Chapter 9

Web-based pest and disease management in tea[+]

9.1. Introduction

Starting from the first generation expert systems which include DENDRAL, MYCIN etc., several commercial expert systems have been built in different application areas including medicine, industry, agriculture, science, military etc.

Despite the relative success of these systems, some problems are associated with the commercial expert systems [1]. These are (i) availability, (ii) software distribution, and (iii) communication. The problem associated with availability dictates that one has to put the expertise at the place and time where it is needed. The second problem is the software distribution. As knowledge base and user interface components of stand-alone applications are updated; those updates must be physically distributed to all users, along with necessary documentation and instructions. This can require many separate software installations and upgrades over time, often beyond the competence of users. Communication is the another problem. A lack of common protocols for the exchange of knowledge among expert systems has discouraged designs involving cooperation or dynamic information sharing among applications. The resulting isolation is a factor in the brittleness of expert system products, which tend to have a tightly circumscribed area of competence.

In India, tea is one of the major agricultural crops. The crop loss due to the attack of insect pests and diseases is a vital problem. For crop protection in tea, expert systems for insect pest and disease management is very essential and already we have developed and implemented such systems which include TEAPEST [2] for insect pest management and TEADISEASE [3] for disease management. Architecturally, all these applications in different domains are stand alone systems based on mainframe, minicomputer or personal computer platforms or LAN-based distributed applications.

In case of pest and disease management in tea, the human experts are very scarce commodity. Each and every tea garden can not provide human experts for pest and disease management. Moreover, tea is a wide spread cultivation. Most of the tea gardens are situated at remote places from both rural and urban establishments. So,

[+] This is based on the publications [The Journal of the Computer Society of India, Vol. 32, No. 1, 2002; Proc. of the National Seminar on Present impact and future prospects of Internet, 29-30 March, 2001, Varanasi, India and The 5th Int. Conf. (ICSC'2002) on System Simulation and Scientific Computing, Shanghai, China, Nov. 3-6, 2002 (accepted for presentation)] of the author.

physically a large number of tea gardens are not easily accessible. Particularly, in rainy season, some tea gardens remain detached from the rest of the country.

With the advent of World Wide Web and Internet technology, the above discussed problems can be solved and thereby the usage of expert systems can be enhanced as it is globally accessible. The Internet can offer a large potential for delivery of various information-based services, including the services of intelligent applications. As the availability of the Internet through dial-up network has grown in tea areas, it can be used as a medium for the delivery of expert systems. User friendly online expert systems would be very useful for insect pest and disease management in tea, so that one can get assistance from anywhere at any time. Here is an attempt to develop rule-based online expert systems for insect pest and disease management in tea. These expert systems can be accessed through URL http://samanta_rk.tripod.com/homepage.html.

In the next section, the benefits of using expert systems through the Internet has been discussed. Section 9.3 presents some representative expert systems on the net. Section 9.4 presents a discussion on the Web. In the next section (section 9.5) solution alternatives for connecting ES to web-sites(s) have been discussed. In section 9.6, the issues related to tools and languages available for internet-based expert systems have been discussed. A small discussion is provided on Perl in section 9.7. The interrelations between Perl, CGI, and the web are presented in section 9.8. Section 9.9 presents our web-based version of *TEAPEST* and *TEADISEASE*. Implementation issue has been presented in section 9.10. Section 9.11 presents the performance evaluation. Lastly, summary and conclusions are presented.

9.2. Benefits of using expert systems through the Internet

The number of Internet based expert system applications are gradually increasing due to some advantages. There are several factors that combine to make the Internet, by contrast to traditional platforms, a more effective base for expert system delivery. The Internet provides several advantages for expert system development [1].

- *The Internet is widely accessible.* Internet usage continues to grow rapidly with varying estimates putting the number of Internet users at around 100 million in the USA and 200 million worldwide (NUA Internet Surveys, 2000). Wireless communication makes Internet access completely portable as well. Due to the ubiquity of the Internet, Internet-based expert systems can be accessed literally anywhere in the world, at the precise location at which they are needed.

- *Web-browsers provide a common multimedia interface.* HTML-compatible browsers are installed on virtually all desktop workstations and personal

computers. The common controls and format directed by HTML provide a standard user interface platform on which developers can build. The presence of a standard user interface framework not only simplifies development efforts but also greatly reduces user training and supports requirements for expert system developers. Graphics, sound, text, etc. can be easily incorporated into user interfaces based on Web browsers when appropriate.

- *Several Internet-compatible tools for expert system development are available.* There are a variety of expert system development tools with features that facilitate Internet-based development, include server components, HTML interface, and compatibility with Internet protocols and languages. These include commercial development environments and free expert system shells.

- *Internet-based applications are inherently portable.* The Internet provides a development environment that is platform independent and widely available. For Internet-based expert systems, this means that there is no need for special distribution and installation of expert system software in advance of its use. Rather, applications are provided on demand at the time and place they are needed. Applications can easily be relocated, upgraded and modified without affecting users directly.

- *Emerging protocols support cooperation among expert systems.* Protocols such as CORBA, DCOM and Jini provide standard mechanisms for the exchange of information and services among applications. Though these protocols support data and service sharing through object interactions, they must still be adapted to the expert system application level in order to enable cooperation among intelligent processes.

9.3. Expert systems on the net

The content of this section is based on our web-searching results during 1st week of January, 2002 as well as on reference [1]. Although the list given here is not exhaustive, but readers may have some information on the recent advancements of internet based expert systems in different disciplines.

