An Intelligent MCDM Approach for Selecting the Suitable Expert System Building Tool

Khalid Eldrandaly
Information Systems Department, Zagazig University, Egypt

Abstract: *Expert Systems (ES), a promising branch of Artificial Intelligence (AI), have achieved considerable success in recent years. This area of AI has concentrated on the construction of high-performance programs in specialized professional domains. Building a new expert system is a major investment. Choosing the right expert system building tool or shell is critical to the success and failure of such investment. The selection of a suitable tool requires consideration of a comprehensive set of factors and balancing of multiple objectives in determining the suitability of a particular tool for building a defined expert system application. Because of the complexity of the problem a number of tools must be deployed to arrive at the proper solution. A new decision making approach is presented in which Expert Systems, and Multi-Criteria Decision Making techniques (MCDM) are integrated systematically in solving expert system building tool selection problem. To implement the proposed decision-making approach, a prototype system was developed in which ES, and MCDM methods (Analytic Hierarchy Process (AHP)) were successfully integrated by using the Component Object Model (COM) technology to achieve software interoperability among the systems components. A typical example is also presented to demonstrate the application of the prototype system.*

Keyword: *Expert Systems, MCDM, AHP, ES building tools selection, ES shells.*

Received March 28, 2006; accepted July 23, 2006

1. Introduction

Expert systems are fast becoming the leading edge of artificial intelligence technology because of the need for such systems in commercial and scientific enterprises and also because AI technology has evolved to the point where expert systems development has become well understood and feasible in many domains. An expert system is a computer program that embodies the expertise of one or more experts in some domain and applies this knowledge to make useful inferences for the user of the system [5].

The expert system paradigm has spawned a host of new expert system tools for building expert systems. The proliferation of these tools can be very confusing to a system developer or project manager who must decide which (if any) tool to use for building a new expert system. These tools are typically large, complex systems themselves, requiring major investments of time, money and effort to acquire, learn and use. Even developers with considerable experience in building traditional systems may feel understandably lost when confronted with this new paradigm. It is therefore important to develop guidelines for evaluating and choosing expert system tools. Evaluating requires discovering which tool characteristics are best suited to accomplishing a given task [14].

Choosing the correct problem scope and picking the right tool for building the expert system are two of the most difficult decisions to make in expert system development [18].

Although there are numerous ES building tools evaluation and selection studies have appeared in the literature they are:

1. Tool specific, limited to specific versions, and quickly outdated.
2. Limited to views of application characteristics.
3. Limited in explanation of functional capabilities
4. They compare features rather than capabilities.
5. They do not present a comprehensive evaluation model [7, 17].

Because of the complexity of the problem, a number of tools must be deployed to arrive at the proper solution. A new decision making approach is presented in which Expert Systems, and Multi-criteria decision making techniques are integrated systematically in solving the expert system building tool selection problem. To implement the proposed decision-making approach, a prototype advisory system was developed in which ES, and a Multicriteria Decision Methods (MCDM) method Analytic Hierarchy Process (AHP) were successfully integrated by using the Component Object Model (COM) technology to achieve software interoperability among the systems components. A typical example is also presented to demonstrate the application of the prototype system.

2. Expert Systems Building Tools

The term expert system tools loosely describes the software that is used for constructing an expert system. These tools range from programs that are used for building the expert system to programs that can aid the knowledge acquisition process. The main software tools for developing expert systems fall into the following three generic categories:

2.1. Programming Languages

These are single languages which are used to build expert systems from scratch (i. e., the ES builder has to develop the user interface from scratch and implement the inference engine using the structures available in the language) and may be subdivided into two general categories:

1. *Conventional Languages*: These languages are also called problem-oriented languages or algorithmic programming languages. Examples are C, COBOL, and Ada. One of the main benefits of using conventional languages is the availability of interfaces to conventional software, such as databases or spreadsheets. However, many of the specific expert system development tools now commercially available also have these facilities. It is possible to build an expert system using a conventional programming language like C or Pascal, or even COBOL, just as it is possible to cut a lawn with scissors! But these languages are unsuitable for building expert systems because they are not suited for manipulating the structures for which knowledge is represented. COBOL, for example, was designed for data processing and not the representation and control of knowledge. Nevertheless, expert systems have been built using languages like C, whose main advantage is its speed of compilation However, using such languages may result in a very long development time, unacceptably large code which results in difficult maintainability, and lack of flexibility and feedback for building the ES [3].
2. *AI Languages*: These languages are also called symbol-manipulation languages because they have been designed for AI applications. The first common examples are LISP and PROLOG. These languages process symbols and have a great advantage over conventional languages in categorizing, analyzing, and reaching conclusions on a logical level of knowledge representation [11]. However, using such languages requires another level of expertise, an individual who is proficient in AI language programming [3].

