Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 1998 Proceedings | Americas Conference on Information Systems (AMCIS)

December 1998

Acquiring Systems Knowledge with GOOI (Graphical Object-Oriented Interfaces)

Steven Walczak
University of South Alabama

Acquiring Systems Knowledge with GOOI (Graphical Object-Oriented Interfaces)

Steven Walczak
School of Computer and Information Sciences
University of South Alabama

Abstract

Information system development, in particular expert systems and other knowledge-based approaches, require extensive human expert knowledge. Often, acquiring such knowledge is problematic with regard to efficiently acquiring the expert's knowledge and translating this knowledge into a system usable form. Knowledge acquisition has long been viewed as the bottleneck of knowledge-based systems and more recently is being recognized as a significant issue in general information systems analysis and design. Object-oriented techniques are presented as a uniform method for overcoming translation difficulties and implementing graphical interfaces. A graphical interface provides a modeling platform that is easily understood by experts and knowledge engineers. The object-oriented base for our tool provides an additional benefit in developing implemented systems by providing a representation independent methodology that can easily be mapped into any other object-oriented based expert system or other object-oriented information systems.

Introduction

Knowledge acquisition has long been viewed as the bottleneck in the analysis, design, and implementation processes for information systems, including expert systems. Development of an accurate and detailed requirements model may mean the difference between success and failure of an information systems project[3]. Over the past decade, the concern for improving the knowledge acquisition process has produced much research aimed at automating the knowledge acquisition phase in the design of knowledge-based systems. Many of the automated knowledge acquisition systems produced by research suffer from several problems: application limited to a specific domain and limited transportability of the knowledge representation system to specific expert system solutions [5].

Interface design is a critical issue in information system, knowledge acquisition, and expert system design. Systems designers must try to design user/expert interactions to enhance the recall performance of the expert and compensate for recall limitations [7]. Graphical information presentation (and elicitation) methods are the most popular user interface choice [12]. Additionally, graphical (white board) models facilitate communication and expression of ideas, especially when encountering differing terminology [1,8].

The research goal presented in this paper is to develop a generalized knowledge acquisition system that can be applied universally to knowledge acquisition problems. By developing a generalized automated graphical knowledge acquisition system, we will solve the problem of limited application, since the system can be applied to all domains regardless of the specific problem-solving format. The cross domain applicability is inherent in the graphical modeling context, which abstracts out representation problems specific to the problem solving paradigm [2] and enhances the cognitive fit between the representation and underlying knowledge of the domain expert [13].

The knowledge representation and transportability problem with the use of an object-oriented framework for acquiring expert knowledge. Object oriented databases provide a rich data modeling capability [10]. This same modeling richness should be utilized by automated knowledge acquisition tools. Furthermore, because humans naturally learn to classify their world into objects [1], understanding and assimilation of the object-oriented representation protocols should be moderately effortless.

Benefits of Object-oriented Modeling in Expert Systems

Any graphical model of information may be represented by using two distinct types of classes: knowledge element classes and knowledge hierarchy classes. The graphical knowledge acquisition methodology used for OOEKA (Object-Oriented Expert Knowledge Acquisition) is based on the modified Petri net [11] graphical model proposed by Walczak [14]. For the object-oriented implementation of this graphical knowledge acquisition method, three different classes of objects are required: one class each for the places and events corresponding to action objects and decision objects (knowledge elements), and finally a container class object that maintains the other two objects in a directed graph similar to the Petri net formalism (knowledge hierarchy) used by Walczak. The separation of the knowledge from the knowledge representation scheme via classes, implies that both understandability and ease of maintenance will increase.