- **In Education and research**

TEST, an expert system for Thermodynamics, developed at San Diego State University [4], assists in solving problems in Thermodynamics. TEST is designed as an HTML-based decision tree that incorporates Java applets to perform calculations and what-if scenario.

- **In Business and industry**

The Coating Alternatives Guide [5] and Solvent Alternatives Guide [6] both developed at the Research Triangle Institute, have the goal of reducing industrial pollution. Both systems are rule-based expert systems, developed in a custom shell written in ColdFusion, a web application server framework.

- **In Medicine**

Starting from MYCIN, a number of expert systems have been developed with their relative merits and demerits. However, a list of 65 important medical expert systems along with their objectives is available from the reference [7]. But, however, they are mostly stand alone systems. We have been able to find out the following web-enabled systems.

HEPAXPERT/WWW is an internet-based interface to an expert system for interpreting serological tests for hepatitis infections [8]. Here test results are entered via the web, while processing is done off-line and conclusions are e-mailed to the user.

An internet-based system for intelligent management of the treatment of diabetic patients was designed by Riva et al. [9]. This application interacts with both patients and doctors as it collects data from patients and makes recommendations to doctors about therapy. The design includes fuzzy logic control, a LISP web server and client-side Java and Java Script components for performing calculations.

A decision support system to support the ear, nose and throat unit of a primary health centre is described by af Klercker and Klercker [10]. The system is designed as decision tree, derived by induction from actual past cases, and is implemented in Tripod CGI-PERL modules that dynamically generates HTML.

9.4. The Web (WWW)

The WWW or World Wide Web is one of the most popular services provided via the Internet. Probably the most appealing aspect of the Web, however, is the fact that it is not just for spectators. Once one has some experience with the Web authoring tools, he can publish himself and can offer over the Web anything he wants to make available.

The Web is the collection of all browsers, servers, files, and browser-accessible services available through the Internet. It was created in 1989 by Tim Berners-Lee. He designed the Web in such a way that documents located on one computer on the Internet could provide links to documents located on other computers on the Internet.

The most familiar elements of the Web is the browser. A browser is the user's window to the Web providing the capability to view Web documents and access Web-based services and applications. Today's Web browsers extend Mosaic's GUI features with multimedia capabilities and with browser programming languages. These programming languages make it possible to develop Web documents that are highly interactive. They do more than simple connection to another Web page elsewhere on the Internet. Web documents created in association with programming languages run entirely within the context of the Web pages that are currently displayed. This is a major advancement in Web publishing technology.

In order to publish a document on the Web, it must be made available to a Web server. Web servers retrieve Web documents in response to browser requests and forward the documents to the requesting browsers via the Internet. Web servers also provide gateways that enable browsers to access Web-related applications. Today's servers are designed for higher performance and to facilitate the development of complex Web applications. They also support the development of server based applications.

Because the Web uses the Internet as its communication medium, it must follow Internet communication protocols. The Internet's Transmission Control Protocol (TCP) and Internet Protocol (IP) enable World Wide Web connectivity between browsers and servers. In addition to using the TCP/IP protocols for communication across the Internet, the Web also uses its own protocol, called the Hyper Text Transfer Protocol (HTTP) for exchanges between browsers.

9.5. Connecting ES to Web-Site

9.5.1. Solution alternatives

Two different solutions for transference of system to Web had been conceived as:

Solution 1

Validating the existing system for its access through Web by using front-end technologies with the GUI truncated part of an existing stand-alone system (knowledge base, Inference engine, and Control strategy) at the back-end. This requires CGI or native interface from C++ (say) to an internet language.

The Steps

1) Truncation of existing GUI component of any existing stand-alone ES.

2) CGI interface development and legacy code modification for interface of ES on server side.
3) Web-based client side development for consultation with the server component.

Solution 2

This involves code conversion and implementation through the use of Internet programming languages and markup technologies.

The Steps

1) Database/ knowledge base conversion for Internet server based processing.
2) Entire code conversion to Internet.
3) Appropriate client design.

9.5.2. Technology requirements in transference

9.5.2.1. Web connectivity

Web connectivity of software system to make it accessible was prime concern. The language chosen needed to be capable to connect the program for access through Web. Interface through popular and standard Web-browsers using GUI capabilities is basis for better utility of ES over Web. Also, the system developed required to handle multi-user interaction and manage client users.

9.5.2.2. Database connectivity

Since case data, insect images, knowledge base are to be accessed and processed in the system, database connectivity is essential. Selection of only relevant part of data or KB through database queries is necessary parameter for increased efficiency of the ES.

9.5.2.3. Image processing and GUI

Some times images are very useful in identification task. So, a language with image processing capability should be useful. Also the GUI is the ideal way to communicate with the ES helps avoiding ambiguous dialogue.

9.5.2.4. Knowledge processing and inference

This capability is required as no ES based on knowledge base can be built without it. But at the same time efficiency of processing is important feature for an Internet based system.

9.5.2.5. Procedural

Knowledge base needs restructuring for database storage and access. This constraints the inference procedure. A convenient algorithm is needed to modify the facts and rules and then make inferences in working memory.

9.5.2.6. Hardware and setup

It is essential to investigate the hardware, environment and hosting and connectivity issues while transference to the Internet based system from an existing stand-alone system. The issues had been classified into hardware setup when deploying on server and environment factors when using it on Internet. These are listed in the table 9.1.

