Web-enabled Healthcare System for Hypertension: Hyperlink-based Inference Approach

Yong Uk Song[a], Young Moon Chae[b], Seung Hee Ho[c], Kyoung Won Cho[d]

[a] Department of Management Information Systems, Yonsei University Wonju Campus
234, Maji, Wonju, Gangwon 220-710, Korea
Tel: +82-33-760-2340, Fax: +82-33-763-4324, E-mail: yusong@dragon.yonsei.ac.kr

[b] Graduate School of Health Policy and Administration, Yonsei University
134, Shinchon, Seodaemoon, Seoul 120-752, Korea
Tel: +82-2-361-5048, Fax: +82-2-392-8996, E-mail: ymchae@yumc.yonsei.ac.kr

[c] Graduate School of Management, Korea Advanced Institute of Science and Technology
207-43, Cheongryangri, Dongdaemoon, Seoul 130-012, Korea
Tel: +82-2-958-3685, Fax: +82-2-958-3604, E-mail: hsh@kgsm.kaist.ac.kr

[d] Department of Medical Information Systems, Choonhae College
72-10, Gorkchun, Woongchon, Ulju, Ulsan, 689-872, Korea
Tel: +82-52-270-0252, Fax: +82-52-270-0461, E-mail: kwcho@choonhae.ac.kr

Abstract

In the conduct of this study, a web-enabled healthcare system for the management of hypertension was implemented through a hyperlink-based inference approach. The hyperlink-based inference platform was implemented using the hypertext capacity of HTML which ensured accessibility, multimedia facilities, fast response, stability, ease of use and upgrade, and platform independency of expert systems. Many HTML documents, which are hyperlinked to each other based on expert rules, were uploaded beforehand to perform the hyperlink-based inference. The HTML documents were uploaded and maintained automatically by our proprietary tool called the Web-Based Inference System (WeBIS) that supports a graphical user interface (GUI) for the input and edit of decision graphs. Nevertheless, the editing task of the decision graph using the GUI tool is a time consuming and tedious chore when the knowledge engineer must perform it manually. Accordingly, this research implemented an automatic generator of the decision graph for the management of hypertension. As a result, this research suggests a methodology for the development of Web-enabled healthcare systems using the hyperlink-based inference approach and, as an example, implements a Web-enabled healthcare system for hypertension, a platform which performed especially well in the areas of speed and stability.

1. Introduction

Along with the explosive increase of information services using the World Wide Web (WWW), the practical application of web-based expert systems has shown tremendous growth (Song and Lee, 2000). Among these expert systems, medical expert systems are of particular interest. Indeed, accessibility and ease of use of the WWW offer a unique opportunity for remote healthcare in medical domains. Traditionally, diagnosis has been a major application domain of backward chaining inference.

Numerous inference engines, which were used in stand-alone environments, have been adapted for the Web environment. The Web-based inference engines use some of the latest technologies, including multi-thread and Java, but they suffer from the problems of heavy traffic burden to Web servers and a corresponding high price. The hyperlink-based inference approach, which was introduced by Song and Lee (2000), offers a solution to the Web-enabled, rule-based systems by combining speed of response with application stability. The authors developed a Web-enabled healthcare system for the management of hypertension using the hyperlink-based inference methodology.

It should be noted that the hyperlink-based inference approach is not without its own problems, specifically in the areas of difficulty in development and maintenance. To cope with the problem, the researchers adopted a proprietary HTML generator, called WeBIS, and developed a graph generator. The graph generator is a tool that draws a decision graph based on domain knowledge, and the HTML generator is a tool that converts the graph to a set of HTML files suitable for upload to Web servers; once located on the Web servers, the HTML files provide inference service in cyberspace.

This paper is organized as follows. In section 2, the motivation and methodology of the hyperlink-based inference approach are introduced. The domain knowledge for the management of hypertension is described in section 3, and the issues and methodology of implementing the system are explained in section 4. Finally, conclusions are presented in section 5.

2. Web-enabled, Rule-based Systems

2.1 Structural Comparison of the Web-enabled, Rule-based Systems

We can classify existing methodologies to build Web-enabled, rule-based systems into five categories according to the applied technologies. Table 1, below, shows those five categories and their summarized technical features.

