

Rule-based reasoning in an object-oriented manufacturing environment

H. Atabakhsh

*National Research Council of Canada, Institute
for Advanced Manufacturing Technology, Ottawa
Ontario, K1A 0R6, Canada*

Abstract

Rule-based reasoning has been integrated in a pure object-oriented environment taking advantage of the benefits offered by both paradigms by implementing a number of key expert system features in Smalltalk-80. A software toolkit has resulted called ExTool, which offers facilities for the user to create, modify and delete flexible production rules with no knowledge of the underlying programming language. In ExTool, decision-making is distributed rather than centralized in one large knowledge base. Inferencing is, therefore, opportunistic and is performed only on a subset of the relevant rules. The knowledge base is decomposed into a number of smaller knowledge bases representing specific decision-rules associated with only those objects involved in decision-making. Rules are activated through forward chaining inferencing, carried out by message passing. When the inference engine has insufficient information, it prompts the user for more input and continues with the new information. ExTool has explanation facilities for its reasoning, its production rules, and why it requires certain information. It has a friendly user interface which enables direct user-system interactions. ExTool has been successfully applied in an Enterprise Modelling application.

1. Introduction

The aim of this research has been to develop a flexible, generic, interactive and user-friendly decision support software system for Manufacturing applications. Flexibility in adding or modifying decision rules has been achieved through the use of Artificial Intelligence features such as production rules and inferencing mechanisms which are separate from the rest of the system. Genericity and

506 Artificial Intelligence in Engineering

re-usable components have been achieved through the concept of inheritance provided in Smalltalk-80. A user friendly interface for communication with the user has been developed using VisualWorks (a user interface builder for Smalltalk-80 from ParcPlace). The user interface is interactive in that when information is missing, the system prompts the user for input and provides explanation on its reasoning and production rules, whenever required. In addition, the system provides opportunistic reasoning by distributing the task of decision making among the objects.

The research described in this paper has resulted in a software toolkit called ExTool (Expert System Toolkit) and incorporates production rules into Smalltalk objects. Among other reasons for its selection, Smalltalk-80 code is flexible and re-usable, has a large library of classes and most of the features an object-oriented package should have (encapsulation, polymorphism, dynamic binding, etc.), is a relatively mature language, is a pure object-oriented language which helps to ensure one is truly programming in an object-oriented manner, and provides helpful programming tools such as debuggers, browsers and garbage collection. Goldberg[1] ExTool has a number of features which are not found in most other related systems studied, thus making it rather unique. These features together with an application in Manufacturing Enterprise Modelling, are described in the next section.

2. Object-Oriented Rule-based System

2.1. Distribution of Decision Making and Inferencing

The knowledge base consists of a fact base and a rule base where the fact base includes objects plus their instance variables and class variables. The rule base contains production rules having the following format:

IF <LHS> THEN <RHS>

The rules are implemented as Smalltalk-80 Objects and have a flexible style where both LHS and RHS may contain conjunctions. The LHS and RHS are each implemented as a class in Smalltalk. Both LHS and RHS may contain hypotheses and internal variables (instance variables and class variables), and can

directly send messages to objects. The LHS typically sends a message to retrieve a value but it may also contain functions (implemented as Smalltalk methods) which evaluate to a boolean. These functions can be used when a more complex premise is required. While the RHS typically sends a message to perform a task (which could be to modify a value), it may also contain actions (implemented as Smalltalk methods). Either LHS or RHS can therefore trigger the execution of one or more Smalltalk methods.

Distributed intelligent objects are created by associating a knowledge base with specific objects (called "intelligent objects"). At any given time, the inference engine will consider only the rules associated with the current object, rather than all the rules in the model, thereby allowing opportunistic reasoning. These intelligent objects can also cooperate through message-passing and participate in group decision-making. There is a rule base belonging to each object that needs decision making capabilities. In other words, each object needing decision-making capabilities has its own rule base of production rules for the decision making process. The knowledge base consists therefore of many small rule bases rather than one large rule base.

