Intelligent Agent for Designing Steel Skeleton Structures of Tall Buildings

Zbigniew Skolicki[1]
Rafal Kicinger[2]

Abstract

The paper discusses a study on the application of intelligent agents (IAs) to conceptual designing. It provides an overview of the state-of-the-art in the areas of ontologies and IAs. Next, the system *Disciple*, a learning intelligent agent shell, and the system *Inventor 2001*, evolutionary design support tool, both developed at George Mason University, are briefly presented. Further, the paper introduces the developed ontology for a class of steel skeleton structures of tall buildings. This ontology was used to build an IA for the selection of initial parent design concepts in evolutionary designing. A description of the developed agent is provided as well. Finally, examples of design concepts proposed by the agent are presented. The paper also contains conclusions and recommendations for further research.

1. Introduction

Recent advances in Artificial Intelligence and the growth of computer power have resulted in a search for automation in the areas that were previously reserved for humans. A new methodology of intelligent agents substitutes old, simple and highly specialized systems with knowledge-acquisition based learning architectures. The characteristics of the new systems are described by Tecuci (Tecuci 1998). This new approach requires the system to possess some degree of self-independence, taking decisions upon the information collected. The reasoning, however, should be "transparent" and explicit to the user. Naturally, one would expect such an agent to have a huge repository of knowledge and proper mechanisms to maintain it. On the other hand such an approach would result in searching vast search spaces. Thus, some heuristics are often used to speed up the process.

The new methodology of agent based-systems no longer requires that the agents have to be programmed by highly skilled knowledge engineers. The agent, instead, learns the facts and processes by interacting with a user who is to some extent

[1] Ph.D. student, Computer Science Department, IT&E School, George Mason University, Fairfax, VA 22030; zskolick@gmu.edu
[2] Ph.D. student, Civil, Environmental and Infrastructure Engineering Department, IT&E School, George Mason University, Fairfax, VA 22030; rkicinge@gmu.edu

Citation:
Skolicki, Z., and Kicinger, R. "Intelligent agent for designing steel skeleton structures of tall buildings." Computing in Civil Engineering. Proceedings of the International Workshop on Information Technology in Civil Engineering, Washington, DC, November 2-3, 2002, A. D. Songer and J. C. Miles, eds., Reston, VA, 25-35.

ignorant to the matters of computer programming. Therefore, an agent should be able to conduct a dialogue with the user and gather extra information by asking easy to understand questions. Communication in natural language is probably the most suitable method. However, regardless of how easy it is to "talk with an agent", many inconsistencies and much incompleteness must surely appear in the knowledge base due to human errors and simplifications in modeling of the reasoning processes. Hence, one must be prepared for such situations and construct the agent so that it can reason with a level of knowledge that models the world only partially and sometimes maybe even incorrectly. Obviously, it is not an easy task to create such a system. When it is done, however, the benefits are unquestionable, since the agent can perform the most tedious tasks on its own.

2. Ontologies and Intelligent Agents for Design

2.1 Intelligent Agents

Although there is no agreement among researchers on what exactly constitutes an agent, there is generally an agreement that an agent should be characterized by several of the following features:

- *autonomy, decision making* – an agent should base its behavior on the internal state, and should take decisions on its own, being to some extent independent (Leitao and Restivo 2000; Wooldridge and Jennings 1995).
- *social ability, communication and cooperation* – an agent should be able to communicate with other agents or humans to achieve the goal, possibly in a mixed-initiative way (Leitao and Restivo 2000; Wooldridge and Jennings 1995).
- *monitoring, perceiving* - an agent should sense and observe the environment within which it is located. (Leitao and Restivo 2000; Russell and Norvig 1995)
- *reactivity* – an agent should adapt to the changes in the environment (Wooldridge and Jennings 1995)
- *acting, operational control* - an agent should perform chosen actions (Leitao and Restivo 2000; Russell and Norvig 1995)
- *knowledge, ontology* – an agent needs to have an understanding of the environment to perform the task (Nwana and Ndumu 1999).
- *learning* – an agent should enhance its behavior during its lifespan (Nwana and Ndumu 1999; Tecuci 1998).
- *continuity* – some researchers require an agent to be a continuously running process (Franklin and Graesser 1997).