The three basic classes of objects required to model domain knowledge via the Petri net formalism are referred to as the Base objects or alternatively, the Knowledge-Model objects. Other control and interface class objects are defined, such as the menu object class, that are specific to the windows interface design. Most of these additional object classes are not directly related to the knowledge acquisition process, but instead deal with providing a uniform Windows© interface to reduce cognitive dissonance and increase knowledge elicitation efficiency. To maintain the desirable object-oriented properties of class encapsulation and identity, those menu actions and Windows© buttons that are involved with the knowledge elicitation, modeling, or verification process are instantiated through the activation of a corresponding Knowledge-Model object method (behavior). Furthermore, with MFC© and other GUI interface class libraries, most of the interface-environment classes are already defined.

Human expert knowledge is typically categorized via either inheritance or aggregation. Object-oriented modeling provides simple representations and implementations of these two basic types of knowledge categorization. Since the purpose of an object-oriented paradigm is to more closely simulate natural human knowledge representation, ease of understanding will also increase [1].

Once knowledge has been acquired, the next problem encountered by knowledge engineers is transforming the acquired knowledge into a format that is usable by the expert system. There is frequently a difference in perception of the elicited domain information between the expert and the knowledge engineer based on different backgrounds in the problem domain and ensuing goals for use of the knowledge [2,6]. Encapsulating each acquired piece of knowledge as an object within an object-oriented framework reduces this implementation problem.

By the very nature of object-oriented design and programming, objects can be incorporated into any other object-oriented system, by including the class definition and the created object instances of the class (the acquired knowledge base). An object-oriented expert system can then reference the acquired domain objects directly without any knowledge representation transformation problems. For example, production rule based expert system (the most popular form of expert system [9]) can use the decision objects as the antecedent for the production rules and the action objects that follow the decision object (within the directed graph container class) as the consequent.

Additionally, new capabilities are extremely easy to add to either the knowledge acquisition tool or expert system. New capabilities only require that a new attribute and corresponding method (object behavior) be defined in the class definition for the type of object that will gain the new ability. For example, an audio component can be added to both actions and decision objects by including an *audio_segment* attribute that points to a corresponding .wav file (or other audio formatted file) and a modification to the ON_CLICK method that plays the audio file whenever the corresponding object is selected.

Because of object encapsulation, each object class is self-contained and may be validated using a bottom-up testing methodology [1] independently of other class definitions. Class communications protocols may be validated as the testing process continues. If an ontology can be specified regarding the necessary and required public methods for all Knowledge-Model objects, then object classes may be re-used in other information systems (including all O-O artificial intelligence applications). The polymorphism capabilities of most object-oriented languages (C++, Delphi, etc.) promote the definition of an object-oriented knowledge-based system ontology.

An Object Graphical Tool for Knowledge Acquisition (KA)

The use of directed graphs to represent the various aspects of expert knowledge provides several benefits. Directed graphs can be directly displayed via a graphical interface that limits the quantity of translation required to represent the expert's knowledge for use in a knowledge-based system. Directed graphs reduce the cognitive inference required of the domain expert during the elicitation process. The reduction in cognitive complexity is achieved because humans process a majority of their sensory information through visual channels (i.e., a picture is worth one thousand words). Subsequently, the communication requirements between the expert and the knowledge acquisition system are reduced. Because knowledge is being elicited graphically, the knowledge acquisition system only needs to have a simple vocabulary with little or no grammatical rules and the system can still effectively communicate with the domain expert.

The OOEKA tool has been developed using Microsoft© Visual C++™. The use of object class libraries decreases the development time of the knowledge acquisition tool and provides many of the graphical classes needed to implement the interface. The Draw menu allows the user to select the type of object to be placed on the graphical model, places, events, connections, or text. The first three object types are also accessible via buttons on the button bar. The place and event interface methods each create an object of the corresponding action or decision class, while the connection object creates a new directed graph container class object and makes the corresponding connections between the objects now held within the container class. File and Edit menus consistent with other Windows© applications are included for uniformity.

