In Proceedings of IJCAI-2001 Workshop on Empirical Methods in AI, pp. 47-53. Seattle, Washington, August. AAAI Press, Menlo Park, CA, 2001.

Evaluation of Mixed-Initiative Knowledge Base Development Methods and Tools

Ping Shyr*, Gheorghe Tecuci, Mihai Boicu
Learning Agents Laboratory, Department of Computer Science, MS 4A5
George Mason University, 4400 University Drive, Fairfax, VA 22030-4444
Shyrp@nasd.com,{tecuci, mboicu}@gmu.edu, mboicu@gmu.edu, http://lalab.gmu.edu
*Also with NASD Regulation, Inc.

Abstract

This paper presents a general framework for the empirical evaluation of mixed-initiative knowledge base development methods and tools. It also illustrates it with a knowledge acquisition experiment performed to validate a mixed-initiative method for acquiring problem solving rules directly from a subject matter expert. The main feature of this framework is that the evaluation is an integral part of system design and prototyping. The framework includes a repertoire of evaluation methods, guidelines for designing experiments based on these methods, and reusable evaluation utilities.

1 Introduction

Mixed-initiative methods and tools synergistically integrate human and automated reasoning to take advantage of their complementary reasoning styles and computational strengths. Such approaches represent a very promising avenue for the development of powerful man-machine systems. However, because these are relative new approaches, there is no well-established evaluation methodology.

Already the existing methodologies for the evaluations of knowledge-based systems are very complex, costly and labor-intensive [Adelman and Riedel, 1997, Cohen, 1995]. The evaluations of mixed-initiative knowledge-based systems raise additional difficulties. For instance, one has to deal with the added complexity of the mixed-initiative execution of a task which involves several activities taking place in parallel and a dynamic shifting of the initiative and control between the human and the system. More importantly, however, it is difficulty to assign the credit or the blame for a specific result between the human and the system, because both of them contributed to it in a complex way.

For several years, the GMU Learning Agents Laboratory is developing a mixed-initiative approach to the development of knowledge bases and agents. This approach, called Disciple [Tecuci, 1998], is indented to allow a subject matter expert that does not have prior knowledge engineering experience, to build a knowledge base by himself, with no or very limited support from a knowledge engineer.

Evaluation of the developed Disciple methods was a primary concern for us, both because this represents the main way of demonstrating the utility of the Disciple approach to knowledge base development, and because it helps us in identifying the most promising research avenues to pursue.

In this paper we discuss our research on developing a general framework for the evaluation of mixed-initiative knowledge base development methods and tools. We first discuss the integration of the evaluation design into system's design and development. In particular, we present an architecture of Disciple where the evaluation is as basic a component of each module as it is its interface. That is, each main module of Disciple is designed from the very beginning to facilitate its evaluation. In section 3 we present a general evaluation framework that includes a repertoire of evaluation methods, ready-made evaluation utilities, and guidelines for the development of experimental set-ups. Then, in section 4 we present a knowledge acquisition experiment that illustrates this framework. In this experiment we demonstrated that a subject matter expert can teach a Disciple agent to create problem solving rules for a complex military application.

2 Evaluation during Design and Prototyping

We have developed a series of Disciple prototypes, each new prototype incorporating a more powerful knowledge representation, and more complex methods for problem solving and learning, than the previous prototype. However, evaluating these prototypes and getting an understanding of WHY one prototype is better than another has not been an easy task. One reason is that the evaluation has always been an end process that had to deal with the entire system as a black-box. While such an approach is appropriate for evaluating the end-to-end performance and usability of the system, it is not well-suited to clarify the reasons for system's behavior. This is even more critical in the case of a mixed-initiative system where its good (or bad) behavior might be due to the qualifications of the user, and not to some specific system features. For that reason, we have changed the way we approach system's evaluation, making it an integral part of system's research, design and prototyping. This has significantly changed the architecture of our research prototypes, as illustrated in Figure 1.

First, each basic system module, such as the Rule Learning Module, is designed in a modular way that allows independent or combined evaluation of its features. This, for instance, allows the performance of tool ablation experiments where some of the features of a tool are disabled, allowing one to identify the causes for system's performance improvement or degradation.