Table 9.1. Stand-alone vs. Web-based ES: Hardware and setup

Hardware and Setup	
PC based ES	Web Based Expert Information System (WBEIS)
Single PC	Dedicated Web server or server space and hosting.
Legacy software, C++ runtime, Database support	Software support / Java virtual machine/Database support
Operating system	Web server / Internet Operating System platform

9.6. Tools and languages available

Several tools and languages are available for developing Internet based applications. In general, these employ traditional expert system techniques and offer, in addition, the capacity for Web based development. Some of them are discussed below.

ExSys by Multilogic is an expert system shell incorporating rule based and fuzzy reasoning. An Internet based runtime component supports client-server designs through CGI scripts and can integrate other server-side components. Additional technical information is available at http://www.multilogic.com/solutions/.

Acquire, by Acquired Intelligence Inc., is an expert system shell that also includes a knowledge acquisition tool. Acquire's knowledge structures include qualitative influence graphs, decision tables and traditional rules. Inference can be both rule based and pattern based. The development of Internet based user inferences is supported through a client-server development kit that supports Java and ActivX controls. Additional information, including demos, is available at http://www.aiinc.ca

The Java Expert System Shell (Jess) is an expert system shell in Java that processes a CLIP-like rule based language. Jess includes backward chaining as well, and has many object-oriented features including a direct interface to Java components. Detail information is available at http://herzberg.ca.sandia.gov/jess.

XpertRule KBS is a rule based expert system shell that interfaces over the Web with a thin client using Microsoft's Active Server Page technology. Additional information is available at http://www.attar.com.

KB Agent by Explorer Reasoning Systems is an expert system shell based on the SOAR system of Allen Newell. It includes a CORBA interface for Internet based applications. More information and demos are available at http://www.ers.com

Perl offers language and scripting facilities. It can be used as programming vehicle to develop inference engines. The Web servers can communicate with Perl through CGI. The inference engine can be designed and implemented in an Unix/Linux environment by using Tripod CGI-Perl Module. Information can be obtained from http://www.tripod.com/

9.7. Perl

Perl is a powerful, easy to use and full featured programming language available today. Larry Wall, a linguist-turned-programmer built Perl to evolve over time as a language does. Traditional programming languages evolve slowly and at some point stop changing. Perl like a spoken language, evolves quickly to meet each new generation's needs. Perl stands for Practical Extraction and Report Language.

The distribution of a version of Perl, Perl 5 has always been freely available on the Internet and that distribution includes the source code. It helps anyone to modify the language to meet individual's needs and goals. Perl attracted attention of Unix system administrators who needed a language that was easier to use than the C programming language and more powerful than scripting languages such as Bourne and C-shell. Originally, most Perl users were Unix system administrators and other people with similar needs, who used Perl's text-processing power to generate reports and write scripts that aided in the configuration and monitoring of Unix systems. Unix programmers snapped up Perl as a tool of choice almost immediately for doing

various tasks because one has the ability to call most of the standard Unix system services from a Perl script including the internetworking functions.

Perl was designed to run on any computer, but because it usually ran on a Unix computer, it has a decidedly Unix flavor. But in the later half of the 1990's, applications and versions of Perl targeted towards Windows programming environments started appearing. In 1994, the World Wide Web through the Netscape browser became a new and powerful influence on the jobs of Unix system administrators. They turned the Perl as their tool to help them with their new World Wide Web tasks.

A Web page is a text document that is formatted with a set of commands called the Hyper Text Markup Language or HTML. HTML is a markup language, that is, it controls the way a document looks. HTML instructions tell a Web browser how it should go about displaying the page on screen. But HTML all by itself has practically no facilities for making a Web page do things.

The Perl programming language offers the most popular method of making a Web page 'do' something mainly because Perl is freely available and will run on every computer platform that can host a Web server [11]. Coupled with the Common Gateway Interface (CGI), Perl is used on the vast majority of Web sites to create Web pages that have to do more than sit there and look pretty. When new users of the Web wanted to create dynamic Web pages through CGI programming, they were generally working on a Unix Web server. Perl was freely available on those Unix Web servers and users started using Perl for their CGI applications. Because Perl was built to process text, and much of the CGI programming is processing user input and returning HTML text pages, Perl was a natural fit for this new programming environment. It is the de facto programming language for dynamic HTML Web pages. It is easy to use and there are thousands of free CGI, system administrative and text processing programs written in Perl available on the Internet.

Perl 5 is an interpreted language and so program is converted into machine format at the moment it runs, that machine format is more likely to be compatible with the machine it is running on. Perl programs run on a Unix, Windows, or Macintosh operating system with little or no change required to the code written. This feature is called portability. This portability is one of the primary features of Perl 5. Though most interpreted programs are slower than similar programs that are compiled, but Perl 5 is an extremely optimized and fast interpreted language. It is not much slower than compiled code. Another benefit of an interpreted language is that coding and testing process is much easier and faster.

Like shell scripts in Unix or batch and command files on MS-DOS and Windows NT, Perl programs are just text files that run through an application to process their commands. Moreover, Perl 5 runs on all of the brands of Unix such as HP-UX, Sun Solaris, Linux, Linux Red Hat etc. without modifications.

9.8. Perl , CGI and the Web

CGI or the Common Gateway Interface is the key to fit Perl programs into the World Wide Web. CGI is an interface and has been used for many years as a facility for passing information from a Web page to a program that can process the information. HTML is quite good in describing how a Web page should look in a browser, but the language all by itself has virtually no facilities for processing information or even making rudimentary decisions. CGI allows a program to take all its input from a page on the World Wide Web and do something with that input.