OOEKA implements a consistency check, performed by first verifying that all action and decision objects exist in the container class' directed graph model and then simulating the graph to make sure that all contained nodes are reachable. A coaching strategy is used to check the completeness of a directed graph model given by the expert. Each node within the directed graph container class is highlighted and the expert is asked if any additional detail can be specified concerning the highlighted object. Should the expert respond affirmatively, then either a new object is added to the existing graph or a new directed graph model is started and the expert is asked to fill in the sub-graph model with the additional detail. Addition of new knowledge to an existing model is facilitated in the object-oriented paradigm via creation of new objects within a class or class inheritance hierarchy. These two tools assist expert users in building accurate models of their problem solving methods and overcoming the widely recognized problem [1, 6] of experts not being able to fully recall or describe their problem solving methods. If the model specified by the expert is incomplete, then the resulting expert system will perform poorly. The consistency and completeness methods help guarantee a complete and usable model.

The Help menu item initially displays general information about the graphical modeling process (e.g., that actions are necessarily followed by decisions which in turn are necessarily followed by actions) and the differences between action graph objects and decision graph objects. The expert user may also provide a description of the task or decision to be performed for each corresponding object and these descriptions are displayed in pop-up windows when Help is invoked on a corresponding object. Future work on the Help facility will include the implementation of an on-line tutorial that demonstrates to a domain expert how to create a simple directed graph (Petri) model and how to use the Consistency and Completeness menu choices of the Check menu.

Analyzing the Benefits of Graphical KA Interfaces

An analysis of the benefits provided through the use of the OOEKA tool is performed to extend the previous research on graphical knowledge acquisition tools that report methodology without an explicit evaluation [14]. An expert accountant, who is already familiar with the Windows© environment (thus eliminating additional training time that would be required of non-Windows users), is asked to define the accounting domain problem solution to determining all of the required tax forms that would need to be filed by a corporation under United States tax laws and regulations. The expert accountant went through both a traditional interview knowledge acquisition session and a session that used only the OOEKA tool.

The first session uses the OOEKA object-oriented knowledge acquisition tool. OOEKA is used prior to other knowledge acquisition techniques to eliminate any transitional benefits achieved by having thought through the problem previously during another knowledge acquisition method trial. A preliminary description of the program and its functionality is given to the expert accountant to familiarize him with the OOEKA tool. The expert is then left alone to complete the desired model. After several days time, again to reduce the effect of knowledge (problem solving) carry-over, knowledge acquisition for the same problem is conducted with the expert accountant using traditional interview techniques. Three iterations of interviews are performed to complete the model, which correspond to the sub-graphs elicited by the OOEKA program described above. The two models produced by the automated and interview knowledge acquisition techniques are comparable.

The time taken to acquire the expert knowledge for this particular domain problem can be expressed as:

$$t\ (\text{Interview}) = 1.5 * t\ (\text{OOEKA}) \qquad (1)$$

Equation (1) states that the interview technique took 50 percent additional time to complete, even when the time to explain the OOEKA program is factored into the equation's time. Thus, the graphical knowledge acquisition methodology provides a 34 percent time savings to both the experts and systems developers. The time reduction in the knowledge acquisition task is consistent with Gruber's [4] findings.

Furthermore, an additional factor that is not figured into the time equation (1), is the amount of time required of the knowledge engineers to familiarize themselves with the domain problem to be modeled. Depending on the complexity of the domain problem the knowledge engineer is attempting to model, the additional domain familiarization time can last from hours to weeks, since most knowledge engineers are not inherently familiar with the domain in which their expert system will operate. If this additional, pre-knowledge acquisition task time is incorporated into the time equation (1), then the resultant savings will be greater than 50 percent, with the actual savings dependent on the amount of time spent in the domain familiarization stage.

Summary

The use of graphical modeling for performing knowledge acquisition of procedural tasks provides a user-friendly interface (due to minimal keyboard input) which efficiently acquires the knowledge of domain experts. Expert knowledge elicitation is enhanced since the graphical model is well understood and reduces cognitive disparity. Menu-based tools within the OOEKA knowledge acquisition system compensate for memory recall limitations of experts by validating the connectivity of the graphical model produced and inquiring for any potential missing knowledge. By using this graph-based methodology, the difficulty of communicating with the expert in appropriate terminology is virtually eliminated (except for the help windows). Additionally, the description attribute of each object enables the expert to provide a natural language description of the process just elicited.