Agent systems can be roughly divided into two categories – multi-agent systems and autonomous interface/information agents (Nwana and Ndumu 1999). The division between the two categories is not absolute, since agents in both categories can share some characteristics, like learning. Multi-agent systems are very often used in situations requiring either the use of distributed knowledge, or multi-objective planning. Research in this area stresses the issue of cooperation between agents and this goal is usually realized by means of negotiation (Anumba et al. 2002).

Interface agents, on the other hand, are designed to be assistants which monitor users' behavior, infer some rules, and suggest further actions (Tecuci 1998). They are often used to aid users faced with a huge amount of information. This is not a simple task, since giving some useful and proper advice requires a thorough understanding of the problem domain. Thus, the knowledge must be appropriately stored in meaningful, flexible and expandable repositories.

2.2 Ontologies

Agents performing complex tasks and interacting with their environments must contain extensive knowledge and understanding about their task domain. There are generally two approaches to the problem. First, one can try to build a huge all-knowledge ontology that can be used for most problems. Second, smaller and more domain specific ontologies for each specific problem could be built.

Probably the most widely known attempt to build a vast reusable ontology is the *CYC* project (Lenat 1995) started in 1984. The *Ontolingua* project, developed at Stanford (Farquhar et al. 1996), aims to create a shareable and reusable repository of knowledge. Big ontologies must store the information in an organized, possibly hierarchical way. It's not uncommon to have two complementary parts: a taxonomy component to maintain the hierarchy of entities and the relations between them, and a rule component for expressing expert knowledge (MacGregor 1991; Tecuci 1998). The *Loom* system (MacGregor 1991), and its successor *PowerLoom* are examples of knowledge representation tools developed to help construct intelligent software applications (Fikes and Farquhar 1997; MacGregor 1991). *Disciple*, described in the following section, is an advanced system developed at George Mason University, that shares properties of both knowledge-acquisition and machine learning systems, and enables the creation of intelligent agents by simple expert-agent interaction process. Another example of a system designed to create ontologies is *Protégé-2000*, primary developed for medical purposes (Grosso et al. 1999).

There have also been attempts to join separated knowledge-based systems. One of the examples in engineering domain could be PACT experiments (Cutkosky et al. 1993).

3. Disciple - Learning Agent Shell

Disciple is one of the most advanced systems for creating intelligent agents. This system was developed in the Learning Agents Laboratory at George Mason University (Tecuci et al. 2001). *Disciple* can be generally classified as a learning agent shell that is a tool for building intelligent agents. It has three major components: a knowledge base, a problem solving engine, and a learning engine. The knowledge base consists of two parts: 1) object ontology, and 2) rules, cases, and methods. The problem solving engine contains programs that manipulate data structures stored in the knowledge base in order to perform inference. The learning engine implements methods for extending and refining knowledge in the knowledge base. It uses multi-strategy learning that combines several single-strategy machine learning techniques.

Initially the learning agent's knowledge base is empty. Building an agent using the shell consists of customizing the shell and developing the knowledge base. The authors decided to use this tool to build an intelligent agent for *Inventor 2001*, an evolutionary design support tool (described shortly in the following section) for generating conceptual and detailed wind bracing designs.

4. Inventor 2001 – Evolutionary Design Support Tool

Inventor 2001 is an evolutionary design support tool, developed at George Mason University, for generating both conceptual and detailed designs of wind bracings in steel skeleton structures (Murawski et al. 2001). Designing wind bracings in tall buildings is one of the most difficult and time-consuming tasks for structural engineers because of the complexity of the problem and the difficulty in finding formal selection and evaluation models (Mustafa and Arciszewski 1992). Therefore, modern design support tools like *Inventor 2001* may significantly improve the process of designing complex structures and enhance designers' capabilities of producing better, innovative designs.

