

# An Agent Based Approach to Expert System Explanation

**Samhaa R. El-Beltagy & Ahmed Rafea**  
Central Lab for Agricultural Expert Systems  
Agricultural Research Center  
Ministry of Agriculture and Land Reclamation.  
El-Nour St. Dokki 12311, Giza Egypt  
Cairo, Egypt  
{samhaa, rafea,}@esic.claes.sci.eg

**Ahmed Sameh**  
Department of Computer Science  
The American University in Cairo  
113 Kasr Al Aini Street,  
P.O.Box 2511,  
Cairo, Egypt  
sameh@acs.auc.eun.eg

## Abstract

During the last few years, expert system explanation has become an active research area after recognizing that its role goes beyond expert system verification as assumed by early systems. After examining current systems, it has been found that only a few of these meet the requirements of an end user interested in learning more about the domain being addressed by an expert system and understanding why it has reached a certain conclusion. It was also found that systems that address these requirements, do so at a very high cost, since they embed immense amounts of knowledge in a system without providing any means for accessing this knowledge except by the system for which it was built. The primary goal of this paper is to investigate the use of an agent based approach for the explanation problem, such that knowledge re-usability would be promoted and high quality explanations generated. Through the implementation of an experimental prototype, the approach presented was found to show great promise since it satisfied the addressed explanation goals, achieved knowledge re-usability, and modularity. The devised architecture was also found to be scaleable and open, and to promote parallelism.

## Introduction

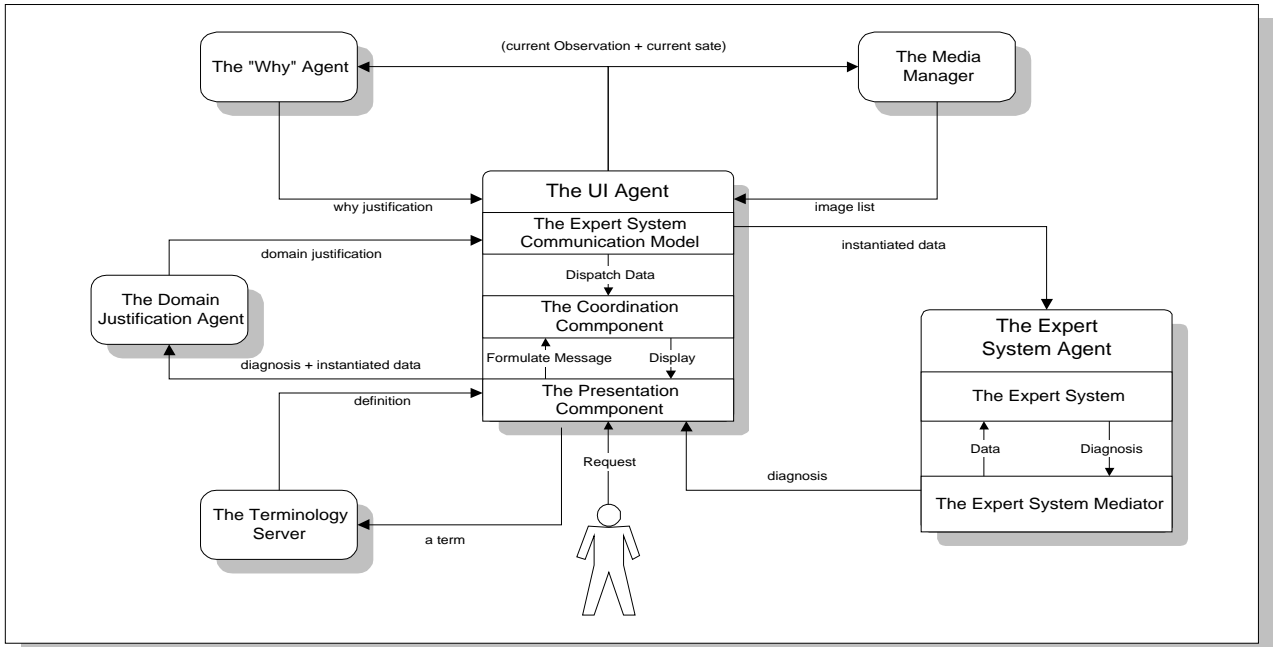
Ever since the inception of expert systems, explanation has played a significant role in their construction and usage. However, developments in the expert system field has resulted in greater interest in explanation. The result is that explanation acquired new dimensions and roles and thus came to mean different things to different people. One of the new roles identified for explanation is that of a tool capable of addressing the needs of an end user interested in learning more about the domain being addressed and obtaining explanations using straight to the point, understandable domain oriented knowledge rather than lengthy, uninformative problem solving knowledge. Very few exiting explanation models address these requirements (Swartout, Paris, and Moore, 1992; Ford et al 1996). Such systems either embed immense

