is declined; when the input is between 0 and 5, then it is normal; when the input is larger than 5, then it is improving. In Fig. 8, a user can click the right button of the mouse on the Goal Icon representing "Financial structure" to pop a sub-window and fill in the "icon name" and "icon caption" fields. Then the "if" part will show up automatically based on the pre-icons connecting to it and the user should fill in the "then" part.

3.3.3. Reasoning results

After clicking the "Run" button in the toolbar in Fig. 8, the inference window as shown in Fig. 9 will pop up. The actions in the inference window contain go, restart and stop the inference process. The inference result area shows the inference consultation and relative messages. The status message box shows the status of the inference agent (IA). When the inference agent detects incomplete facts, an interactive mechanism is provided by prompt- ing the related question in the question area to remind the user to input the required items. The item can be selected from the item menu, and the corresponding confidence factor (CF) can be filled in. Other interactive functions such as help, OK, and unknown buttons are provided too. After a user clicks the "go" button the rules and fact retrieving algorithms will be transferred to the MHA from which messages are sent to the inference agent and database server. The inference result is sent back to the Web browser through the MHA. Based on the data and inference rules provided, the results of the financial diagnosis are

generated as a text form that shows the "long-term inventory financing status" is improved with 79% confidence factor, etc. (see Fig. 9). Fig. 10 is a line chart of the basic concept "Cash". The user can also terminate the visual program at the BAsIc coNcepT or Computation Icon to see the trend of the query result or computation result. The sequence of numbers of the trend is the result of a query or computation. In this example, the "Cash" icon with SQL can generate a sequence of numbers, and the chart window will draw the line chart based on these numbers. More visual presentation objects can be inserted into the system by dynamic loading Java Classes in the future. Since the system provides the default knowledge base, a user can load those default files to modify and execute too. The application of the system to technical analysis is in Ref. [21].

4. System architecture

Distributed Web clients, Web servers, MHAs, inference agents, and database servers make up the overall functional system shown in Fig. 11. When users request information from a database, the system acts as a go-between for the browser and the server. Tight coupling to database servers is possible if they implement standard SQL and ODBC or comparable protocols. Because of this, the system may be integrated to fulfill its function without requiring modifications to the Web client or the current database servers.

Anywhere in the network may host the web clients. To access the Web server agent's HTML, Java classes, picture files, and associated functionalities, it use the conventional HTTP protocol. After Java applets move to the web client, there will be no longer be a link between the web clients and the web server. Web clients, database servers, and rule base servers make up the remaining connections. As a middleman, a solitary server carries the request and answer messages from the applet to the distant server in this three-tier design [14]. Agents are able to communicate with one another via the MHA. At one end, it uses system-defined protocols to interact with the Java applet; at the other end, it uses the original client/server protocols to contact the database servers. Currently, the MHA communicates with the inference agent using DDE, the database server through ODBC, and the visual programming agent through the TCP/IP module. The sections that follow provide in-depth explanations of every component seen in Figure 11.

4.1.1 Agent for the visual programming environment

An agent that allows users to interactively draw inferences and get information is the visual programming interface agent. Figure 3 shows how the visual programming interface (VPI) processes user inputs including dragging and dropping visual components from the VCI, clicking and modifying text, and then displays the appropriate messages and icons.

A user may request that the Web server agent load all associated functionalities wrapped in Java applets in order to add the visual programming interface agent to the Web client. In Fig. 3 a user can click buttons on the toolbar to load, save and interpret the visual sentence. The system provides a visual component editor to allow users to input parameters.

The visual sentence is interpreted by an interpreter that is embedded in the visual programming interface agent, then send the parsing results to the MHA. The message handling takes the requests from the visual programming interface agent through the Internet, passes the requests to the inference agent to do inference or to the database server to retrieve the facts. The requests include parameters, data or execution commands that are accepted by the inference agent.

4.1. Distributed communication model

The information retrieval process is a combination of the inference agent's reasoning processes, the MHA's calculation activities, and communication between those agents and the database servers. As seen in Figure 12(a), we use a communication paradigm similar to Common Object Request Broker Architecture (CORBA) [22] to prevent the system's communication traffic from becoming crowded. In comparison to the models in Figure 12(b) and (c), this model's three types of components—service providers, service brokers, and service requesters—enable faster communication and a more dependable dispersed connection.