<Table 1> Categories of Web-enabled, rule-based Systems

Location of Inference Engine	Type of Inference Engine	Technical Features
Server-side	CGI program	* Uses Common Gateway Interface (CGI) standard * Web server invokes a CGI program while passing required parameters according to CGI standard
	Server-side script	* Inference engines are developed under environments such as JSP, ASP, and PHP * Transaction processing and multi-threading functions are provided by default
	Web server embedded module	* Inference engines are embedded into Web server as a sub-module using API such as NSAPI
Client-side	External viewer	* Is developed as an independent program * Is invoked by Web browser according to predefined MIME type
	Java applet	* Bytecodes for the inference engine are located in the server-side but transmitted to a Web browser. * Java Virtual Machine (JVM) in the client-side interprets and executes the bytecodes.

As shown in Table 1, existing methodologies fall into two broad categories: server-side and client-side. This distinction depends on the location of the inference engine of a Web-enabled, rule-based system. The server-side category can be further divided into three more detailed categories: CGI program, server-side script, and Web server embedded module depending on the type of inference engine actually implemented. The client-side category is further classified into two subcategories, external viewer and Java applet, also with regards to the implemented type of inference engine. The technical features column of Table 1 describes the characteristic features of each category.

Concerning practicality and efficiency in developing a Web-enabled, rule-based system, let us examine the pros and cons of each category mentioned above.

The most popularly used approach for Web-enabled, rule-based systems is the CGI program approach. Well-known rule-based management systems such as EXSYS (EXSYS), Blaze Advisor (http://www.blazesoft.com/), and ILOG (http://www.ilog.com/) are based on this approach. One of the major reasons for the popularity of the CGI program approach is that it is generally easier to maintain or extend rule-based systems as compared to either Web server embedded module or client-side approaches.

However, this CGI program approach has revealed a serious weakness in responding to requests from clients as the number of clients has grown faster than increases in computing resources. Currently, server-side script approaches are recognized as an alternative to shore up this weakness in the CGI program approach. Efficiencies are gained through the server-side script approach's use of multi-threading instead of forking and provides an automated transaction processing functionality to reduce traffic burdens to the Web server. Even though the server-side script approach is better than the simple CGI program approach, the Web server burden incurred by this approach is still much more than in the case of HTML documents, thereby causing an over-burden problem due to limited computer resources.

The third approach belonging to the server-side method is the Web server embedded module approach in which the Web server contains a rule-based system as a sub-module. This system design complicates both maintenance and the extension of the rule-based system making it more cumbersome than either the CGI program or server-side script approaches. The authors conjecture this is the major reason there are few commercial Web-enabled, rule-based systems developed using the Web server embedded module approach.

As a subcategory of the client-side approach, the external viewer method uses an independent rule-based inference system, which resides in the client computer. Therefore, it may reduce the computational burden to the Web server at some limited level but maintenance of the system is more difficult than in the server-side approaches because each program must be installed separately from the Web server. This requires reinstallation of the system every time the system is updated, which effectively prevents its application to developing a commercial-purpose, Web-enabled, rule-based system.

Another subcategory of the client-side approach is the Java applet approach which actually solves some of the major disadvantages of the external viewer approach such as version management and consistent maintenance based on Java's platform independent framework. Some firms such as ILOG and e2gLight (http://www.expertise2go.com/) use the Java applet based approach due to its obvious merits, yet, it is nevertheless true that Java has not gained wide acceptance in industry as a practical and commercial approach in developing Web-enabled, rule-based systems. By design, the Java applet approach needs bytecodes, that is, executable programs downloaded from Web server to client at the time of consultation of a rule-based system while not placing any further burdens on the Web server. Instead, bytecodes cause additional data traffic such as rule base and sharing parts of databases (in some cases, it may be whole databases) between server and client. But the size of the rule base and the required part of databases for inference are usually substantially larger than the size of the inference engine found in most commercial cases. This seems to be the main reason the Java applet approach has not gained wider acceptance for commercial application.

Reducing latency time is one of the most critical design factors for success of a Web site (McCarthy, 1999) and this can be directly applied to Web-enabled, rule-based systems. From a review of related approaches to Web-enabled, rule-based systems, the authors have identified reducing latency time (defined as reducing burdens caused by rule-based systems to the Web server) as the most important but not fully resolved issue. Reducing latency time, then, becomes the focus of improving the efficiency of Web-enabled, rule-based systems.

Under the premise that we can make a rule-based system only with hyperlinked HTML pages and there are no functional differences between hyperlink-based systems and conventional Web-enabled, rule-based systems approaches, the hyperlink-based systems approach is clearly more efficient than the conventional approaches using CGI program or server-side scripts in terms of latency time. Yet, if we want to avail ourselves of this advantage of the hyperlink-based systems, we must first satisfy the premise. To satisfy the premise, we need to show how a rule-based system can be implemented with a set of hyperlinked HTML pages.