amounts of knowledge into an expert system without providing means for sharing or re-using that knowledge, or build a separate explanation system that can only be used by a given expert system. This makes the process of building an explanation module, rather expensive.

The motivation behind this work lies in the realization that the required level of explanation signifies a clear deviation from an expert system's goal. Explanation on that level, requires deep knowledge and hence knowledge representation forms and reasoning strategies that are different than those required by the expert system. By distributing the explanation problem among a number of independent agents various advantages are capitalized among which is knowledge shareability and reuse. Thus, through a focus on diagnostic types of problems, this work will show that the use of an agent based approach offers a "natural" solution to the explanation problem.

## Agent Task distribution

Explanation is generally needed in two different phases. The first of these is the dialogue phase, in which the system interacts with the user in order to collect observations/symptoms regarding a problem. During that phase, the system is in total control of the dialog and the user must fully understand the meaning of questions presented to him/her, otherwise, he/she might answer questions incorrectly and hence the system might reach an incorrect conclusion. Thus, understandability in this phase directly affects problem solving. Explanation that can provide definitions and illustrate the relevance of a question, must be provided in this phase. The capability of going in depth such that the user can continue to probe related items in the explanation, until he/she is satisfied, must also be supported. In that way, the user fully understands the question, and at the same time learns more about the domain for which the expert system was constructed. The second phase is the one that follows problem solving. In this phase, the user may want to understand the domain justification for the given



**Figure 1: A simplified Diagram of Overall Agent Interaction**

conclusion. Just as with the first phase, the user might need more clarification on aspects of the explanation.

Through the analysis of the problem at hand, the identification of logical groupings of knowledge and tasks is possible. The first of these tasks is a simple one and consists of providing knowledge about domain terminology and abstractions, as well as global domain information on the various terms such that queries in the form of "Tell me more about X" can be satisfied. The second task, is one that involves maintaining knowledge about how different domain concepts contribute to the diagnostic problem solving process such that questions in the form of: "Why are you asking me about X?" could be answered. The third task, involves the intelligent management of various media types. The fourth and final task entails the construction of a justification "story" after a diagnosis has been reached. Together, these four tasks solve the explanation problem from a domain oriented perspective. Each of these tasks could be assigned a separate agent due to their independence in terms of problem solving and knowledge representation. The agents will thus be named as follows: a terminology server, a "Why" agent, a media management agent and a domain justification agent. In addition to the explanation agents, there are a number of other agents that must exist within this architecture. Among these, is of course the expert system "agent" which is responsible for producing diagnosis based on a set of input observations. Another, is the a facilitator with which the other agents register and advertise their capabilities. Finally, there is user interface

agent which plays a central role within this architecture. In the proposed framework, the user interface agent, with collaboration with the facilitator, acts as a communication coordinator between the various agents, as well as an intelligent data collector. Figure 1 shows a simplified diagram of overall system interaction.