2.2 Expert System Shells

Shells provide an easy starting point for building an expert system because of their ease of use. They are expert systems that have been emptied of their knowledge bases. This means that the developers can concentrate only on entering the knowledge base without having to build everything, including the inference engine and user interface, from scratch. Even non-programming experts can familiarize themselves with shells fairly rapidly. However, using a shell to build an expert system can seduce the builder into oversimplifying the application domain because shells are inflexible, in that it is difficult to modify or change the way they work with regard to both representation of knowledge and the inference mechanism. There are several shells commercially available such as EXSYS. The primary disadvantage of using a shell in building an expert system is that it will generally embody only one reasoning methodology and knowledge representation technique, while sophisticated applications often require a combination of techniques [1, 3].

2.3 Knowledge Engineering Languages (AI Toolkits)

These are very sophisticated "hybrid tools", which typically contain code structures for a range of expert systems tasks. They make use of rules, frames, Object-Oriented Programming (OOP), and logic or semantic networks. They may also use forward and backward chaining, CBR, and a wide Variety of inheritance techniques. AI toolkits are more specialized than shells. Therefore, they can increase productivity. However, because of their complexity, they require more skill than shells or programming languages. Unlike shells, which are predominantly suited to small standalone applications, AI toolkits are more suited to larger client/server corporate applications. Two of the most commonly used AI toolkits are ART-IM and Level 5 Object [3, 18].

3. New Decision Making Approach

A new approach for ES building tool selection is presented. The approach integrates the capabilities of ES, and MCDM and provides an advisory system to assist the user during the tool selection procedure. Recommendations submitted by others [2, 7, 8, 13, 14, 16, 17, 18] regarding the design of a good evaluation and selection methodology were observed in the design of the proposed approach. Figure 1 depicts the three phases of the proposed approach (i. e., justification, screening and evaluation phases) and their procedural steps

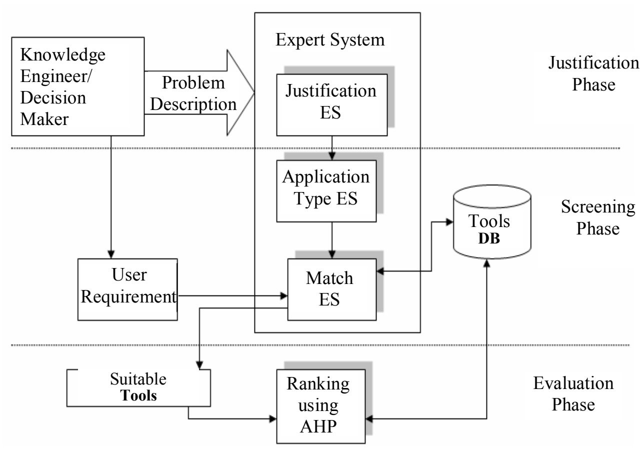


Figure 1. Framework of the proposed approach.

1. *Justification Phase*: In this phase an expert system is used to justify building an expert system for the application under consideration. That is, this phase will answer the question "Will expert system work for my problem?"
2. *Screening Phase*: This phase consists of the following two steps:

   a- *Identifying the end-user application type*: The expert system is used to assist the decision maker in defining the application type and to provide the recommended tool capabilities required for building the proposed application. The output of this step is a set of recommended tool capabilities. The decision maker has the option of accepting or modifying these recommended capabilities.

   b- *Tool Screening*: The expert system is used to identify candidate tools that meet the desired tool capabilities. The output of this step is a list of candidate tools for further assessment.
3. *Evaluation Phase*: After identifying the tools that are best suited for the subject application in the screening phase, selection of the most appropriate tool can be made. Comparing alternative tools involves consideration of multiple criteria that have not been considered in the screening phase and may have conflicting characteristics. AHP, a MCDM technique, is used to address this multicriteria problem. The output of this phase would be either a recommended tool or a list of tools ordered by their level of suitability.