Inventor 2001 uses evolutionary algorithms for exploring large and complex search spaces. The evolutionary computation approach has proved to be especially suitable for producing solutions for these types of problems. The system has seven major components:

1. Evolutionary Computation Component.
2. Feasibility Filter.
3. Structural Analysis, Design and Optimization Component (SODA).
4. Wind Forces Analyzer (WindLoad).
5. Evaluator.
6. Statistical Component.
7. Visualization Component.

A detailed description of the system is provided by Murawski et al. (Murawski et al. 2001).

5. Ontology of Steel Skeleton Structures for Inventor 2001

The objective of the work on the ontology of steel skeleton structures of tall buildings was to create a knowledge representation of design objects used by *Inventor 2001*, but not to develop an exhaustive ontology of tall buildings. Thus, only the entities necessary for conceptual representation of wind bracings structures were modeled and unnecessary details omitted. The ontology contains objects starting from the very physical level of structural elements of a tall building (like beam, column, diagonal, etc.), up to more abstract concepts of logical components (story, bay, truss, etc.), and entire buildings structures in general (16-story building, etc.). Moreover, notions of connections types, static characters of joints, etc. were added. Since *Inventor 2001* uses Evolutionary Computation as an underlying optimization paradigm, some concepts and instances related to the EC representation, like Inventor population size, Inventor initial design, etc. were also included. The detailed description of the developed ontology is presented below.

5.1 Building

One of the most important objects in the ontology is obviously a concept representing a building. Buildings have different heights and the proper information must be stated in the ontology. The general concept representing a building is simply called `Building`. The building height is one of the most important factors determining the building structure. Therefore three subconcepts have been added to `Building`: `Low_Building`, `Medium_Building` and `High_Building`. In *Inventor 2001* one can define 7 possible buildings heights, and hence 7 subconcepts of specific building heights were defined and grouped within the above-described concepts. Thus, `16_Story_Building` and `20_Story_Building` concepts were classified to belong to the concept `Low_Building`. Similarly, concepts of `24_Story_Building`, `26_Story_Building` were considered to be a `Medium_Building`, while `30_Story_Building`, `32_Story_Building` and `36_Story_Building` were subconcepts of a `High_Building` concept. Each of these subconcepts had a specific instance defined to belong to it, for example the concept `16_Story_Building` has an instance *`16_story_building_01`*, etc.

5.2 Logical Component

Every tall building is a complex structure consisting of many elements. Fortunately, many elements can be organized into subgroups based on their function and/or placement within the structure. In order to properly represent these subgroups, a concept called `Logical_Component` has been created. One of the most obvious subgroups is `Story`. The subconcept `Story` was added under `Logical_Component`. Next, 36 specific instances called *`story_01`*, …, *`story_36`* were added, to represent all possible stories that could be designed using *Inventor 2001*.

Another obvious choice for a logical component is `Bay`. *Inventor 2001* considers only three-bay buildings, so a concept `Bay` was introduced under `Logical_Component`, together with 3 subconcenpts of `Bay`: `Left_Bay`, `Middle_Bay`, and `Right_Bay`. Each of these subconcepts contains a specific instance, for example `Left_bay` has an instance `left_bay_01`.

In a similar way, remaining subgroups like `Vertical_Truss`, `Horizontal_Truss`, and `Ground` have been defined.

5.3 Structural Component

Structural elements that form steel skeleton structures can be divided into four major groups: beams, columns, diagonals, and ground connections. As *Inventor 2001* uses this representation, a concept `Structural_Element` and 4 subconcepts: `Beam`, `Column`, `Diagonal`, and `Ground_Connection` have been defined as well. Also, all the instances were added for each of the subconcepts to represent an entire set of structural elements forming the structure. Thus, the concept Beam contained instances *`beam01_left`*, *`beam01_middle`*, *`beam01_right`*,

beam02_left, etc., where the numerical index denotes the story to which the beam belongs. Similarly, instances for concepts `Column`, `Diagonal`, and `Ground_Connection` were created. Figure 1 presents a sample story with all the structural element representations in the ontology. Spatial relations between elements were encoded using features described in section 5.6.