Another important issue in the design of the proposed architecture, is that of achieving agent interoperability. For that purpose the use of the Knowledge query and manipulation language(KQML) is suggested (Finin et al, 1994)

### The User Interface Agent

The UI agent is composed of three different components: the expert system communication model, the presentation component, and the coordination component. The expert system communication model is based on the Common-KADS communication model. In the Common-KADS methodology to KBS development, it has been realized that there should be a separation of concerns between the actual KBS problem solving component, and human computer interaction(HCI) activities (Schreiber, Wielinga, and Breuker 1993). Thus, the communication model was suggested as an entity that would handle all HCI. The underlying assumption behind this architecture is that many expert systems could be separated into two parts: the problem solving component and an interface or a front end component. Since this architecture addresses diagnostic types of problems, a communication model based on hierarchical classification(HC) is

suggested (Chandrasekran 1996). Hierarchical classification, is a problem solving method identified for solving diagnostic types of problems as part of the Generic Task approach to expert system development. Within the proposed framework, knowledge components for which input is desired, should be classified, grouped and organized just as in hierarchical classification. However, in this case, no actual problem solving should take place. Since nodes(hypotheses) in this hierarchy are composed of a number of observations, the evaluation of the tree nodes, will only serve to intelligently handle the dialog between the user and the system by deducing observations for which to ask the user based on knowledge of the problem addressed. The process of building this hierarchy is a fairly simple one, given that dependencies and relations between data input items, are known.

The presentation component is that concerned with the display of the interface components.

The coordination component acts as the interface between the UI and the other agents. Basically, it is responsible for structuring messages using the agreed upon agent communication language(ACL) and interchange protocol, and handling message exchange. The coordination component is not specific to the user interface agent, but is a constituent of all agents to be described.

### **The Terminology Server**

The main responsibilities of the terminology server, are providing terminology definitions and abstractions, as well as detailed term knowledge within a given predefined domain. Descriptors for this agent include the ontology that it uses, as well as the domain that it addresses.

### **The “Why” Agent**

The “Why” agent is responsible for justifying the question asking strategies using familiar domain oriented knowledge. From observations of human behavior during a diagnostic process, it was noticed that the questions they ask fall under two classes: general questions that serve to eliminate or indicate the possibility of certain disorders or disorder classes, or very specific questions that relate to particular disorders. When explaining why a given question is being asked, a human expert might also indicate a dependency relationship between an already asked observation and the current one. This indicates that for each observation it is possible to understand the reason it is being questioned in light of questions/observations already asked and knowledge of the strategy employed. Further more, it means that the reason a certain

observation X is being questioned, can not be separated from the overall context of the system which includes its state, and the followed problem solving approach. One approach to this problem would entail following a similar technique to that adopted by participatory explanation (Ford et al. 1996). In this approach, a knowledge engineer uses knowledge obtained during knowledge acquisition sessions about “why” a given question is to be asked on an abstract and detailed level, and knowledge about dependencies between the various observations to build the “why” agent’s knowledge base(KB). In this case, the KB represents a documentation of knowledge that is implicit within the expert system and that would otherwise be lost. When answering a ‘why’ type of question, the “Why” agent must receive a complete account of the system’s state in the form of instantiated data as well as the observation that is being questioned.

### **The Media Manager**

Most current knowledge based systems(KBS) provide facilitates for handling media elements such as images, sound, and graphics. There are two approaches for handling media elements, the first of these integrates the elements into the KBS, while the second, builds a separate system to handle all media elements. Unfortunately, the majority of the systems that integrate media elements into a KBS, fail to do so intelligently. These systems use techniques that augment observations or data items, with media types without taking into account previous inputs to the system. For example, for the question “*Are there any pustules on the leaves?*”, a KBS might simply present an image that contains a sample of pustules irrespective of the information a user had previously entered such as the growth stage, the leaf color, the leaf state etc. In many cases, this confuses the user more than it clarifies a question. Since an explanation system must present media items that serve to clarify observations, the proposed architecture employs the technique that handles media elements separately from the KBS through the employment of a media manager component. The media manager takes as input the current state of the system. It then returns URLs of media elements that best match the state. It also returns a complete list of features that accurately describe each of the returned media elements. In case the returned media element does not completely match the user’s inputs, the user is made aware of the differences.

Although media items will be described by a set of features, it is important to note that not all these features will be available as input at a given point of time. To clarify, consider the question about pustules presented above. Within the image database, some of the images

that satisfy this category are described by pustule variations (brown pustules, orange pustules, etc.) as well as other features. So, there will be no exact match to the user's query. The system should be intelligent enough to specialize from generalizations (yellow pustules are a specialization of pustules) and return a complete list of specializations. This could be done in one of two ways, the first of these is explicitly building a specialization hierarchy and indicating where each observation fits within this hierarchy, keeping in mind that an observation could fall under multiple generalizations. In case an exact match is not found, this hierarchy could then be used to specialize the query. Another more simplistic way, is filling unknown values with don't care conditions before sending the image query to the server. The media manager then should attempt to find images that best fit this input using a relatively simple search engine.