When a Perl program is requested to run from the command line, generally it takes its input from keyboard and sends its output at the screen. CGI reroutes those standard conventions. The Perl program's input comes from the Web page and most importantly, CGI sends Perl program's output back to the Web server. If the output happens to be formatted correctly in HTML, the server will put it out as an HTML document to whatever browser is connected to it. This is the foundation of using CGI as a pipeline between Perl and the Web.

A Web page can be created from a Perl program. Because Perl is a fully functional programming language, rather than a markup language such as HTML, one can decide what to draw within his program based on what has been entered in the page and sent to him. Of course, this facility is not limited to Perl. Any program written in any language can be interfaced with CGI. But, most of what one needs to do can be accomplished more easily from a Perl script than from a compiled program. Additionally, a Perl program might not have to be rewritten and recompiled when moved to another operating system or computer platform. Perl interpreters have been written for every popular computer platform, from Sun to Alpha to Apple to Intel and more and with very few exceptions the Perl programs should transport unchanged into every environment.

9.9. Web-based version of *TEAPEST* and *TEADISEASE*

Web-based *TEAPEST* or *TEAPEST/WWW* is a rule based online prototype expert system for insect pest management in tea. The system accepts observed damage symptoms on the tea bushes caused by various insect pest and the morphological findings of the insect pest(s) through HTML page(s). The inputs are then submitted to the inference engine in the server side. The inference engine identifies the insect pest(s) present in the field and sends output to the user. After submitting the field parameters such as humidity, temperature of the day, drainage, shade status, pesticides that previously used etc., one can get the control advice online.

TEADISEASE/WWW is the Web based implementation of *TEADISEASE* – a stand alone rule based expert system for disease management in tea. It is a prototype online

expert system. In real field practices, diseases of tea are mainly identified on the basis of damage symptoms they produced on the tea bushes. In *TEADISEASE/WWW*, HTML forms are designed to accept such observed damage symptoms as inputs. These inputs are then submitted to the inference engine in the server and after processing, the advice(s) is/are displayed online. The advice(s) of control measures include both chemical and cultural practices.

9.9.1. Insect pests of tea considered in *TEAPEST/WWW*

Tea is a perennial plantation crop spread over a large area. It provides favorable conditions for a variety of insect pests. Some of them are region specific. The species that make major damage have been discussed in details in **chapter 2**. In case of designing this prototype system, most damaging insect pests have been considered and presented with their group and taxonomic nomenclature which can be found in table 7.1 (**chapter 7**).

9.9.2. Diseases considered for *TEADISEASE/WWW*

The diseases of tea are rather numerous. They attack different parts of the plants such as leaves, stems and roots as described in details in **chapter 2**. The nursery plants are not being spared. To design this prototype expert system, the diseases that severely damage the crop have been considered and presented with group and names which can be found in table 8.1 (**chapter 8**).

9.9.3. Knowledge engineering

Acquisition and representation of knowledge are vital tasks of building expert systems. The knowledge used to design these two Web based expert systems *TEAPEST/WWW* and *TEADISEASE/WWW* was acquired when their stand alone versions were developed. The knowledge had been acquired through different sources (i) human experts, (ii) published literatures, and (iii) field data observation and collection. As a first source of knowledge, multiple competent and experienced human experts, having 15-25 years of experience had been selected. The experts had been consulted through structured interviews. Their judgements for different sets of possible observations were recorded in structured forms.

Tea Research Association of India has a long history of research in tea. As a second source, various literatures, relevant books and workshop reports published by them were used to elicit knowledge [12 -27].

For real field study, regular visits were conducted to the pest and disease affected sections of various tea estates of North Bengal districts of West Bengal, India in different seasons. All real field observations were structured in such a way that the HTML forms for input could be effectively designed. The inference engines in the

server side have been designed using Perl programming language, where rule based representation is more suitable. So the acquired knowledge is represented in rule based form for later implementation.

9.9.4. Architecture of *TEAPEST/WWW*

The system is designed to aid the decision making process for identification of insect pest(s) of tea and the control measures (selection of chemical mitecides/pesticides) through WWW. The block diagram of the system is shown in fig.9.1.