The rules are activated by a forward chaining inferencing strategy, using one of the options: deduce all facts, prove a hypothesis, or suggest a hypothesis. Deduce All Facts: the system can deduce all facts that are possible from the current set of facts and update the fact base. Prove Hypothesis: the system can try to prove a hypothesis. A list of hypotheses that are not yet proven is shown to the user. He/she can then choose a hypothesis from the list or add a new hypothesis to the list. Suggest Hypothesis: the user can interactively suggest a hypothesis. A list is shown to the user from which he/she can choose, or he/she may decide to add new entities. The system will then consider this new information and automatically deduce all possible facts to update the fact base.

Different inferencing methods can be implemented for different objects. For example, backward chaining could be implemented for an object from which other objects needing a backward chaining strategy would inherit. In the current version, forward chaining is the only inferencing technique implemented.

508 Artificial Intelligence in Engineering

The main reason for this is that the current target application areas are related to decision-making and in this type of application, inferencing requiring forward chaining seems to be more prevalent. In a diagnostic application, for example, a backward chaining strategy might be more suitable.

When the system requires a value for a hypothesis, or a variable whose value is unknown (e.g. to prove a hypothesis or deduce all facts), it will prompt the user for the value, and will then continue inferencing, taking the new input value into consideration. When the system prompts the user for a value, the user can ask the system why it requires the information by selecting the button labeled *Why*. An English description of the production rule which is currently being evaluated, and which needs the missing information will be displayed. The system can provide an explanation of the production rules in plain English, instead of in Smalltalk syntax. The system can also explain its reasoning by giving a list of those production rules that were fired (in the order they were fired) to arrive at a conclusion, or to prove a hypothesis.

2.2. Main User Interface and Rule Edition

A window-based user interface facilitates user-system interactions. It consists of a window containing two scrollable panes, and a menu bar (Figure 1). The bottom pane contains an English explanation of all rules that can be used in the currently selected knowledge base. The menu bar includes: options for triggering forward chaining on the current rule base (deduce all facts, suggest a hypothesis), options for providing explanations (on reasoning, on production rules, on information requirements), options for displaying information such as current hypotheses which are true or false, current knowledge base, options for changing current hypotheses values interactively, options for rule creation, modification, deletion, restoring deleted rules and rule selection (the selection of current set of rules is useful for testing various scenarios), and an option for deleting rules temporarily (this task is not destructive and the rule can be brought back easily through another option *restore rule*). There is also a *help* option which provides an explanation on each of the options in this window.

In order to add or modify a rule, the user can choose the option Rule Editor - add or modify, in the main user interface. Rule creation and modification uses two user interfaces of the same type called the Rule Editor (Figure 2) and the Rule Comment Editor, for entering a rule and comment for a rule, respectively. There is no need for the user to know Smalltalk in order to add or modify production rules.

An existing rule can be modified by selecting the option *Rule Editor - modify*, in the main user interface. The user is shown a list of existing rules to choose from. This time the Rule Editor window opens, displaying the rule that the user has selected to modify. The user can enter modifications in the rule (rule editor) as well as the comment of the rule (rule comment editor), then save the rule with the current modifications.

2.3. Application in Manufacturing

ExTool has been applied to, and tested with, an Enterprise Modelling System (EMS) which is being jointly developed by a consortium of Canadian companies and the National Research Council of Canada. The objective of the EMS is to provide a set of tools for the creation of models of enterprises, including both business and manufacturing operations Atabakhsh[2], Chan[3].

Using the capabilities of ExTool, the Enterprise Model can be used as a test-bed for comparing decision-making strategies and fine-tuning those decision-rules related to the selected strategy. The interactivity provided by the rule editor and the explanation facilities of ExTool enable the decision-maker to test out any innovative ideas he may have before actual implementation, while at the same time, he can iteratively formulate his decision rule set and get immediate feedback from the model regarding its implications. ExTool not only allows Artificial Intelligence features to be made available to any object requiring decision-making capabilities in the system, but it also permits intelligence and capabilities to be inherited through a class hierarchy.

Decision rules have been added in the Enterprise Modelling System through the Rule Editor described in the previous sections. Examples of these rules in-

510 Artificial Intelligence in Engineering

clude rules representing start and stop conditions for a task and selection of the next task from the input queue of a resource. The object to which the first type of rule base belongs is Task while the object to which the second type of rule belongs is Resource. Through the use of ExTool, a non-programmer can modify or add rules very easily without having to write code in Smalltalk.