### **The Domain Oriented Justification agent**

The process of problem solving in diagnostic systems involves attempting to reduce the search space by questioning general observations in order to determine which disorders might exist and hence which specific observations to ask about. When a user asks for a justification of a conclusion in terms of domain facts, he/she is usually not interested in the details employed by the problem solver to reduce the search space, but rather how a conclusion is justified in light of inputs directly related to that conclusion. So, within the proposed architecture, justifications generated, represent "straight to the point" explanations based on knowledge of the various disorders, their primary symptoms and supporting symptoms. The domain oriented justification agent is assigned the task of justifying a conclusion in terms of the user's inputs and the expert system's conclusion. The main issue of concern in the construction of this agent is devising a method for generating a "plausible story" from the given inputs that justify the conclusion. Since the goal and the audience addressed do not require fidelity, the explanation story does not necessarily have to follow the expert system's line of reasoning. For the purpose of problem solving justifications, this issue has been handled in re-constructive explanation through the use of directed graphs depicting problem states where the process of "problem solving" constitutes of finding a path from an initial state to the solution state using the user inputs. Data labels associated with the edges of the path, are then used to generate the explanation text (Wick et al. 1995). This method or any other that can support the construction of an explanation story, could be used within this architecture.

### **The Expert System Agent**

The underlying premise behind this work, is that the specified architecture could be applied to expert systems that are yet to be constructed, as well as already built ones. However, in the second case, special consideration must be taken for the agentification of such a system. Agentification of the expert system could be achieved in cases where it is possible to separate the HCI component from the problem solving component, and build a wrapper that takes as input the KQML messages then translates them to commands understandable by the expert system. To prevent semantic mismatches across the various agents and the expert system it is important to adhere to ontological specifications or to implement an ontological server.

### **The Architectural Testbed**

In order to test the applicability and effectiveness of the proposed framework, some development tools that provide a simplified, experimental architecture, were implemented in Java and Prolog. Network communication was achieved through the use of TCP/IP sockets. Due to the experimental nature of the implemented architecture, actual KQML messages were not implemented, but instead, very simple protocols were used. Modifying the prototype to communicate using KQML should not be a problem since it only involves modification of the communication parsers and message senders.

For the development of an interface agent, a complete Java template which defines the agent's behavior and essential functionality, was constructed. To facilitate the construction of the expert's system communication model, a Java HC library with a data types dialog interface support, was also implemented. For integrating an existing expert system with the system, a Prolog mediator was built for expert systems developed in KROL (Rafea and Shaalan, 1998). Two components were developed to support the development of the terminology server. The first is a Prolog inference engine that accepts queries for terms, and using a simple KB, returns a pointer to an HTML file that explains that terms. The second is a Java tool, that allows a knowledge engineer to enter terms and their definitions. It then generates HTML files for each of the terms, while automatically adding links and cross references to other terms. This way, as the terms expand within the application, the developer does not have to worry about manually maintaining links. The tool also generates the Prolog representation of terms and their HTML mapping.

For the development of a "why" agent, a KB template and an inference engine that is capable of processing that KB, were developed in Prolog. For the media

management agent, an image manager agent was implemented in Prolog. It, consists of a reusable search engine, a domain specific image database mapping observations to images, and a reusable controller. In order to support the development of this type of an agent, an image tool was constructed in Java. For the construction of a domain justification agent, a KB template and a "How" inference engine were implemented in Prolog. The input to the domain oriented justification agent is a conclusion and instantiated data that led to that conclusion. This agent tries to create a story in terms of the inputs. If it succeeds, it returns the explanation text. If on the other hand it fails, a contradiction is logged. Since a contradiction will only arise if the expert system's knowledge or the explanation knowledge is incorrect, this duplication of knowledge in two different ways serves to verify both systems.