2.2 Hyperlink-based Inference

Song and Lee (2000) introduced a hyperlink-based inference mechanism. They showed that inference sites could be published on the Web using only the hypertext capability of HTML without the aid of inference engines. To illustrate, let us assume the following example of a rule base (Figure 1):

```
If signal-light IS red
THEN stop

If signal-light IS green
THEN go
```

[Figure 1] An Example of a Rule Base

The above rule base can be formed on the Web as shown in Figure 3 when a Web site is set up with the three HTML files shown in Figure 2. The signal-light "Red" hypertext in Figure 3.a becomes the hyperlink in Figure 3.b, and the signal-light "Green" hypertext becomes the hyperlink in Figure 3.c. When a user clicks the hypertext "Red" in Figure 3.a, the site shows the conclusion "Stop" in Figure 3.b, and, when a user clicks the hypertext "Green", the site shows the conclusion "Go" in Figure 3.c. This procedure is exactly same as that of inference on the rule base in Figure 1.

File: a.html

```
<HTML>
<HEAD> <TITLE> Decision Tree </TITLE> </HEAD>
<BODY>
<BR> Signal light <BR><BR>
<BLOCKQUOTE>
<A HREF="b.html"> Red </A> <BR>
<A HREF="c.html"> Green </A>
</BLOCKQUOTE>
</BODY> </HTML>
```

File: b.html

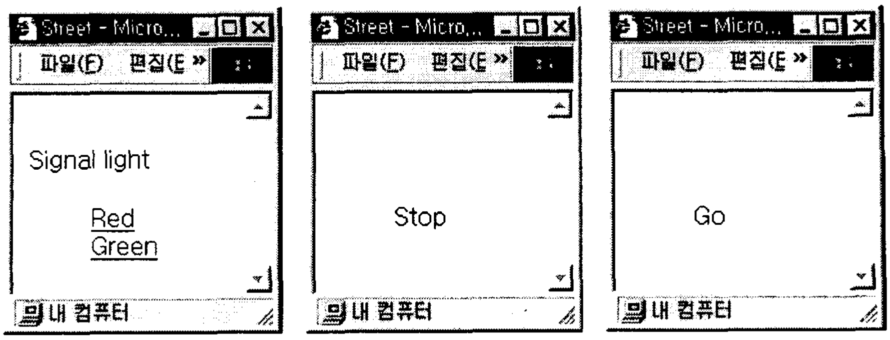
```
<HTML> <HEAD>
<TITLE> Decision Tree </TITLE></HEAD>
<BODY>
<BR><BR><BR>
<BLOCKQUOTE>
Stop
</BLOCKQUOTE>
</BODY> </HTML>
```

File: c.html

```
<HTML> <HEAD>
<TITLE> Decision Tree </TITLE></HEAD>
<BODY>
<BR><BR><BR>
<BLOCKQUOTE>
Go
</BLOCKQUOTE>
</BODY> </HTML>
```

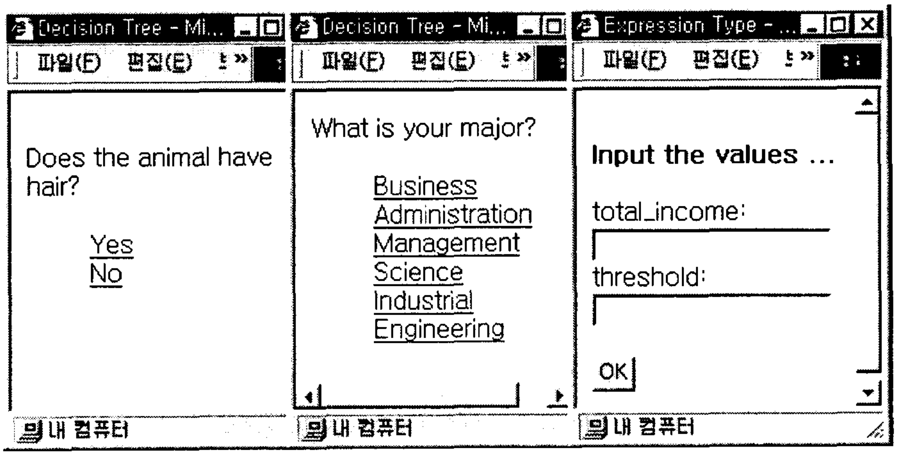
[Figure 2] HTML Files for Inference

To implement inference, such things as fact input, chaining based on subgoals, and output of inference results have to be supported. (Lee et al., 1996B) Chaining based on subgoals can be formed by the hyperlink as discussed above, and the output of inference results will work if the Web browser is made to display the HTML files. Therefore, the remaining problem is identifying and applying a method that supports fact input. The value is saved as a variable in inference. The format of the variable from inference can be of three types: fact type, object-attribute-value type (OAV), and numeric type. (Lee et al., 1996A) The variable "signal light" in the rule base in Figure 1 is the OAV-type variable having a value of "Red" or "Green," and this is formed as in Figure 3.a. This also works in the case of the fact type variable as well when designated "true" or "false." For example, the first screen image in Figure 4 depicts the condition of the rules that the fact-type variables shown below generate.