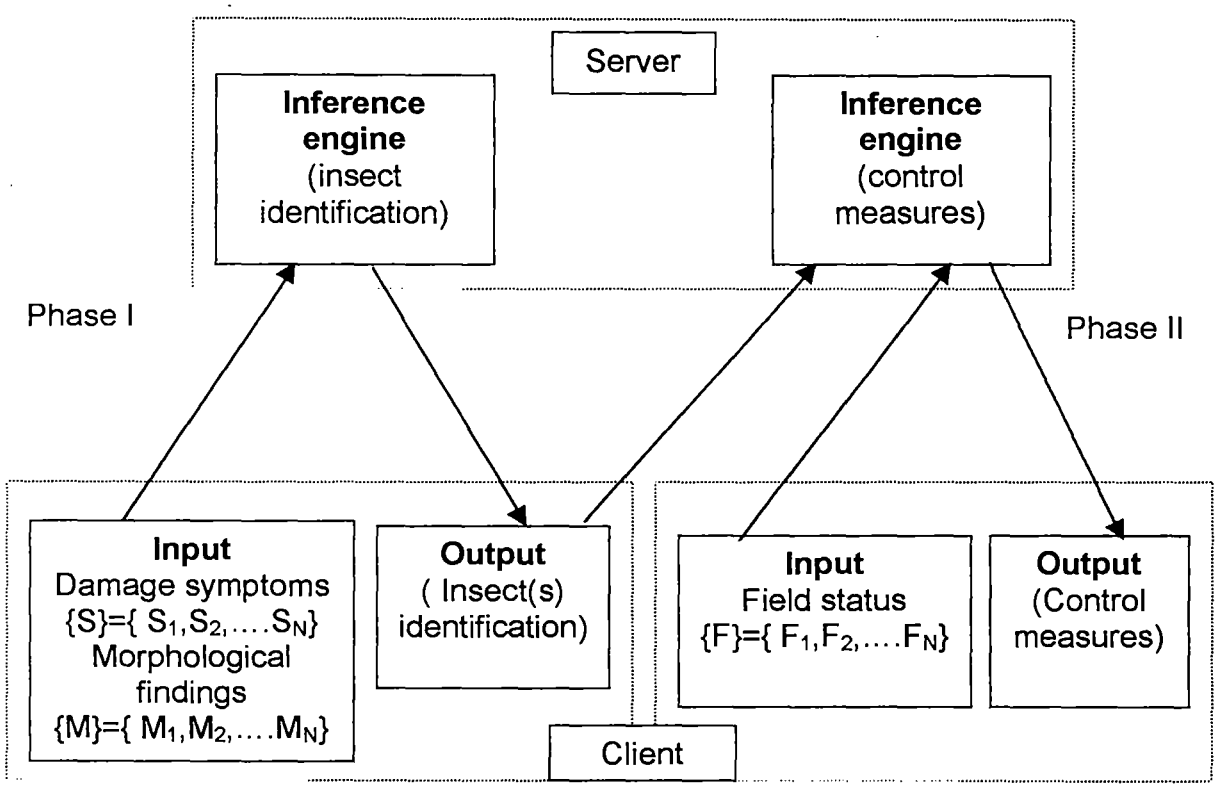

Fig. 9.1. Block diagram of *TEAPEST/WWW*.

During the first phase, preliminary identification of active insect pest in a field can be done on the basis of plant damage symptoms observed in the field. This is further confirmed with the knowledge related to the morphological characteristics such as length, body color, special identity etc which the users should have carefully watched during field visit.

TEAPEST/WWW uses domain knowledge in this phase. During this phase, the system asks for inputs related to the damage symptoms caused by insect pests to the different parts of the plants such as buds, leaves, stems etc and the morphological findings of insect (if any), through HTML page(s). The inputs are then coded to environmental variables. When submitted, the environmental variables become available to the inference engine at server for decision making tasks.

In this phase, the production rules of inference are used and a good level of accuracy is achieved in results of identification. The reasoning is performed by forward chaining with this rule based knowledge.

' IF {S} THEN {P} CF' rules have been used for inferences, where {S} is the plant damage symptoms, {P} is the insect pest and CF is the certainty (confidence) factor associated with the rule. The certainty factor CF attached to every rule represents the truth membership (confidence) of that rule. The value assigned to CF ranges from 0 to 100. The higher the value of CF, higher is the confidence associated with that consequent. For each rule, the value of CF has been obtained from multiple experts during knowledge acquisition and finally modified after test run with real field data.

The format chosen for definition of rules allows flexibility in structuring the knowledge. An antecedent of any rule may be a composite of a number of clauses or atomic propositions connected via the logical operations .AND. (&&) and .OR.(||). Examples of decision making rules are:

Rule (rs 5)

```
if ($qs{'maturedleaf'} eq "true" && $qs{'leafsurface'} eq "upper" &&
($qs{'leafcolour'} eq "Bronze" || $qs{'leafcolour'} eq "Brown") &&
($qs{'reddish'} eq "true" || $qs{'reddots'} eq "true" ||
$qs{'redpatch'} eq "true") && ($qs{'dryup'} eq "true" ||
$qs{'deformed'} eq "true") && ($qs{'sick'} eq "true" ||
$qs{'defoliation'} eq "true"))
   { $insect[1] = "Red Spider"; $cf[1]=95; }
```

.
.
.

Rule (si 3)

```
if ($qs{'maturedleaf'} eq "true" && $qs{'crinckled'} eq "true" &&
($qs{'leafcolour'} eq "Yellow" || $qs{'leafcolour'} eq "Brown") &&
($qs{'chlorosis'} eq "true" || $qs{'dieback'} eq "true" ||
$qs{'newflush'} eq "true" || $qs{'sick'} eq "true" ||
$qs{'tendershoot'} eq "true" ))
{ $insect[22]= "Scale Insects "; $cf[22]=80;}
```

.
and so on.

As inference propagates from damage symptoms (starting with inputs) to the identification of insect pests(goals), forward chaining of rules had been found suitable this case [28]. Phase I consists of 111 rules. A threshold value (CF >20) was set for meaningful results as suggested by the experts.

After identification of the insect pest(s), in Phase-II, the system asks for inputs related to the field observations (crop status, weather conditions, previous mitecide/pesticide

used etc) and suggests the chemical mitecides and/or pesticides according to the priority. The priority has been given first to select the chemicals exempted by various international agencies and then at per the higher order of ppm (part per million) values of MRL allowed by them. Toxicity and environmental persistence of chemicals have also been considered for optimal solution. In case of multiple appearance of the pests, the system suggests cocktail application of chemicals.

Suggesting suitable control measures for insect pests is again a decision activity. 'IF {P} AND {F} THEN {C} CF' rules have been used for inferences, where {P} is the insect pest, {F} is the field observations and {C} is the chemical mitecide / pesticide. Phase II contains 63 rules.

9.9.5. Architecture of *TEADISEASE/WWW*

As TEADISEASE/WWW is a Web accessible expert system for disease management in tea. Client-server technology has been used in designing the system. The block diagram of the system is shown in fig. 9.2.