## Applying the Concepts

In order to demonstrate the applicability of the presented architecture, it was applied to an expert system for wheat disease diagnosis. The wheat diseases identification expert system was built a hierarchical classification problem solver using KROL(Rafea and Shaalan, 1998).

Examples of entered definitions include long, detailed versions (Figure 2) or short ones(Figure 3). The underlined words in both examples indicate that there is a hyper-link to a definition for the underlined term.

**Fungal Diseases**  
Fungal disease are those caused by fungi. Fungi differ from other plants in that they have no chlorophyll and thus lack photosynthetic capability. Instead of manufacturing their own food, fungi absorb nutrients from either living or dead host tissue. Fungi are dispersed in many ways: they may be seed borne or soil borne, or they may be spread by way of wind, water (rain, irrigation water), insects, animals, and man. Infection by fungal pathogens depends on several factors: free water on the host plant surface is usually required, the susceptibility of the host, the density of inoculum, and surrounding temperature, as well as other environmental factors. While some fungi attack only one or a few host species, others attack many. Symptoms and disease development are a function of the host-parasite interaction. Symptoms may be similar or distinct, depending on the fungi involved. Positive identification of fungi should therefore be based on their morphology. Wheat fungal diseases include: Leaf Rust, Stem Rust, Yellow Rust, Powdery Mildew, Loose Smut and Common Root Rot

**Figure 2: A detailed definition of a term**

It is important to note that *why* explanations presented are classified into two classes. First there is the general class that gives a rather abstract view of why a question is being asked(Figure 4). Then, there is a specific class of explanations that clearly states why a question is being asked in terms of disorders to which the given question contributes in either confirming or ruling out(Figure 5)

**Hypha**  
A tubular, threadlike filament of fungus mycelium.  
(pl. hyphae)

**Figure 3: An example of a short definition of a term**

**Question: Are there any pustule on the leaves?**  
By knowing whether or not you have pustules, I will be able to investigate the presence of rust diseases, or rule them out completely.

**Figure 4: An example of a general Why explanation**

For either of these classes, dependencies between the current observation and already questioned observations, may exist. Figure 6, shows an example of a dependency between a root and leaf observation.

**Question: Has the entire inflorescence except therachis, been replaced by smut spores?**  
Since you have stated that the spikes are abnormal, I am trying to see whether or not this abnormality is caused by loose smut which is characterized by the observation presented in this question.

**Figure 5: An example of a specific Why explanation**

**Question: Did the root, and basculum tissues change to dark brown or black ?**  
You have stated that the leaves are yellow in color. I am trying to see whether this discoloration is caused by a root related disorder(common root rot) which stops nutrition from reaching the leaves thus causing the observed discoloration.

**Figure 6: An example of a Why dependency**

As stated, the main aim of the *justification* agent is to construct a plausible domain oriented justification story given the expert system's conclusion and its inputs. In the implementation prototype, cases where an explanation was possible for each disorder, were prepared and represented in a KB as specified. In order to show the quality of generated justifications as compared to those that would be generated by some of the other explanation systems, possible justifications for a disorder called loose smut are presented. Figure 7, Figure 8, and , Figure 9 show the justifications that would have been produced by Mycin like Systems (Buchanan and Shortliffe 1984), the generic task approach (Chandrasekran 1996) and the implemented justification agent, respectively.

**User:** How was the diagnosis "Loose smut" reached?  
**System:** This follows from Rule *n*:  
IF growth\_stage = Post Ear Emergence AND  
IF Abn\_plant\_part = Spike AND  
IF inflorescenceRelpacedBySmut AND ....  
THEN Loose smut  
growth\_stage = Post Ear Emergence was given by the user.  
Abn\_plant\_part = Spike was given by the user.  
inflorescenceRelpacedBySmut was given by the user

**Figure 7: Justification from a Mycin Like System**

The agent's generated justification shows considerable improvements in terms of understandability compared to Mycin like systems and Generic task generated

explanations. Compared to reconstructive explanation (Wick et al, 1995), the generated explanations are quite similar, but in addition, in the proposed system's explanation, hyperlinks from within the explanation text appears which further adds to the enhancement of understandability of the generated text. It is important to note that for both this agent and the *why* agent, hyperlinks in the generated HTML change automatically as the system's knowledge about new terms increases.