<Figure 3.a> <Figure 3.b> <Figure 3.c>

[Figure 3] A Hyperlink-based Inference Site



[Figure 4] Inquiry screens by variable type

IF "Does the animal have hair?" IS TRUE
THEN ...
IF "Does the animal have hair?" IS FALSE
THEN ...

The numeric-type variable is not solved by a simple enumeration of the hypertext because calculation must occur after the input value is received. A simple and effective way to solve the problem is using such client-side script language[14] as JavaScript (Danesh, 1996) or VBScript (Jerke et al., 1997). The following is an example of those rules.

1 Script language is a programming language that directly operates while interpretation takes place by the interpreter, not the compiler. In the Web environment, there is a script language that interprets and operates in the Web server and a script language that interprets and operates in the Web browser. JavaScript and VBScript, which are the typical client-side script languages, operate via the interpret function within the Web browser and show the results in the Web browser.

```
IF total_income >= 0.2 * threshold
THEN pay_tax
IF total_income > 0.2 * threshold
THEN do_not_pay_tax
```

The variable "total_income" and "threshold" in the above two rules are numeric-type variables. HTML files using JavaScript as in Figure 5 can set up these two rules. As shown in Figure 5, the value of all variables occurring in the conditions set by the rules are received and inputted by the FORM tag, and the hyperlink, based on the evaluation of a numerical expression that includes the input variables, is set in the function "Branch" within the SCRIPT tag. Another advantage of this method is that the numerical expression of the conditions using JavaScript, one of the programming languages, can be any numerical expressions. This is shown in the third screen image in Figure 4 as an example of forming an inquiry screen for each variable.

```
<HTML>
<HEAD>
    <TITLE> Expression Type </TITLE>
    <SCRIPT LANGUAGE="JavaScript">
       <!--
           function Branch(form)
           {
               if (form.total_income.value >= 0.2 * form.threshold.value)
                   location="tax.html"
               else
                   location="notax.html"
           }
       //-->
    </SCRIPT>
</HEAD>
<BODY>
<FORM NAME="exprform">
Please input followings. <BR>
total_income: <INPUT TYPE="text" NAME="total_income"> <BR>
threshold: <INPUT TYPE="text" NAME="threshold"> <BR>
<INPUT TYPE="button" VALUE="OK" onClick="Branch(this.form)">
</FORM>
</BODY>
</HTML>
```

[Figure 5] Implementation of Numerical-type variables using JavaScript

Song et al. (2003) has shown that the hyperlink-based inference approach is between 9 and 225 times faster than conventional approaches. The hyperlink-based inference approach is fast without the usually associated burden on Web servers, does not need to use additional inference engines, and can be adopted by anyone who has elementary knowledge of HTML. Hence, the approach is a very simple and powerful one for Web-enabled inference.

3. The Management of Hypertension

For the management of hypertension, we use a guideline for hypertension management acquired from a cardiologist as the source of domain knowledge. The guideline is described in Table 2.

<Table 2> The Guideline for Hypertension Management

<Table 2> The Guideline for Hypertension Managemen

Condition		Management		
Symptom	Criteria	1st	2nd	Contra
Systolic hypertension	SBP ≥140 and DBP < 90	Diuretics Ca		
Elderly patients	Age > 70	Diuretics, Ca	β-blocker	
Heart failure	Chest PA = cardiomegaly and EF ≤ 50	Diuretics, ACE		β-blocker
LVH	LVH	β-blocker	ACE	α-blocker
Proteinuria	Protein(U/A) ≥1	ACE	Diuretics	
Diabetes	Glucose(AC) > 140	ACE	Ca	β-blocker
Angina	EKG = ischemia pattern	β-blocker	Ca	α-blocker
Hyperlipidemia	T.chol > 250	Ca	ACE α-blocker	β-blocker Diuretics
Peripheral vascular disease	Phx = VD or Complication = VD		Ca	β-blocker
Hyperkalaemia	K ≥ 5.5			ACE
Gout	Phx = Gout or Complication = Gout			Diuretics
No complication		Diuretics, β-blocker		

The first column is the name of the symptom, and the second column is the criteria to determine the symptom. The third column is the preferred method for the management of hypertension for each symptom, and the fourth column is the second best method. The contra method is the one that should not be taken. For example, the first line of the guideline says that:

(1) If the SBP value of a patient is greater than or equal to 140 and the DBP value is less than 90, the patient is exhibiting a symptom of systolic hypertension.