4. ES Building Tools Advisory System

To implement the presented decision making approach, a prototype advisory system was developed using three COM-compliant commercially available software packages: Microsoft®Visual Basic 6.0, Visual Rule Studio®, Microsoft® Access 2003. Microsoft®Visual Basic 6.0 was used to develop the MCDM (AHP) module, to provide the shell for the COM integration, and to develop the system's user interface. Visual Rule Studio® was used to develop the expert system module. Microsoft® Access 2003 was used to develop the database module.

The proposed system was developed as a three-tire architecture as shown in Figure 2.

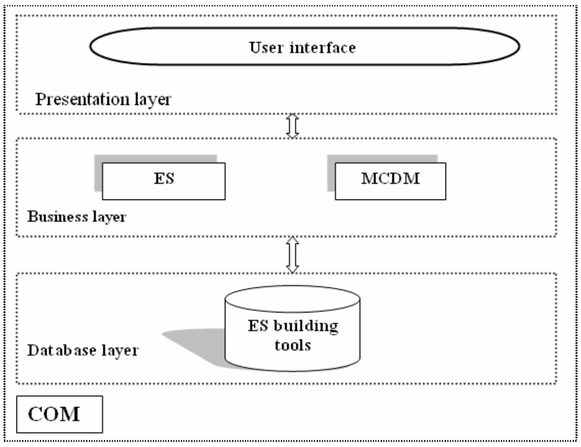


Figure 2. Three-tire architecture of the proposed system.

5. Development of the Prototype Expert System

Visual Rule Studio® (an object-oriented COM-compliant expert system development environment for windows) was used to develop the prototype expert system. Visual Rule Studio solves the problem of software interoperability by allowing the developers to package rules into component reusable objects called RuleSets. By fully utilizing OLE and COM technologies, RuleSets act as COM Automation Servers, exposing RuleSet objects in a natural COM fashion to any COM compatible client. Visual Rule Studio installs as an integral part of MS Visual Basic 6.0, Professional or Enterprise Editions, and appears within the Visual Basic as an ActiveX Designer. This allows the developers to add rule objects to their existing or new Visual Basic application in much the same manner they would extend their application with a new form or ActiveX control. RuleSets can be complied within Visual Basic. EXE, .OCX, or .DLL executables and used in any of the ways the developers normally use such executables [15].

The knowledge base of the proposed expert system consists of three different RuleSets. The first RuleSet consists of 4 classes and 18 rules, the second RuleSet consists of one class and 42 rules, and the third RuleSet consists of 3 classes and 12 rules. Each one of them represents a separate Knowledge Source (KS). These KSs are independent chunks of knowledge and do not directly communicate with each other. Instead, they participate in the problem solving process by writing messages on a global database called blackboard and reading messages from other knowledge sources. This type of architecture is called blackboard architecture and is shown in Figure 3. The blackboard architecture is intended to support development of systems in domains characterized by interaction between diverse sources of knowledge and hence provides a framework for integrating knowledge from several sources. The blackboard serves as a global data structure, which facilitates this interaction. Usually, in typical blackboard architecture, the inference mechanism consists of the agenda and the monitor. The agenda keeps track of all events in the blackboard and calculates the priority of execution for KSs that were generated as a result of the activation of other KSs. The monitor takes the element with the highest priority and executes it. However, there is no fixed agenda and monitor in the current blackboard architecture. Since different solution steps of this system are explicitly seen on the main screen display, the sequences of the different processes are primarily selected by the user. Without fixed agenda, the user is free to change input data and check intermediate results given by the system during the consultation session. Detailed description of blackboard architecture is reported elsewhere [6, 12].