<p><b>User:</b> How was the diagnosis "Loose Smut" reached ?</p> <p><b>System:</b></p> <ul style="list-style-type: none"> <li>• Post Ear Emergence is true because <ul style="list-style-type: none"> <li>• growth_stage is Post Ear Emergence</li> </ul> </li> <li>• Spike is Abnormal because, <ul style="list-style-type: none"> <li>• Abn_plant_part is Spike</li> </ul> </li> <li>• LoosDis is true because, <ul style="list-style-type: none"> <li>• inflorescenceRelpacedBySmut is true</li> </ul> </li> </ul> <p>Loose Smut has plausibility value +3 because,</p> <ul style="list-style-type: none"> <li>• Post Ear Emergence is true</li> <li>• Spike is Abnormal</li> <li>• LoosDis is true</li> </ul>
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Figure 8: Justification from a Generic task System**

<p><b>User:</b> How was the diagnosis "Loose Smut" reached ?</p> <p><b>System:</b> I attempted to find out the cause of the spike's abnormality. Based on the fact that the entire inflorescence, except the <u>rachis</u>, is replaced by masses of smut I was able to deduce immediately that you have <u>Loose Smut</u> which causes this unique observation. You see, loose smut is a seed borne disease. The loose smut fungus survives as a dormant <u>mycelium</u> within the embryo of the seed. When the infected seed germinates, the fungus is activated and grows as the plant grows eventually reaching the head where it causes all the head tissue except the rachis, to be converted into fungus spores.</p>
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Figure 9: justification from the implemented System**

## Evaluation

Three major characteristics were identified for the analysis of an explanation module. These are fidelity, understandability, sufficiency (Swartout, and Moore 1993). We add to these, the factor of reusability. Fidelity, in the case of "why" explanations, will be maintained as long as the knowledge engineer's documentation of the experts system's behavior is accurate. However, this is not the case with conclusion justifications since they do not necessarily convey a reflection of actual expert system's actions nor are they a true representation of it's line of reasoning. Since fidelity is more important to knowledge engineers and domain experts than to end users, lack of fidelity is not a major limitation given that the proposed system is targeted at end users. Understandability within the proposed system is enhanced in a number of ways. First the user is given the facility to inquire about any term he does not understand. Second, the use of hypertext within the generated explanations for justifications or "why" type explanations, allows a user to get into deeper

levels of details without having to be concerned with his actual knowledge level, since he is capable of determining and guiding his own explanation requirements. Finally, the provision of a facility that retrieves images while taking into account the user's inputs history, eliminates the confusion factor associated with presenting a user with images that do not completely represent his inputs and serves a clarification aid in cases where textual descriptions are not enough.

The sufficiency requirement was satisfied on two levels. The first of these is illustrated by the ability to present conclusion justifications based on domain knowledge rather than on a simple trace of the system's reasoning. This implies that the system had sufficient extra domain knowledge which enabled it to justify a conclusion. The second, is embodied by the system's ability to provide the user with means by which he/she could ask about any subject in the problem domain regardless to its relevancy to problem solving, at any point in time.

Re-usability of explanation on the knowledge level was achieved in several of the implemented agents. The most obvious of these, are the terminology server and media manager agents. Either of these agents could be used by applications that address wheat management aspects. These could be other expert systems, tutoring systems, or standalone applications that provide an interface for these agent services. The re-use of the justification agent is more restrictive since the services it provides confirm to special requirements usually needed by expert system like applications. However, it could be reused across expert systems that diagnose the same disorders regardless of the internal representation of these systems provided that these systems use the same ontology. Since the "why" agent is usually built specifically for a certain expert system, the possibility of reusing it with different applications is significantly low.