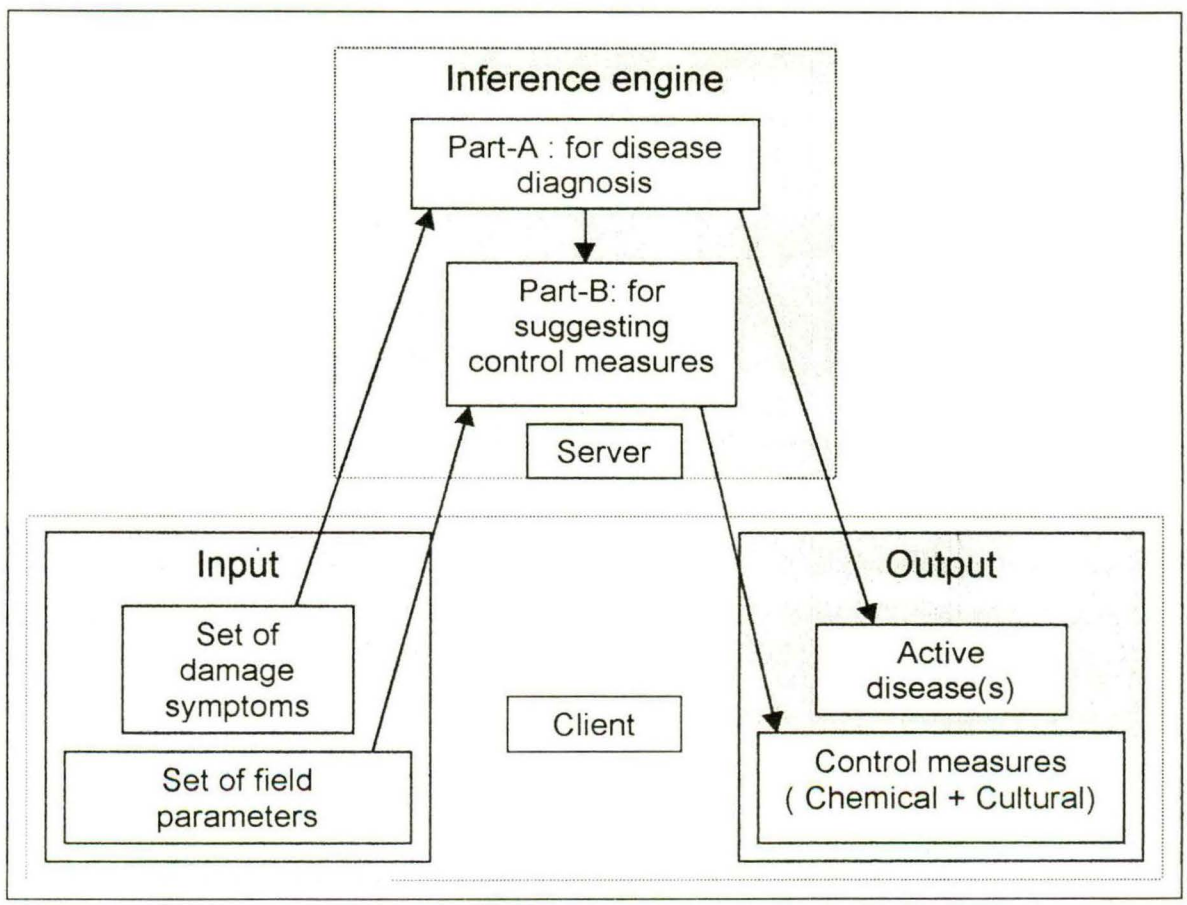

Fig. 9.2. Block diagram of *TEADISEASE/WWW*.

In real field cases, tea diseases are diagnosed on the basis of observed damage symptoms on different parts of the bush, caused by them. Any decision regarding

control measures depends on the nature of disease and the field conditions at that time. Field conditions include age of the bush, soil conditions, temperature of the day, humidity, rainfall etc. Thus two sets of inputs are essential for the system (i) set of damage symptoms, and (ii) set of field parameters. To avoid complication and interaction time delay, the system has been designed to start with these two sets of inputs. Domain knowledge has been used to design the input forms.

Users are asked to input the damage symptoms and field parameters. The inputs are then transformed in codes and submitted to the inference engine in the server. Some examples of codes are shown in table 9.2.

Table 9.2. Examples of codes of the inputs for *TEADISEASE/WWW*.

Observations	**Codes**
Bush appearance:	
1. Apparently normal	1. ba1
2. Completely dead	2. ba2
. . . . .	. . . . .
. . . . .	. . . . .
Leaf appearance:	
1. Curl or bended over	1. la1
2. Withered leaves remain attached	2. la2
3. Green leaves drop	3. la3
. . . . .	. . . . .
. . . . .	. . . . .
Observations on upper leaf surface:	
1. Yellowish spots	1. uls1
2. Yellowish glistering spots	2. uls2
3. Brown spots	3. uls3
. . . . .	. . . . .
. . . . .	. . . . .
. . . and so on	

The inference engine consists of two parts, Part-A and Part-B. Part-A is for diagnosis of the disease(s) and Part-B is for suggesting control measures. Initially, depending upon the submitted damage symptoms, Part-A diagnoses the active disease(s) in the field. Suggesting control measures is also a decision activity performed in Part-B. On the basis of identified disease(s) and submitted field observations, Part-B suggests the control measures. The control measures include both chemical and cultural practices.