The following gives a typical example of the classes and rules used in the third RuleSet:

Class Category_Criteria
With Category *Numeric*
With Inference_Tracing *String*
With Forward_Chaining *String*
With Backward_Chaining *String*
With Bi_Directional *String*
With Certainty_Factor *String*
With Fuzzy_Sets *String*
With Blackboard *String*
With Production_Rule *String*
With Decision_Table *String*
With Frames *String*
With Semantic_Network *String*

Rule 1 Category Criteria
If Category_Criteria.Category =1
Then Category_Criteria.Inference_Tracing := “Yes”
And Category_Criteria.Forward_Chaining := “No”
And Category_Criteria.Backward_Chaining := Yes”
And Category_Criteria.Bi_Directional := "No”
And Category_Criteria.Certainty_Factor := "Yes”
And Category_Criteria.Fuzzy_Sets := "No”
And Category_Criteria.Blackboard := "No”
And Category_Criteria.Production_Rule := "Yes”
And Category_Criteria.Decision_Table := "No”
And Category_Criteria.Frames := "No”
And Category_Criteria.Semantic_Network := "No”

The inference engine of Visual Rule Studio’s production system acts as the "unseen hand" or executor which causes processing to take place. Processing here is defined as the combining of supplied data with rules to create inferred data. It is the inferred data that is the desired end result of the production system processing. The Visual Rule Studio inference engine provides two primary problem-solving engines relevant to production systems: the forward chaining engine and the backward chaining engine [15]. In the proposed expert system forward chaining engine is used. Starting from an initial or current set of data, the forward chaining inference engine makes a chain of inferences until a goal is reached.

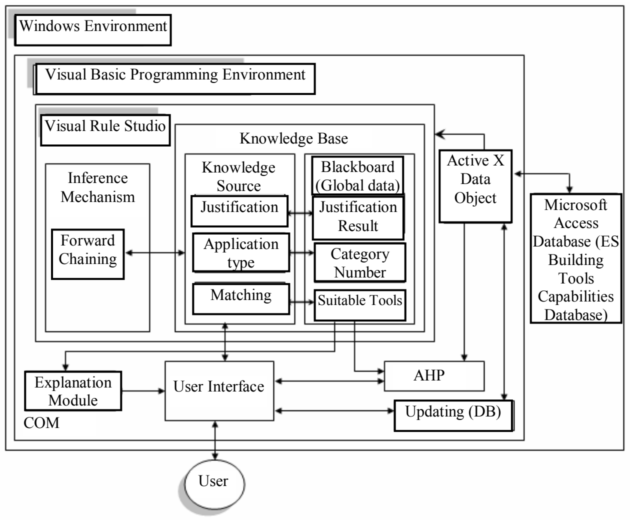


Figure 3. Blackboard architecture of the proposed expert system.

6. MCDM (AHP) Module

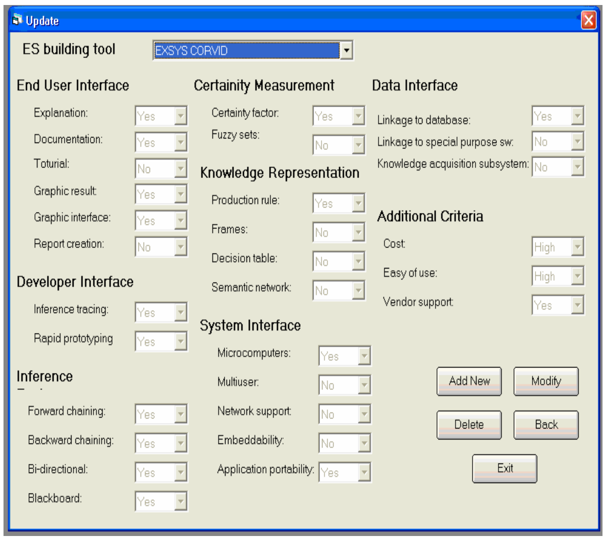
Over the last three decades, a number of MCDM have been developed. Among them, the AHP is perhaps the most prominent and successful method. AHP is a method that allows the consideration of both objective and subjective factors in ranking alternatives. Since its introduction in the mid 1970s, AHP has been applied in a wide variety of practical applications in various fields including economics, planning, energy policy, health, conflict resolution, site selection, project selection, and budget allocation. It assists the decision making process by allowing decision-makers to organize the criteria and alternative solutions of a decision problem in a hierarchical decision model.