Since OOEKA is implemented using an object-oriented framework, the resulting knowledge base can be directly utilized by any other object oriented expert system or intelligent system with little or no transformation. To assist with the cross platform transportability of object-oriented knowledge bases, a taxonomy of required and desired attributes and access methods must be developed. This taxonomy is the subject of future research in object-oriented expert systems.

The object-oriented framework enables rapid adaptation of the OOEKA and resulting expert systems to new requirements. New capabilities are added into the knowledge base by simply adding an attribute and corresponding methods (for access) to the class definition for either action objects or decision objects.

The use of an object-oriented graphical knowledge acquisition tool provides many significant benefits to knowledge engineers and domain experts. Encapsulation enables consistency checking. Growing libraries of object-oriented knowledge representation data structures significantly decrease problematic issues associated with encoding elicited knowledge. In addition to virtually eliminating knowledge representation problems, the OOEKA system enables a 34 percent decrease in the knowledge acquisition time for the expert and additionally reduces the front-end learning curve of the knowledge engineers performing the acquisition. Furthermore, the object-oriented model improves the usability of the knowledge model and provides the opportunity for reuse of the knowledge in similar domains.

References

[1] Brown, D. *An Introduction to Object-Oriented Analysis*, Wiley: New York, 1997.

[2] Byrd, T. A., Cossick, K. L., & Zmud, R. W. A Synthesis of Research on Requirements Analysis and Knowledge Acquisition Techniques, *MIS Quarterly*, 16 (1), 117-138, 1992.

[3] Ewusi-Mensah, K. Critical Issues in Abandoned Information Systems Projects, *Communications of the ACM*, 40 (9), 74-80.

[4] Gruber, T. *The Acquisition of Strategic Knowledge*, Academic Press: London, 1989.

[5] McGraw, K. L., & Harbison-Briggs, K. *Knowledge Acquisition Principles and Guidelines*, Prentice Hall: Englewood Cliffs, NJ, 1989.

[6] Meyer, M. A. & Booker, J. M. *Eliciting and Analyzing Expert Judgement: A Practical Guide*, Academic Press: London, 1991.

[7] Mitchell, C. M. & Sundström, G. A. Human Interaction with Complex Systems: Design Issues and Research Approaches. *IEEE Transactions on Systems, Man, and Cybernetics Part A*, 27 (3), 265-273, 1997.

[8] Peters, T. J. & Waterman, R. H. *In Search of Excellence*, Harper and Row: New York, 1982.

[9] Pigford, D. V. & Baur, G. *Expert Systems for Business: Concepts and Applications*, Boyd and Fraser: Danvers, MA, 1995.

[10] Ram, D. J., Vivekananda, N., Rao, S., & Mohan, N. K. Constraint Meta-Objects: A New Object Model for Distributed Collaborative Design. *IEEE Transactions on Systems, Man, and Cybernetics Part A*, 27 (2), 208-221, 1997.

[11] Reisig, W. *Petri Nets An Introduction*. Springer-Verlag: Berlin, 1985.

[12] Tan, J. K. & Benbasat, I. Processing of Graphical Information: A Decomposition Taxonomy to Match Data Extraction Tasks and Graphical Representations. *Information Systems Research*, 1 (4), 416-439, 1990.

[13] Vessey, I. & Galletta, D. Cognitive Fit: An Empirical Study of Information Acquisition. *Information Systems Research*, 2 (1), 63-84, 1991.

[14] Walczak, S. Graphical Knowledge Acquisition for Multinational Manufacturing Organizations *Proceedings of the Fifth International Symposium on Artificial Intelligence*, 333-340, 1992.