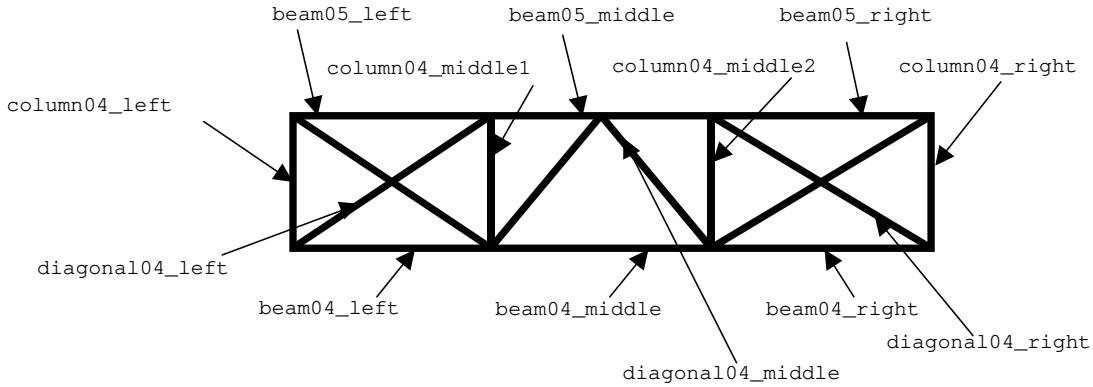

Figure 1. Structural elements in the ontology shown for a sample story

5.4 Element Type

Structural elements described in the previous section can have different structural characteristics. Hence, appropriate concepts needed to be specified in the ontology to describe these characteristics. A concept `Element_Type` with corresponding subconcepts for each structural element type were created, resulting in the `Beam_Type`, `Ground_Connection_Type`, and `Diagonal_Type` subconcepts. Each of these subconcepts were further subdivided into more detailed subgroups as shown in Figure 2.

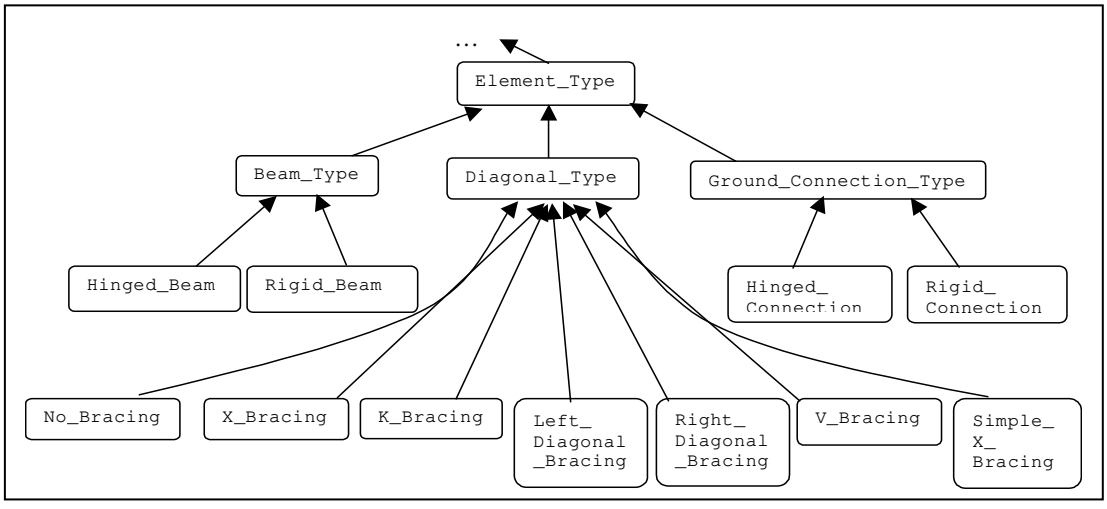

Figure 2. Part of the ontology representing Element_Type concept and its subconcepts

Also, specific instances of each of the subconcepts within this category were added to the ontology.