The AHP decision hierarchy involves a number of steps: Identification of the goal (e. g., to select the most suitable ES building tool), use of a set of decision factors/ variables/ criteria (e. g., vendor support, economic costs, and easy of use), and determination of a set of alternatives/choices (e. g., Tool 1, Tool 2 and Tool 3). The levels of the hierarchy may be expanded as needed (e. g., cost could be considered in terms of initial, and maintenance). At the lowest level on the hierarchy we find the alternative solutions. Comparisons of the available choices/ alternatives are made on a pair-wise basis. For example in considering initial costs, AHP would determine whether Tool 1 is “better” (that is., has lower initial cost) than Tool 2 and if so, by how much? Similar comparisons are performed at each level on the hierarchy. This measure of importance/weight is done using a nine-point scale, which is widely utilized in the AHP technique. The AHP process synthesizes the alternatives' priorities into overall set of values that indicate the relative importance of each factor at the lowest level of the hierarchy. Detailed description of MCDM and AHP is reported elsewhere [4, 10].

7. Database Module

Microsoft® Access 2003 was used to develop the ES tools database module. This database contains more than 50 ES building tools. The system gives the user the opportunity to update the database either by adding new tools or editing the current tools as shown in Figure 4.

Microsoft® ActiveX® Data Object (ADO) was used to read required information from the database. ADO provides consistent, high-performance access to data and supports a variety of development needs, including the creation of front-end database clients and middle-tier business objects, using applications, tools, languages or Internet browsers. ADO is designed to be the one data interface needed for one-to-multitier client/server and web-based data-driven solution development. ADO was implemented using a set of COM-based interfaces that provide applications with uniform access to data stored in diverse information sources [9].


Update

ES building tool: EXSYS CORVID

End User Interface
- Explanation: Yes
- Documentation: Yes
- Toturial: No
- Graphic result: Yes
- Graphic interface: Yes
- Report creation: No

Developer Interface
- Inference tracing: Yes
- Rapid prototyping: Yes

Inference
- Forward chaining: Yes
- Backward chaining: Yes
- Bi-directional: Yes
- Blackboard: Yes

Certainity Measurement
- Certainty factor: Yes
- Fuzzy sets: No

Knowledge Representation
- Production rule: Yes
- Frames: No
- Decision table: No
- Semantic network: No

System Interface
- Microcomputers: Yes
- Multiuser: No
- Network support: No
- Embeddability: No
- Application portability: Yes

Data Interface
- Linkage to database: Yes
- Linkage to special purpose sw: No
- Knowledge acquisition subsystem: No

Additional Criteria
- Cost: High
- Easy of use: High
- Vendor support: Yes

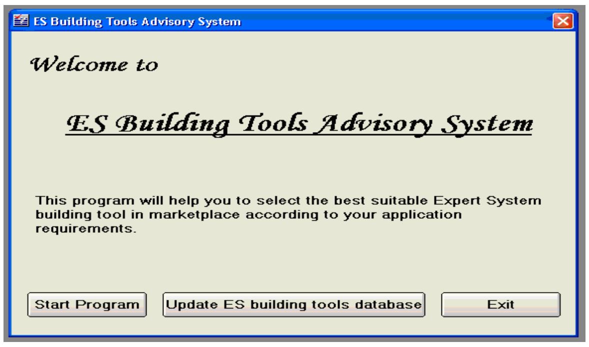
Figure 4. Updating ES building tools database screen.

8. Example of Consultation Session

In order to demonstrate how the proposed system can assist the knowledge engineer in selecting the suitable ES building tool for his application, choosing the suitable tool for building the ES building tools advisory system itself is demonstrated in this section.

Upon execution of the system, it gives the user the option of either starting the program or updating the ES tools database as shown in Figure 5.

Upon choosing to start the program, the system gives the user the opportunity to execute any phase of the three solution phases as shown in Figure 6 because the system is designed in modular form. If the user chooses to execute the first phase (justification phase), the system will help the user in answering the following question" Will expert system work for my problem?" Figure 7 is a sample of screenshots during the justification phase.