Production rules have been used to design the inference engine. The reasoning is performed by forward chaining with rule based knowledge. For each part of the inference engine, 'if <antecedent> (then) < consequent> CF' type production rules have been used for inferences as in the case of *TEAPEST/WWW*. The certainty factor CF attached to every rule represents the truth membership (confidence) of that rule. The value assigned to CF ranges from 0 to 100. The higher the value of CF, higher is the confidence associated with that consequent. For each rule, the value of CF has been obtained from multiple experts during knowledge acquisition and finally modified after test run with real field data. The system contains 217 rules. A threshold value (CF>20) has been set for meaningful results as suggested by the experts. The examples of rules using codes are:

.

.

Rule (Charcoal root rot 2)

```
if (($qs{'ba2'} eq "true" || $qs{'ba3'} eq "true") && $qs{'la2'} eq "true" && $qs{'cf12'} eq
"true" && ($qs{'rf5'} eq "true" || $qs{'rf6'} eq "true") && ($qs{'rbp3'} eq "true" ||
$qs{'rck1'} eq "true"))
{ $disease[11] = "Charcoal root rot";  $cf[11]    = 90;}
```

.

.

Rule (Black rot 14)

```
if (($qs{'la2'} eq "true" || $qs{'la3'} eq "true") && (($qs{'uls5'} eq "true" || $qs{'uls7'} eq
"true" || $qs{'uls9'} eq "true" || $qs{'uls11'} eq "true" || $qs{'uls12'} eq "true") &&
($qs{'uls13'} eq "true")) && ($qs{'lls3'} eq "true" || $qs{'lls4'} eq "true" || $qs{'lls5'} eq
"true") && ($qs{'lls9'} eq "true" || $qs{'lls10'} eq "true" || $qs{'lls11'} eq "true"))
 { $disease[1] = "Black Rot";  $cf[1]    = 95;  }
```

.

.

and so on.

In case of selection of chemicals, as in the case of *TEAPEST/WWW*, priority has been given first to select the chemicals exempted by various international agencies and then at per the higher order of ppm (part per million) values of MRL allowed by them. Toxicity and environmental persistence of chemicals have also been considered for optimal solution. Cultural practices are also very effective against diseases, so along with the chemical application, the system suggests the proper cultural practices.

9.10. Implementation

TEAPEST/WWW and *TEADISEASE/WWW* are both Web based online expert systems. To interact with the user, the systems have been designed to use HTML pages as user interface. The inputs are accepted through HTML pages as environment variables. When submitted, the HTTP request goes to the Web server and the variables are made available to the inference engine through CGI. The inference engine sends the output back to the Web server and the server sends the output back to the user through HTML page(s). To design inference engine, Perl has been accepted as the preferred programming vehicle. The inference engines have been implemented in an Unix/Linux based environment by using Tripod Perl Module. A simple block diagram will be self explanatory as shown in fig. 9.3.

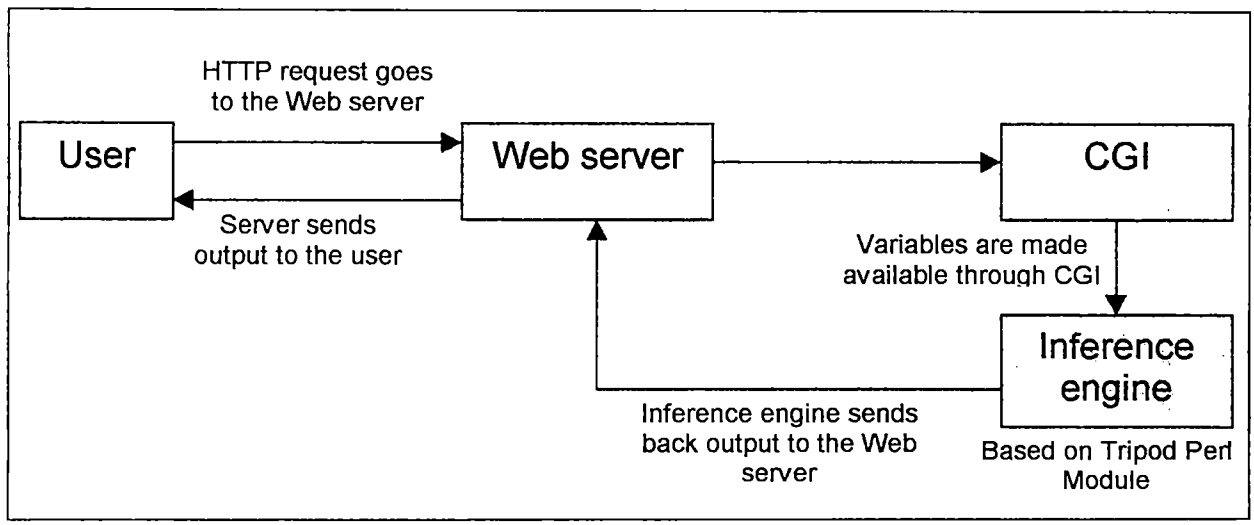

Fig. 9.3. Block diagram for implementation process of *TEAPEST/WWW* and *TEADISEASE/WWW*.

9.11. Performance evaluation

TEAPEST/WWW and *TEADISEASE/WWW* are under field testing since last one year with real field cases of tea estates of North Bengal districts of West Bengal, India. For each case, the system's output has been compared with both the project expert's judgment and as well as real field observations. It produces results of equally good quality and precision as in the case of field experts. For better validation of the system, the expert's judgment/field observations versus *TEAPEST/WWW* and *TEADISEASE/WWW* recommendations have been recorded for a large number of real field cases. The agreement is more than 85%. As tea is a wide spread industry, the experts from other corners of the country are requested to inform us through e-mail about the cases for which the performance of the system is not satisfactory so that the knowledge base can be updated.