5.5 Inventor Population Size and Inventor Initial Design

Some of the EC notions implemented in *Inventor 2001* were represented to define the initial group (population) of designs as well as individual designs. The size of a population depends on an arbitrary choice and is based on previous experience with *Inventor 2001*. A decision was taken to define concept `Inventor_Population` and its subconcepts: `Small_Population`, `Medium_Population`, and `Large_Population`. Instances were added to each of these subconcepts. Also, a concept `Inventor_Initial_Design` was defined along with its subconcepts `First_Design`, `Second_Design`, ... , `Fifth_Design`. Again specific instances were added to each of these subconcepts. Using features described in the next section, appropriate design concepts were assigned to corresponding population sizes. Thus, `Small_Population` contains `First_Design` only, `Medium_Population` consists of `First_Design`, `Second_Design`, and `Third_Design`, and finally `Large_Population` has all five designs.

5.6 Features

All the above-mentioned ontological hierarchies represent concept-subconcept-instance relation, more commonly known as "ISA" relation. However, this relation is not sufficient to represent objects' properties, and other relations between objects. In *Disciple* one can model these properties and relations using so-called features. Many features describing the characteristics of a design and relations between the components were created. One of the most important was *part_of* feature. It represented the containment relation among structural elements, logical components, and buildings. To represent spatial relationships among elements and components, the following features were created: *is_above*, *is_below*, *is_right_of*, *is_left_of*, and defined relations among appropriate concepts and instances. Also, the features *component_of*, and *consists_of* were used to link *Inventor_Initial_Design*, *Inventor_Population*, and *Building*.

6. Learning

The developed ontology served as a basis for the learning and problem solving capabilities of the intelligent agent. The learning process consisted of a few parts. The first, called the modeling phase, consisted of consecutive questions in natural language that elicited knowledge from an expert. During this phase the initial model of the problem solving process was developed. Figure 3 presents a fragment of the modeling process implemented in IA.

Define the designs of tall buildings for the initial population of Inventor2001
I need to perform
Define the designs of tall buildings for the initial population of Inventor2001
Question: What is the height of the building?

Answer: It is a 16-Story_building
Therefore I need to perform
Define the designs of 16-Story_building for the initial population of Inventor2001
Question: What is the size of the initial population of Inventor2001?

Answer: The population size is Medium_population
Therefore I need to perform
Define the designs of 16-Story_building for the initial population of Inventor2001, which is the Medium_population
Question: Which design in the Medium_population do you want to define?

Answer: I want to define Design_01
Therefore I need to perform
Define Design_01 of 16-Story_building
Question: What are the requirements for the static character of joints you want to impose?

Answer: I want to use Rigid_beam and Hinged_beam
Therefore I need to perform
Define Design_01 of 16-Story_building, which uses Rigid_beam and Hinged_beam
Question: What are the requirements for the number of bays entirely occupied by bracings you want to impose?

Figure 3. Fragment of the modeling process implemented in IA

The second part of the learning process consisted of the formalization phase. In this part, all instances from the ontology were replaced by variables. Formalization is a first step towards generalization of agent's knowledge.

With the formalized modeling process completed the process of rules learning could begin. These rules would enable the agent to intelligently guide the user in selecting appropriate designs. One of the big advantages of the *Disciple* approach is the learning by explanations strategy. It provides the agent with detailed explanations of the relationships between different elements in designs based on the concepts, instances, and features encoded in the ontology.