(2) If a patient exhibits the symptom of systolic hypertension, then the preferable methods of treatment are Diuretics and Ca.

However, a patient who suffers from hypertension may have several symptoms simultaneously. So, there are additional rules for the patient who has several symptoms simultaneously. The rules are summarized as follows:

(1) Single condition

If the patient has a single symptom, use the first or second methods as explained above. Given two effective methods, the choice is left to the doctor's discretion with respect to experience, and the patient's medical history and preferences.

(2) Complex condition

If the patient has several symptoms simultaneously, follow the next rules:

(2-1) In the case where there are common drugs among the first or second methods for the symptoms:

(i) If there are common drugs among the first methods, then use the common drugs.
(ii) If there are not common drugs among the first methods but there are common drugs among the first and second methods, then use the common drugs.
(iii) Otherwise, use all the drugs in the first and second methods except the drugs in the contra methods.

(2-2) In the case where there are no common drugs among the first or second methods for the symptoms:

If there are no common drugs among the first or second methods, use all the drugs in the first and second methods except the drugs in the contra methods.

(3) No complication

If the patient does not have any symptoms, use the first two drugs at the last line of the table: Diuretics and β-blocker.

4. Implementation of the Healthcare System for Hypertension

The domain knowledge explained in section 3 can be easily represented as rules for inference. For example, we can build a rule as follows for the case where a patient has a single symptom: systolic hypertension.

IF SBP $\geq$140 AND DBP < 90

AND not(Age > 70)

AND not(Chest PA = cardiomegaly and EF $\leq$ 50)

AND not(LVH IS TRUE)

AND not(Protein(U/A) $\geq$1)

AND not(Glucose(AC) > 140)

AND not(EKG = ischemia pattern)

AND not(T.chol > 250)

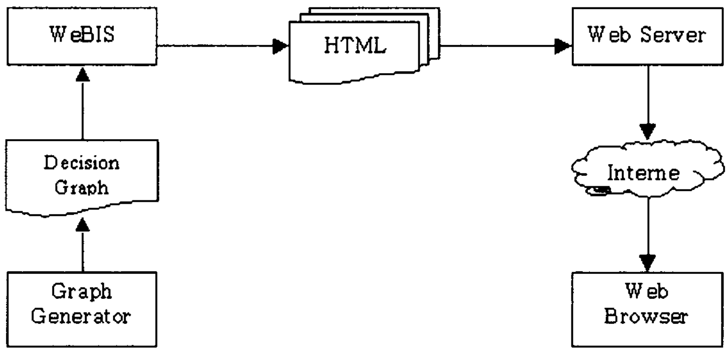
AND not(Phx = VD or Complication = VD)

AND not(K $\geq$ 5.5)

AND not(Phx = Gout or Complication = Gout)

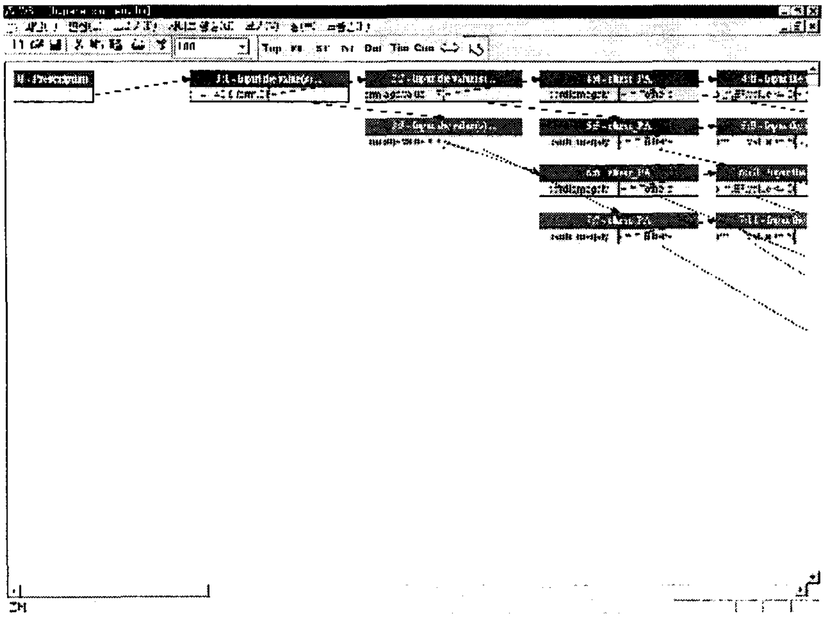
THEN

DISPLAY "1st method: Diuretics, Ca; 2nd method: -; Contra method: -"