9.12. Summary and conclusions

TEAPEST/WWW and *TEADISEASE/WWW* are the interactive web-based expert systems developed to meet up the needs of the tea industry. Suitable GUI components incorporated in the input forms through Web pages provide facility to select and deselect multiple options from a menu. They are user friendly systems and need almost no training for their use. Linguistic variable inputs and outputs which are in natural language and commonly used terms add advantages to a layman or a lesser trained person seeking expert guidance for proper insect pest and disease management practices in tea. *TEAPEST/WWW* and *TEADISEASE/WWW* show good performance within our periphery. But the input forms are designed on the basis of domain knowledge. As tea is an wide spread cultivation in different climatic conditions and these systems are accessed globally, the user observations may have to be extended beyond the scope of input forms and knowledge bases in any particular case. The systems leave scope for further refinement in this direction before launched to be full scale systems.

References

1. Ralph Grove. Internet-based expert systems. Expert Systems, 17(3), 2000. pp. 129-135.

2. Indrajit Ghosh and R. K. Samanta. A Knowledge Based Consultation System for Insect Pest Management in Tea Crop. Proc. International Conference On Modeling and Simulation, MS'2000, Cairo, Egypt, 2000. pp. 43-48.

3. Indrajit Ghosh and R. K. Samanta. *TEADISEASE*: A Rule Based Object-Oriented Expert System for Disease Management in Tea (Communicated). 2002.

4. S. Bhattacharjee. The Expert System for Thermodynamics. http://eng.sdsu.edu/testcenter/ (accessed January, 2002).

5. Research Triangle Institute. Coating Alternatives Guide. http://cage.rti.org (accessed January, 2002).

6. Research Triangle Institute. Solvent Alternatives Guide, http://sage.rti.org (accessed January, 2002)

7. http://www.computer.privateweb.at/judith (accessed January, 2002).

8. C. Chizzali-Bonfadin, K. P. Adlassnig, M. Kreihsl, A. Hatvan and W. A. Horak. A WWW- accessible knowledge base for the interpretation of hepatitis serologic test. Int. J. Med. Informatics, 47(1-2), 1997. pp. 57-60.

9. A. Riva, R. Bellazzi, and M. Stefanelli. A Web-based system for the intelligent management of diabetic patients. M. D. Computing, 14(5), 1997. pp. 360-364.

10. T. af Klercker, and M. af Klercker. Decision support system for primary health care in an Inter/Internet environment. Computer Methods and Programs in Biomedicine, 55(1), 1998. pp. 31-37.

11. Perl, CGI and JavaScript Complete. Indian Edition. Sybex Inc. USA. 2000.

12. B. Banerjee. Tea Production and Processing. Oxford & IBH Publishing Co. Pvt. Ltd., New Delhi. 1993.

13. Pests of Tea in North-East India and Their Control, Memorandum No-27, 2nd ed., Tea Research Association, Jorhat, India. 1994.

14. B. A. Baptist and D. J. W. Ranaeera. The scarlet mite of the genus Brevipalpus as pests of tea in Ceylon. Tea Quarterly, Vol. 26, 1955. pp. 127-136.

15. G. M. Das and N. S. Sengupta. Biology and control of the purple mite, Calacarus carinatus (Green), a pest of tea in North-East India. Jour. Zool. Soc. India, Vol. 14 , 1962. pp. 64-72.

16. S. C. Das and N. G. Goswami. A new pest of tea, *Euproctis subnotata* Walk. Two and a Bud, Vol. 24, No 2, 1977. pp. 44.

17. B. C. Phukan. Towards Attaining Sustainable Productivity, Workshop Report, Tea Research Association, Jorhat, India. 1996.

18. S. Sannigrahi, Karan Shingh and B. C. Barbora. Neem Based Pesticides and Their Use against Tea Pest. Two and a Bud, Vol. 42, No. 1. 1995. pp. 7-11.

19. B. Banerjee. An Introduction to Agricultural Acarology. Associated Publishing Co., New Delhi. 1988.

20. K. C. Sarmah and P. M. Glover. Blister Blight. Tea Encyclopedia, Tea Research Association, Serial No. 62/3, 1993. pp. 2.

21. K. C. Sarmah and P. M. Glover. Control of Primary Root Diseases of Tea. Tea Encyclopedia, Tea Research Association of India, Serial No. 70/1, 1989. pp. 5.

22. A. Agnihothrudu. A world list of fungi reported on tea. *J.* Madras Univ. Section B, 35, 1964. pp. 155-171.

23. K. C. Sarmah. Diseases of tea and associated crops in India. Indian Tea Association of India, Memorandum No. 26. 1960.

24. Newsletter, "Two & A Bud", Tea Research Association of India, Vol. 22, No. 2, 1975. pp. 85-87.

25. Integrated Pest and Disease Management in Tea. Workshop Report, Tea Research Association of India. 1998.

26. B. Subramaniam. Tea in India, Associated Publishing Co., New Delhi. 1995.

27. K. C. Sarmah and P. M. Glover. Thread Blight. Tea Encyclopedia, Tea Research Association, Serial No. 42/2, 1971. pp. 2.

28. E. Rich and K. Knight. Artificial Intelligence, McGraw Hill Inc., 1991. pp. 621.