Having all rules defined, the agent was able to generate examples. However, not all of them were correct. Thus, the next stage was the refinement phase, in which the agent was provided with feedback on which of the generated designs were correct and which were not. Every correct example was supplied with an explanation that was later used in analogical reasoning. Inappropriate examples generated by the agent were rejected. This action resulted in either specializing the previously learned rule, or in adding the inappropriate example to the set of negative examples (negative exceptions).

After going through all the stages, the agent was able to generate solutions. They were given in a natural language form and had to be transformed into a

representation acceptable to Inventor. Although *Disciple* could theoretically provide the appropriate output, it would require building much more complex ontology extended to suit the implementation details. Thus, a decision was taken to write a small translator that transformed the natural language description into a data structure appropriate for *Inventor 2001*.

7. Results and Conclusions

As described in the previous sections, the intelligent agent was able to generate examples of steel skeleton structures using the knowledge contained in its knowledge base and by applying appropriate rules learned during the modeling of the problem solving process. A sample solution generated by an agent, as well as the Inventor 2001 design obtained from this solution, are shown in Figure 4.

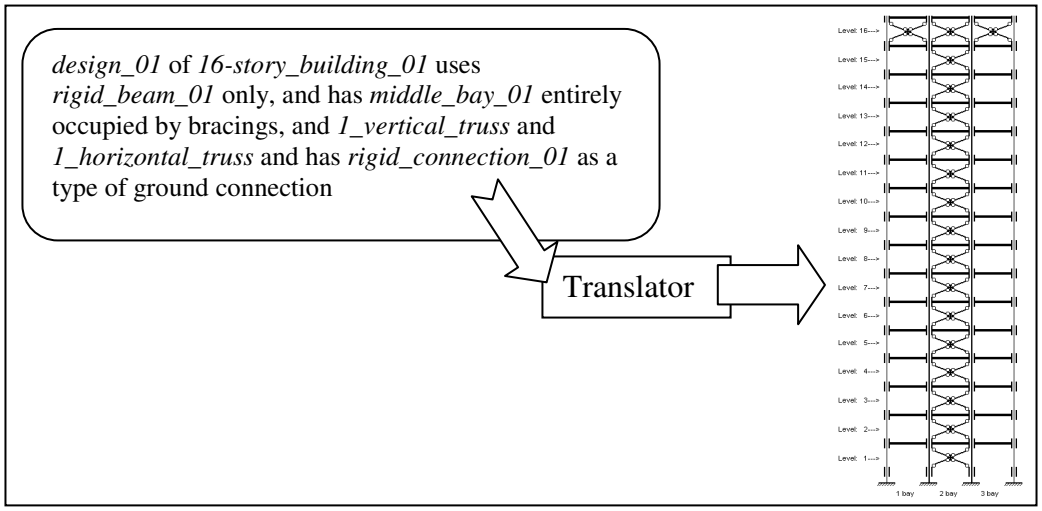

Figure 4. Sample solution generated by agent and its representation in Inventor 2001

Intelligent agent was also able to generalize previously learned rules. For example, when the problem solving process for 16-story buildings was modeled and the upper limit of the application of the rules was defined to include the concept `Low_building`, the agent was able to generalize and apply the same rules for 20-story buildings.

This work had a preliminary character and its main research focus was to estimate the possible potential of using intelligent agents for conceptual designing. The main purpose was to better understand the process of building modern agents using the intelligent agent shell and to build a repository of knowledge. Additionally, the authors wanted to understand the mechanisms, advantages and limitations of the *Disciple* approach.

The agent was taught basic rules concerning the conceptual designing of tall buildings. All the phases of this approach (modeling, formalization, rule learning, rule refinement and solution generation) have been accomplished using *Disciple*, and resulted in an agent able to generalize the process of tall building design.