[Figure 6] Overall Architecture of the System for Diagnosis and Indication of Hypertension

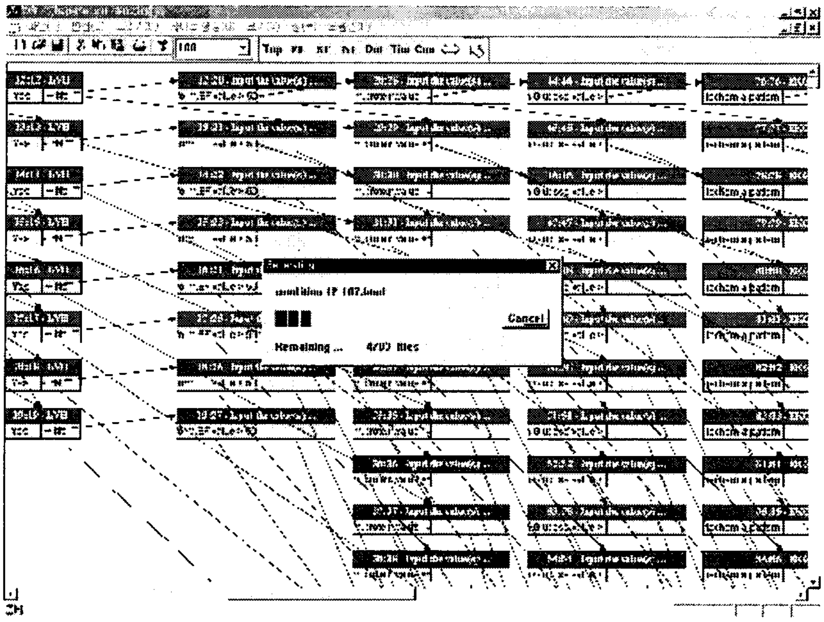
However, there are remaining problems from an implementation point of view. The following characteristics are required for the Web-enabled healthcare systems:



[Figure 7] Generated Decision Graph

- Accessibility: it is accessible from anywhere and at anytime
- Ease of use: user friendly interface
- Multimedia facilities: it supports multimedia
- Platform independency: it is independent of platform
- Fast response: response time should be short
- Stability: it is stable and free from system downtime

The first four characteristics - accessibility, ease of use, multimedia, and platform independency - can be achieved using the WWW. Almost all computers in the world today have Web browsers and are easily connected to the Internet. Web browsers are familiar to most users, so they are by definition easy to use, support multimedia contents, and are independent of platform. The remaining characteristics - fast response and stability - depend on Web servers. We have already compared the relative speed and stability of different structures of Web-enabled, rule-based systems. As we mentioned above, the hyperlink-based inference approach is the preferred approach if the rules could be represented by HTML files. Furthermore, as in the above example about the rule for systolic hypertension (represented by 11 HTML files for the input of answers to questions regarding the 11 symptoms, and one file for the display of a conclusion), the rules for the management of hypertension can be represented by HTML files.