5. Conclusions

A visual programming system for distributed information retrieval has been created by us. The current Web client and database servers communicate with each other via the system. In addition to the standard textual and template inputs, it augments the Web browser with visual programming capabilities, giving users access to a new level of programming. For better adaptability, dependability, and load balancing, the system uses a decentralized communication paradigm. As the quantity of services increases, so does the dependability.

brokers. Financial diagnostics and technical analysis make use of a variety of BAsIc coNcepT icons, each of which has its own set of features. A user may create their own active icons by specifying the scope of applications and then filling out the calculation, parameters, inference rules, and fuzzy function. Users must enter certain values in order for programs to retain their flexibility. Inferences show that the system works effectively when used online.

REFERENCES

[1] Journal of Intelligent Information Systems 4 (1995) 27–37; T. Anand, Opportunity Explorer: Navigating Large Databases with Knowledge Discovery Templates.

[2] In the Proceedings of the ninth IEEE Conference on AI for Applications, 2-8, 1992, T. Anand and G. Kahn presented SPOTLIGHT, a data-explanation system.

A decision support method with applications in management and finance, by M. Klein and L.B. Methlie, published by Addison-Wesley in Reading, MA in 1990 [3].

[4] In W.E. Norton & L.B. Methlie's 1995 book Knowledge-Based Decision Support Systems: A Primer for Business, Chichester: Wiley.

[5] Hypermedia: EIS's future, by M. Frolick and N.K. Ramarapu, Journal of Systems Management 44 (1993) 32–36.

[6] In Decision Support Systems 14 (1995) 117–130, R.T. Chi and E. Turban discuss distributed intelligent executive information systems.

In Principles of Visual Programming Systems, edited by S.K. Chang and published by Prentice Hall in 1990, K.T. Huang discusses visual interface design systems.

[8] In Computer's Special Issue on Visual Programming (1985), R.B. Grafton and T. Ichikawa wrote the following.

The impact of visual technology on the development of language environments was discussed in an article by A.L. Ambler and M.M. Burnett published in Computer 22 (1989) pages 9-22.

S.K. Chang published an article in the IEEE Transactions on Software Engineering in 1990 titled "A visual language compiler for information retrieval by visual reasoning" (Volume 16, pages 1136–1149).

In 1989, N.C. Shu published Visual Programming in New York via Van Nostrand Reinhold.

Supporting user-adapted interface design: the USE-IT system, Interacting with Computer 9 (1997) 73-104, [12] D. Akoumianakis, C. Stephanidis.

[13] In: Proceedings of the fifth International World Wide Web Conference, Paris, 1996, S.E. Dossick and G.E. Kaiser discuss WWW Access to Legacy Client/Server Applications.

[14] N.N. Duan, Java-based distributed database access in a business setting, in: Proceedings of the 1996 International World Wide Web Conference, Paris.

[15] In the Proceedings of the Fifth International World Wide Web Conference, held in Paris in 1996, J.E. Pitkow and R.K. Jones presented Supporting the Web: a distributed hyperlink database system.

[16] Foundations for collaboration in remote problem solving, R.G. Smith and R. Davis, IEEE Transactions on Systems, Man, and Cybernetics 11 (1981), 61–70.

[17] Teknowledge Cimflex, M4: User's Guide, Palo Alto, 1991.

Choosing the optimal expert system development technique for financial decision-making, by K.L.K. Yiu and A.W.K. Kong, Journal of Systems Management 43 (1992) 16-19, with references to pp. 30-31; 37-38; and 40-43.

[19] In Expert Systems With Applications 12 (1997) 247-262, N.F. Matsatsinis, M. Doumpos, and C. Zopounidis discuss knowledge acquisition and representation for expert systems in the domain of financial analysis.

Decision support and expert systems: Management support systems, edited by E. Turban, published by Macmillan in 1993, is cited as [20].

C. Lee and Y.T. Chen are the authors of paper [21]. A Web-Based Intelligent Information Retrieval Visual Programming Interface. California, USA, 1997, pp. 46-53. Published by the IEEE Knowledge and Data Engineering Exchange Workshop.

S. Vinoski, "CORBA: integrating diverse applications within distributed heterogeneous environments," IEEE Communications Magazine 35 (1997) 46-55.

[23] Prentice Hall, Carmel, 1992, J.D. Clark, Windows Programmer's Guide to OLE/DDE.