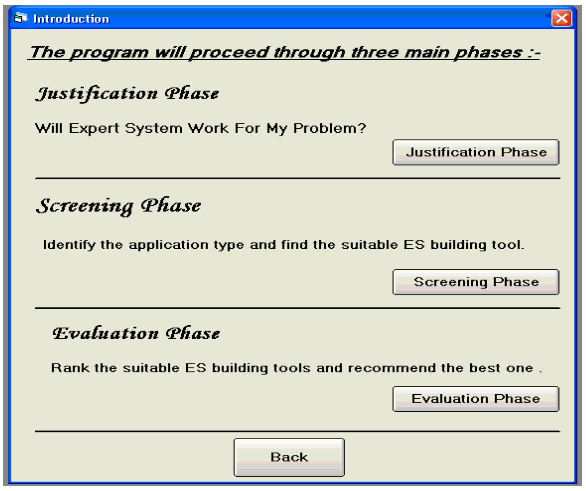

ES Building Tools Advisory System

Welcome to

ES Building Tools Advisory System

This program will help you to select the best suitable Expert System building tool in marketplace according to your application requirements.

Figure 5. Main screen.


Introduction

The program will proceed through three main phases :-

Justification Phase

Will Expert System Work For My Problem?

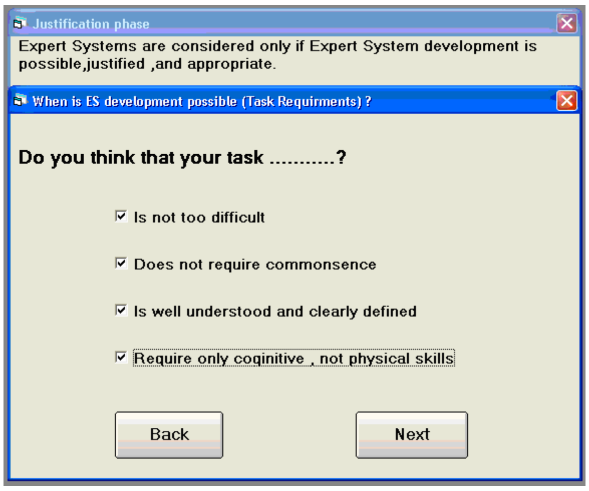
Screening Phase

Identify the application type and find the suitable ES building tool.

Evaluation Phase

Rank the suitable ES building tools and recommend the best one .

Figure 6. Different phase of the program.


Justification phase

Expert Systems are considered only if Expert System development is possible,justified ,and appropriate.

When is ES development possible (Task Requirments) ?

Do you think that your task ....?

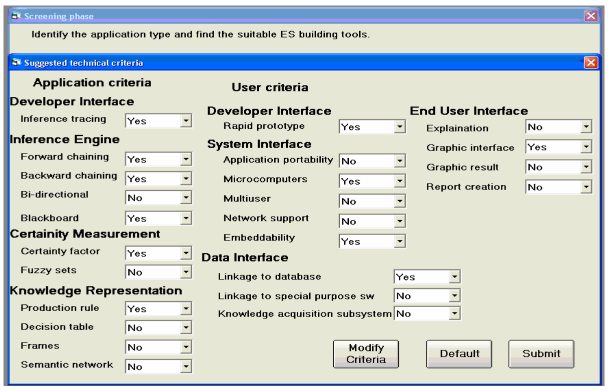
- ☑ Is not too difficult
- ☑ Does not require commonsense
- ☑ Is well understood and clearly defined
- ☑ Require only cognitive , not physical skills

Figure 7. A sample of screenshots during justification phase.

After finishing the justification phase or if the user chooses directly the screening phase, during this phase the system will help the user identifying the proposed application type and the required ES building tool capabilities. The output of this phase is a list of candidate tools for further assessment. Figures 8 and 9 are samples of the screenshots during this phase.

The final phase is the evaluation phase, in which the system will help the user in evaluating the candidate tools based on new evaluation criteria using AHP. Figures 10 and 11 are sample of the screenshots during this phase.

As shown in Figure 11, the system recommends Visual Rule Studio as the most suitable tool for building the current system; this result matches exactly the actual used tool and hence verified the validity of the proposed system.