4. Conclusions and Future Directions

Rule-based reasoning has been integrated in a pure object-oriented environment taking advantage of the benefits offered by both paradigms by implementing a number of key expert system features in Smalltalk-80. A software toolkit has resulted called ExTool, which offers facilities for the user to create, modify and delete flexible production rules with no knowledge of the underlying programming language. An important difference between ExTool and many traditional Expert System Shells is that in ExTool, decision-making is distributed rather than centralized in one large knowledge base. Inferencing can, therefore, be opportunistic and is performed only on a subset of the relevant rules. In fact, the knowledge base is decomposed into a number of smaller knowledge bases representing specific decision-rules associated with only those objects involved in decision-making. Rules are activated through forward chaining inferencing, carried out by message passing. When the inference engine has insufficient information, it prompts the user for more input and continues with the new information. ExTool has explanation facilities for its reasoning, its production rules (in plain English), and why it requires certain information. It has a friendly user interface which enables direct user-system interactions. ExTool has successfully been applied in an Enterprise Modelling System to allow for flexible decision making in a Manufacturing Enterprise. Some of the future plans are the addition of features such as backward chaining, fuzzy rules, truth maintenance and conflict resolution strategies. As well, using ExTool in other application areas such as scheduling, are also planned.



REFERENCES

- [1] Goldberg A.; & Robson D. *Smalltalk-80: The Language*, Addison-Wesley, 1989.
- [2] Atabakhsh H. & Chan A. W. Object-Oriented Rule-Based Manufacturing Enterprise Simulation, *Proceedings of the Object-Oriented Simulation Conference*. Tempe, Arizona Jan. 24-26 1994.
- [3] Chan A. W., Pardasani A., Atabakhsh H. & Graefe U. Concurrent Engineering using an Enterprise Modelling System. in: *Advances in Industrial Engineering, 19: Concurrent Engineering - Methodology and Applications*, ed P. Gu and A. Kusiak. pp. 117-133, Elsevier Science Publishers B.V. 1993.

ACKNOWLEDGMENTS

The author would like to thank Ajit Pardasani and Albert Chan for the Application of ExTool in the Enterprise Modelling System.

©Canadian Government

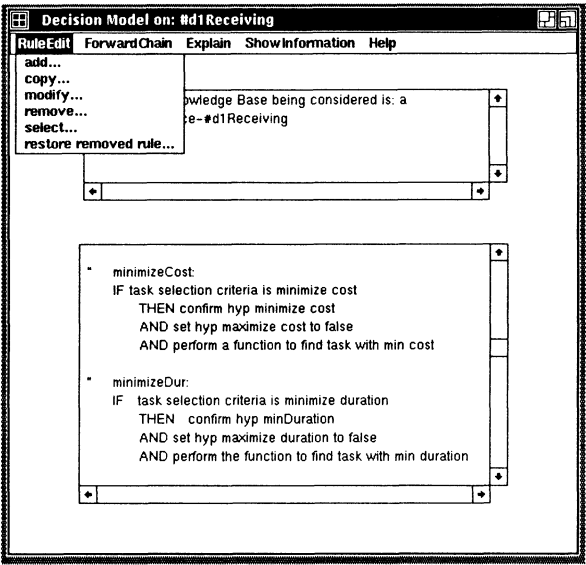


Figure 1: ExTool's Main user interface.

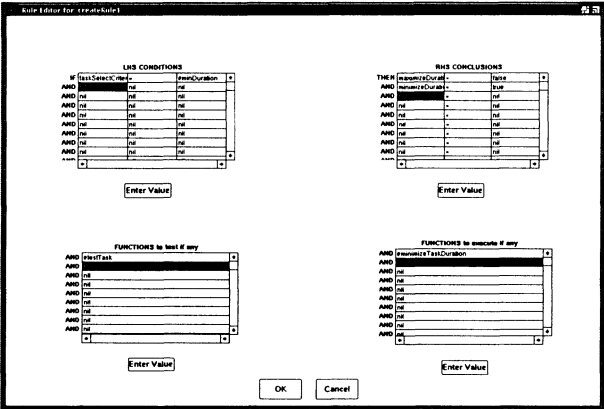


Figure 2: ExTool's Rule Editor