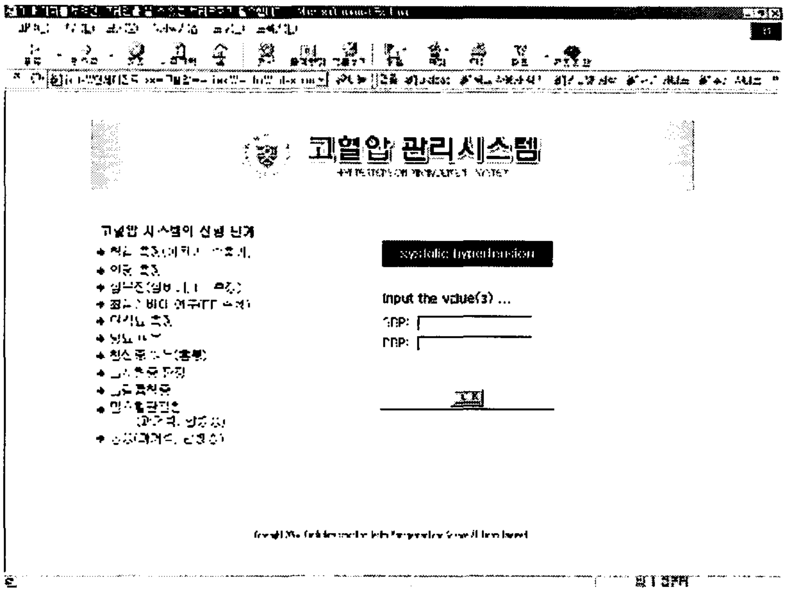
[Figure 8] WeBIS Generating HTML Files

One additional problem in implementing the system is the size of the rule base. Since, in the above example, there are 11 symptoms and a patient may have any combination of the 11 symptoms, the rule base consists of 2,048 (= 211) rules. Hence, the system would be made of 4,095 (= 212 - 1) HTML files including files to input answers to questions regarding the 11 symptoms, and the development and maintenance of the files would be a very time-consuming and tedious task. Therefore, the authors developed a tool that supports development and maintenance of HTML files for the management of hypertension.

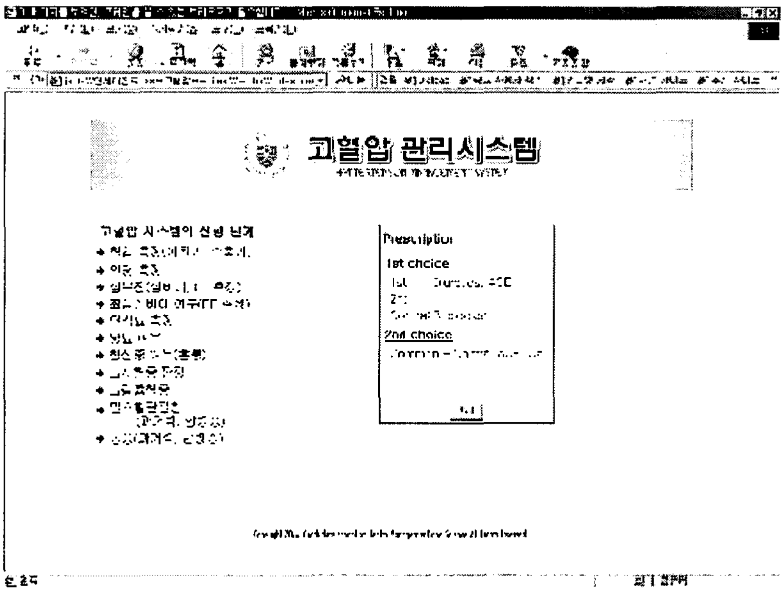
The overall structure of this new approach is depicted in Figure 6. In Figure 6, WeBIS generates HTML files from a decision graph, and the HTML files are delivered to Web browsers from Web servers via the Internet. Since the HTML files are hyperlinked to each other based on the decision graph, Web browser user queries and responses to questions serve the function of an inference engine, in this case as regards the management of hypertension.

The problem of development and maintenance of a large number of HTML files is solved by the graph generator in Figure 6. The graph generator generates the decision graph automatically, and the WeBIS, in turn, generates HTML files from the decision graph. To generate a decision graph, firstly the graph generator builds the Cartesian-products for the possible values (true or false) of 11 symptoms in Table 2. For the rule where a patient exhibits a single symptom: systolic hypertension, as an example, it generates a tuple as an element of the Cartesian-products: (true, false, false, false, false, false, false, false, false, false, false). Then, using the tuple, it draws 11 nodes for the 11 conditions (symptoms) and one node for the conclusion (treatment), and connects the true part of the first node to the second node and the false parts of the other 10 nodes to the corresponding next 10 nodes including the conclusion node. The other elements of the Cartesian-products are converted to the decision graph in the same manner while sharing the nodes for same conditions (symptoms). Figure 7 shows the decision graph generated by the graph generator, and Figure 8 shows the WeBIS generating HTML files.

After uploading to a Web server the HTML files generated by WeBIS, we can use the Internet to service the Web site for the management of hypertension. Figure 9 and Figure 10 show the screen images of the Web site. Figure 9 shows a screen to input an answer for the question on a symptom, and Figure 10 shows a screen for the suggested methods for the management of the patient's hypertension.