Screening phase
Identify the application type and find the suitable ES building tools.
Suggested technical criteria

Application criteria

Developer Interface
Inference tracing Yes

Inference Engine
Forward chaining Yes
Backward chaining Yes
Bi-directional No
Blackboard Yes

Certainity Measurement
Certainty factor Yes
Fuzzy sets No

Knowledge Representation
Production rule Yes
Decision table No
Frames No
Semantic network No

User criteria

Developer Interface
Rapid prototype Yes

System Interface
Application portability No
Microcomputers Yes
Multiuser No
Network support No
Embeddability Yes

Data Interface
Linkage to database Yes
Linkage to special purpose sw No
Knowledge acquisition subsystem No

End User Interface
Explaination No
Graphic interface Yes
Graphic result No
Report creation No

Figure 8. A sample of screenshots during screening phase.

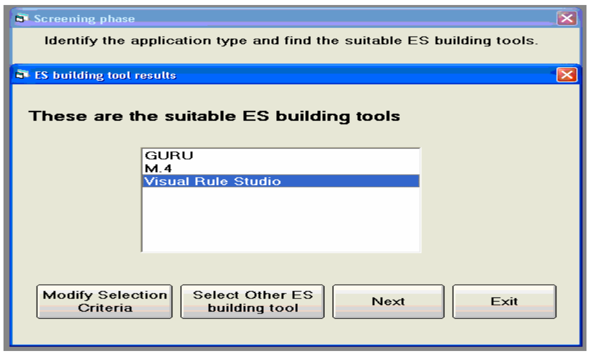

Screening phase
Identify the application type and find the suitable ES building tools.
ES building tool results
These are the suitable ES building tools
GURU
M.4
Visual Rule Studio

Figure 9. Results of the screening phase.

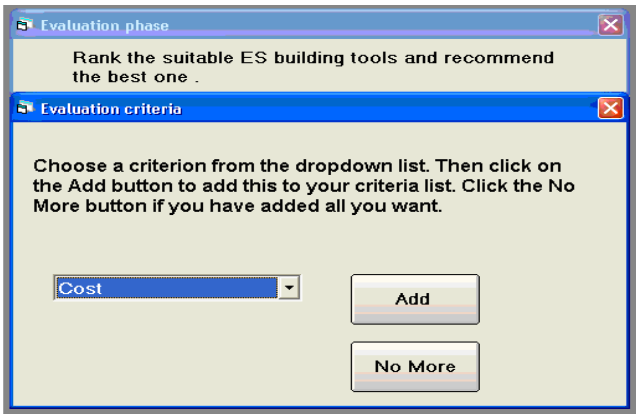
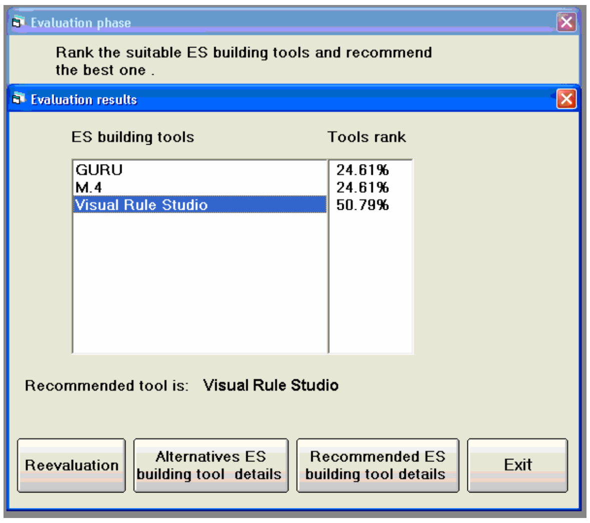

Evaluation phase
Rank the suitable ES building tools and recommend the best one .
Evaluation criteria
Choose a criterion from the dropdown list. Then click on the Add button to add this to your criteria list. Click the No More button if you have added all you want.
Cost

Figure 10. Identifying the evaluation criteria during evaluation phase.