## Overall Results and Future Work

By adopting, the presented model, various advantages capitalized:

- The user can navigate through rich domain knowledge and is offered justifications that clarify the expert system's conclusion in clear arguments obtained from knowledge about the addressed domain.
- Because, the proposed architecture does not constrain the development of the explanation system to the development phase of the expert system, explanation systems could be constructed ad hoc for legacy systems.
- Re-usability on the knowledge level is achieved among a number of agents.
- Through the distribution of responsibilities, modularity is achieved and complexity is reduced. Also, because

each of the agents views the others as black boxes with a well defined communication interface, agents can evolve over time without there being a need to make any changes to their client agents/applications.

- By distributing the different agents, bottlenecks arising from a number of client applications attempting to communicate with a single server that provides an aggregate of these services, are reduced and parallelism is enhanced.
- Due to the openness and scalability of the presented architecture, new services, such as “why not”, “what if”, etc., could continue to be added. Also, the followed explanation strategy, could be completely altered with absolutely no change to the different agents except the UI agent.

Another byproduct of the experiments conducted to carry out this research was a development of a methodology for making legacy expert systems available on the Internet. The methodology is applicable to any expert system provided the observations it uses in its reasoning process are stored in a database that could be modified to allow assertion of data values before the initiation of a reasoning process.

The work presented in this paper opens up lots of avenues for future research. The following are some of the points that the authors think are worth investigating:

- The development of justification modules that could be used for planning/design, scheduling, and other types of problems. Further work should also be conducted for enhancing the diagnostic explanation agent introduced.
- The enlargement of the explanation scope such that it would encompass the needs of knowledge engineers and domain expert. In that case, further research should be conducted to configure explanation presentation according to the type of user and his/her explanation interests.
- Investigation of means of adding other explanation components to cover “why not”, and “what if”, types of questions, would further contribute to the enhancement and effect of explanation
- Carrying out experiments to show how much users learn from the provided explanations, would further clarify the impact of the adopted strategy for the purpose of education.
- A long standing argument against the use of connectionist approaches such as neural networks, for developing expert systems, is the inability of these systems to explain their behavior. Using the explanation architecture presented in this thesis could make such systems explainable to end users. Also, since the justification agent presented can use as little as just the essential observations for the establishment of a given

conclusion, the agent could act as a verification layer for the conclusions of such a system. An investigation of the validity of this argument might prove to be worth while.

## Acknowledgments

The authors wish to thank Dr. Mahmoud Rafea for his significant contribution to this work.

## References

- Buchanan B. G., and Shortliffe E. H. eds. 1984. *Rule-Based Expert Systems*. Reading, Mass: Addison-Wesley.
- Chandrasekran, B. 1986. Generic Tasks in Knowledge-Based Reasoning: High-Level Building Blocks for Expert System Design. *IEEE Expert* 1(3) 1986, pp. 23-30.
- Finin, T., Fritzon, R., McKay, D., and McEntire, R. 1994. KQML- A Language and Protocol for Knowledge and Information Exchange. University of Maryland, UMBC, Technical Report CS-94-02.
- Ford, K. M., Coffey, J. W., Cañas A., and Turner C., W. 1996. Diagnosis and Explanation by a Nuclear Cardiology Expert System. *International Journal of Experts Systems*, 9(4): 499-506.
- J. M. Prescott et al. *Wheat Diseases and Pests: a guide for field identification*. International Maize And Wheat Improvement Center, Lisboa, Mexico.
- Rafea, M. and Shaalan, K. 1998. KROL: A Knowledge Representation Language on Top of Prolog. *Expert Systems with Applications* 14 4).
- Schreiber, A. T., Wielinga, B. J., and Breuker, J. A. 1993. *KADS: A Principled Approach to Knowledge-Based System Development*. London: Academic Press.
- Swartout, W. R and Moore J. D. 1993. Explanation in Second Generation Expert Systems. In *Second Generation Expert Systems*. Eds J. M David, J. P. Krivine, and R. Simmons, pp. 543-585.
- Swartout, W. R., Paris C. L., and Moore J. D. 1992. Design For Explainable Expert Systems. *IEEE Expert* 6(3): 57-64.
- Wick M. R., Dutta, P., Wineinger, T., and Conner, J. 1995. Reconstructive Explanation: A Case Study in Integral Calculus, *Expert Systems with Applications*, 8(4): 527-531.