[Figure 9] Screen for Input



[Figure 10] Screen of Treatment

5. Conclusion

An expert system for the management of hypertension was implemented through a hyperlink-based inference approach. Hyperlink-based inference was executed using the hypertext capacity of HTML. Relevant HTML files, which were hyperlinked to each other based on the expert rules, were uploaded beforehand. The HTML documents were deployed and maintained automatically by the authors' proprietary tool called the Web-Based Inference System (WeBIS) that supports a graphical user interface (GUI) for the input and edit of decision graphs. Nevertheless, the editing task of the decision graph using the GUI tool remains a time consuming and tedious one when the knowledge engineer does it manually. Accordingly, this research implemented an automatic generator of the decision graph for the management of hypertension. The hyperlink-based inference approach ensures accessibility, multimedia facilities, fast response, stability, ease of use, and platform independency of the expert systems. Therefore, this research suggests a methodology for the development of Web-enabled healthcare systems using the hyperlink-based inference approach. This study also presents an example Web-enabled healthcare system for hypertension in order to evaluate the validity of our methodology. The example system demonstrates the desired factors of speed and stability, and serves as a model for future such systems.

References

Danesh, Arman, Teach Yourself JavaScript in a Week, Sams.net Publishing, 1996.

Dwight, Jeffry, Michael Erwin, and Robert Niles, Special Edition Using CGI, Second Edition, QUE, 1997.

Eriksson, Henrik, "Expert Systems as Knowledge Servers," IEEE Expert, pp. 14-19, June 1996.

EXSYS, Inc., Moving an EXSYS Application to the EXSYS Web Runtime Engine (WREN).

Far, Behrouz H. and Zenya Koono, "Ex-W-Pert System: A Web-Based Distributed Expert System for Groupware Design," Expert Systems With Applications, Vol. 11, No. 4, pp. 475-480, 1996.

Fielding, R., J. Gettys, J. C. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, Hypertext Transfer Protocol -- HTTP/1.1, INTERNET-DRAFT <draft-ietf-http-v11-spec-rev-05>, Internet Engineering Task Force, 11 September 1998.

Giarratano, Joseph and Gary Riley, Expert Systems: Principles and Programming, 2nd Edition, PWS Publishing Company, 1994.

"Java Expert Systems Tools," Intelligent Software Strategies, Summer 1997.

Jerke, N., M. Hatmaker, J. Anderson, VBScript Interactive Course, Waite Group Press, 1997.

Kim, P. C., "System for the Classification of Database Paths on the Internet (in Korean)," WWW 96-1, pp. 50-66, 1996.

Lee, J. K., et al., Development of an Expert System for Accepting Automobile Applicants (in Korean), Dongbu Insurance Co, November, 1995.

Lee, J. K., H. R. Choi, H. S. Kim, M. S. Suh, S. C. Chu, and W. C. Jhee, Principle and Development of Expert Systems (in Korean), Bobyoungsa, 1996A.

Lee, J. K., I. K. Lee, and H. R. Choi, "Automatic rule generation by the transformation of Expert's Diagram: LIFT," Int. J. Man-Machine Studies, 32, pp. 275-292, 1990.

Lee, J. K., Y. U. Song, S. B. Kwon, W. Kim, and M. Y. Kim, Development of Expert Systems Using UNIK (in Korean), Bobyoungsa, 1996B.

Lemay, Laura and Rogers Cadenhead, Teach Yourself JAVA 1.2 in 21 Days, SAMS Publishing, 1998.

Lim, K. K., J. Y. Kang, and J. K. Lee, "Structure and Analysis of Web-based Expert Systems (in Korean)," Proceedings of the '97 Fall Conference of the Korea Expert Systems Society, pp 63-73, 1997.

McCarthy, J. C., "The Social Impact of Electronic Commerce, " IEEE Communications Magazine, vol. 37, no. 9, 1999, pp. 53-57.

O'Leary, Daniel E., "The Internet, Intranets, and the AI Renaissance," IEEE Computer, January 1997, pp. 71-78.

Raggett, Dave, Arnaud Le Hors, and Ian Jacobs(ed.), HTML 4.0 Specification REC-html40-19980424, W3C, 24 April 1998.

"SELECTICA: Java-based Configuration for Internet and Electronic Commerce Applications," Intelligent Software Strategies, October 1996.

Song, Y. U., W. Kim, and J. S. Hong, "Web Enabled Expert Systems using Hyperlink-based Inference," Proceedings of the International Conference on Information and Knowledge Engineering (IKE'03), June 23-26, 2003, Las Vegas, Nevada, USA.

Song, Y. U. and J. K. Lee, "Automatic Generation of Web-based Expert Systems (in Korean)," Journal of Intelligent Information Systems, Vol.6, No.1, pp. 1-16, 2000.

Blaze Advisor URL --> http://www.blazesoft.com/

Expertise2go URL --> http://www.expertise2go.com/

ILOG Inc. URL --> http://www.ilog.com/

WAI Solutions URL --> http://www.waisolutions.com/