9. Conclusions

In this paper, a new decision making approach for ES building tool selection is presented. This approach integrates the capabilities of ES, and MCDM (AHP) and provides an advisory system to assist the knowledge engineers and system developers during the tool selection procedure. The architecture, the development, and the implementation of the prototype advisory system are discussed in details. The use of Visual Rule Studio® (an object-oriented COM-compliant expert system development environment for windows) which runs together with Microsoft Visual Basic 6.0 is found to be very effective in producing the system under Windows environment. The advantages of both production rules and object-oriented programming paradigm are accomplished. Also, software interoperability between the different components of the system is achieved by adopting the COM technology in designing the system. The system's ES tools database could be updated easily to match the dynamic nature of the software market.


Evaluation phase
Rank the suitable ES building tools and recommend the best one .
Evaluation results

ES building tools	Tools rank
GURU	24.61%
M.4	24.61%
Visual Rule Studio	50.79%

Recommended tool is: Visual Rule Studio
Reevaluation
Alternatives ES building tool details
Recommended ES building tool details
Exit

Figure 11. Output of the evaluation phase.

References

[1] Awad E., *Building Knowledge Automation Expert Systems with EXSYS CORVID*, EXSYS Inc, Albuquerque, 2003.

[2] Beach S. and Gevarter W., "Standards for Evaluating Expert System Tools," *Expert Systems with Applications*, vol. 2, no. 4, pp. 259-267, 1991.

[3] Darlington K., *The Essence of Expert Systems*, Prentice Hall, 2000.

[4] Forman E. and Selly M., *Decision by Objectives: How to Convince Others that You are Right*, World Scientific Publishing Company, New York, 2001.

[5] Hayes-Roth F., Waterman D., and Lenat D., *Building Expert Systems*, Addison Wesley, 1983.

[6] Hunt J., *Blackboard Architectures*, http://www.jadeetechnology.co.uk, 2002.

[7] Kuesten C. and McLellan M., "Expert system Shells: Selecting the Most Appropriate Development Environment," *Food Research International*, vol. 22, no. 2, pp. 101-110, 1994.

[8] LeBlanc L. and Jelassi M., "An Evaluation and Selection Methodology for Expert System Sells," *Expert Systems with Applications*, vol. 2, no. 2, pp. 201-209, 1991.

[9] Microsoft, *Microsoft Developer Network Online Documentation*, MSDN Library, 2003.

[10] Mollaghasemi M. and Pet-Edwards J., *Making Multiple-Objective Decisions*, IEEE Computer Society Press, USA, 1997.

[11] Nelson C. and Balachandra R., "Choosing the Right Expert System Building Approach," *Decision Sciences*, vol. 22, no. 2, pp. 354-368, 1991.

[12] Nii P., "Blackboard Systems at Architecture Level," *Expert Systems with Applications*, vol. 7, no. 1, pp. 43-54, 1994.

[13] Preece A. and Moseley L., "Empirical Study of Expert System Development," *Knowledge-Based Systems*, vol. 5, no. 2, pp. 137-145, 1992.

[14] Rothenberg J., "Expert System Tool Evaluation," *in Guida G. and Tasso C. (Eds), Topics in Expert System Design: Methodologies and Tools,* Elsevier Science, North Holland, 1989.

[15] RuleMachines, *Visual Rule Studio Developer's Guide*, Canada, 2002.

[16] Stylianou A., Madey G., and Smith R., "Selection Criteria for Expert System Shells: A Socio-Technical Framework," *Communications of the ACM*, vol. 35, no. 10, pp. 30-48, 1992.

[17] Stylianou A., Smith R., and Madey G. "An Empirical Model for the Evaluation and Selection of Expert System Shells," *Expert Systems with Applications*, vol. 8, no. 1, pp. 143-155, 1995.

[18] Waterman D., *A Guide to Expert Systems*, Addison Wesley, 1986.

Khalid Eldrandaly received his BS in civil engineering, his MS in systems engineering (expert systems), and his PhD in systems engineering (GIS). He was a visiting scholar at Texas A&M University, USA, for two years. Currently, he is an assistant professor of computer information systems and interim head of Information Systems and Technology Department, Zagazig University, Egypt. His area of interests includes GIS, expert systems, SDSS, MCDM, and intelligent techniques in decision making. He is a member of the World Academy of Young Scientists (WAY), Arab Union of Scientists and Researchers (AUSR), and Texas A&M International Faculty Network, and Egyptian Software Engineers Association (ESEA).