8. Further Work

The study has shown the application of the intelligent agent approach to conceptual designing. On the other hand, only a very general level of engineering knowledge was modeled, and further research is necessary to estimate its usefulness for more complex design tasks. Another important topic for further research is a complete integration of knowledge-based tools (intelligent agents, knowledge repositories) with modern design optimization systems. In the authors' opinion such integrated tools would significantly advance the engineering design process.

Acknowledgements

The authors gratefully acknowledge the support for their research from the NASA Langley Research Center under the Grant 01-1231. They also greatly appreciate help and support from our advisor Dr. Tomasz Arciszewski, who has always had time for us and was our "civil engineering oracle." The authors would also like to thank Dr. Gheorghe Tecuci and members of the Learning Agent Laboratory for the opportunity to use *Disciple* in this research, and their help during the process of the development of the intelligent agent.

References

Anumba, C. J., Ugwu, O. O., Newnham, L., and Thorpe, A. (2002). "Collaborative Design of Structures Using Intelligent Agents." *Automation in Construction*, 11, 89-103.

Cutkosky, M. R., Englemore, R. S., Fikes, R., Genesereth, M. R., Gruber, T. R., Mark, W. S., Tenenbaum, J. M., and Weber, J. C. (1993). "Pact: An Experiment in Integrating Concurrent Engineering Systems." *IEEE Computer*, 26(1), 28–37.

Farquhar, A., Fikes, R., and Rice, J. (1996). "The Ontolingua Server: A Tool for Collaborative Ontology Construction." *Proceedings of the Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada.

Fikes, R., and Farquhar, A. (1997). "Large-Scale Repositories of Highly Expressive Reusable Knowledge." *KSL-97-02*, Knowledge Systems Laboratory, Stanford.

Franklin, S., and Graesser, A. (1997). "Is It an Agent, or Just a Program?: A Taxonomy for Autonomous Agents." *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*, Budapest, Hungary, 21-35.

Grosso, W. E., Eriksson, H., Fergerson, R. W., Gennari, J. H., Tu, S. W., and Musen, M. A. (1999). "Knowledge Modeling at the Millennium (the Design and Evolution of Protege-2000)." *Proceedings of the Twelfth Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Alberta, Canada.

Leitao, P., and Restivo, F. (2000). "A Framework for Distributed Manufacturing Applications." *Proceedings of the Advanced Summer Institute International Conference, ASI' 2000*, Bordeaux, France, 75-80.

Lenat, D. B. (1995). "Cyc: A Large-Scale Investment in Knowledge Infrastructure." *Communications Of The ACM*, 38(11).

MacGregor, R. (1991). "Using a Description Classifier to Enhance Deductive Inference." *Proceedings of the Seventh IEEE Conference on AI Applications*, Miami, Florida, 141—147.

Murawski, K., Arciszewski, T., and De Jong, K. A. (2001). "Evolutionary Computation in Structural Design." *Journal of Engineering with Computers*, 16, 275-286.

Mustafa, M., and Arciszewski, T. (1992). "Inductive Learning of Wind Bracing Design for Tall Buildings." Knowledge Acqusition in Civil Engineering, T. Arciszewski and L. Rossman, eds., American Society of Civil Engineers, New York, 190-203.

Nwana, H. S., and Ndumu, D. T. (1999). "A Perspective on Software Agents Research." *The Knowledge Engineering Review*.

Russell, S. J., and Norvig, P. (1995). *Artificial Intelligence: A Modern Approach*, Prentice Hall, New Jersey.

Tecuci, G. (1998). *Building Intelligent Agents: An Apprenticeship Multistrategy Learning Theory, Methodology, Tool, and Case Studies*, Academic Press.

Tecuci, G., Boicu, M., Bowman, M., and Marcu, D. (2001). "An Innovative Application from the Darpa Knowledge Bases Programs: Rapid Development of a High Performance Knowledge Base for Course of Action Critiquing." *AI Magazine*, 22(2).

Wooldridge, M. J., and Jennings, N. R. (1995). "Intelligent Agents: Theory and Practice." *The Knowledge Engineering Review*, 10(2).