Second, each basic system module contains its own data collection component that will gather relevant evaluation data for that module.

Third, the system contains a global evaluation module that receives data from the individual modules and performs individual module evaluation, data aggregation, and overall system evaluation.

This architecture allows us to easily perform a comprehensive evaluation of the system. This includes any-time evaluation (at module end, at phase end, and at system end), any-user evaluation (i.e., for a specific user, for a group of users, or for all users), and any-knowledge evaluation (e.g. knowledge reuse, knowledge acquisition rates, knowledge base size, etc.).

The knowledge base of the Disciple agent contains a generic evaluation ontology. The nodes in this ontology correspond to entities for which evaluation data is collected and analyzed. For instance, there are nodes in this ontology corresponding to each user of Disciple, and to each relevant group of users, such as, the group of subject matter experts, the group of knowledge engineers, the group of end users, and the group of all users.

Associated with each user there is a user log file, a user behavior profile, a user knowledge profile, and a user KB profile. The user log file contains all the information related to the detailed system usage by a particular user (such as, login id, login time, window/module switching, what tool features was used and for how long, etc.). The user behavior profile is a summary of the user log file that characterizes more abstractly the behavior of the user by aggregating related information. Aggregated information includes the number of times the system was used, total time, time spent on each module, total number of errors, number of errors per module, various error rates, etc. The user knowledge profile contains the information related to the knowledge changes made by that user (e.g. addition / deletion / modification of knowledge elements). The user KB profile aggregates information for the entire KB (e.g. KB size, knowledge acquisition rate, amount of knowledge reuse).

Associated with each user group (e.g. the group of subject matter experts) there is a group level profile that aggregates data (usually by averaging) from the corresponding profiles of the users belonging to that group.

The system can be easily configured by selecting the types of data to be collected, and the types of processing to be performed with this data, through the evaluation control module (see bottom of Figure 1).

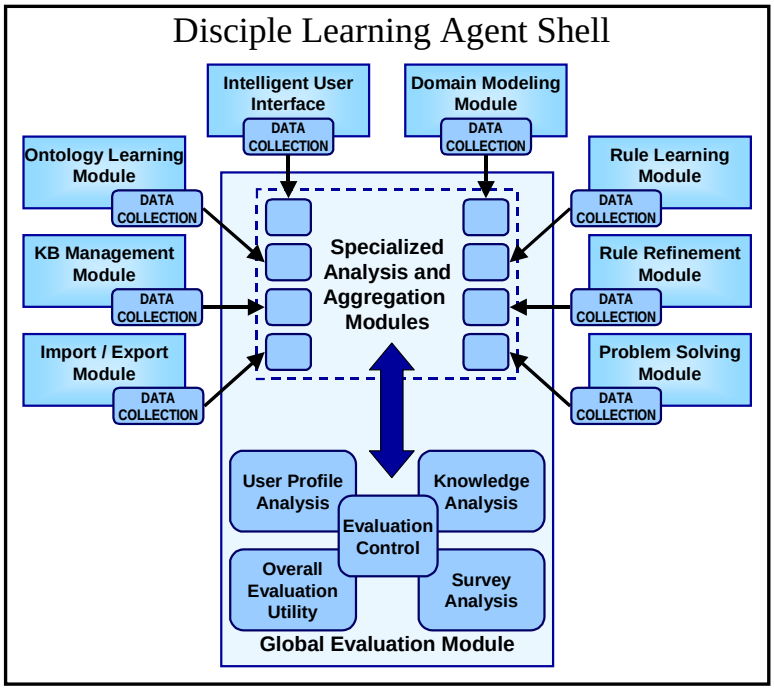

Figure 1: The evaluation architecture of Disciple

3 The Evaluation Framework

Disciple is a complex approach that covers all the stages of knowledge base development, and therefore any Disciple agent integrates a wide variety of modules with different functionalities (e.g. domain modeling, ontology import, ontology development, problem solving, rule learning, rule refinement, knowledge management, knowledge base export, etc.). There are also different developers. Most of them are Ph.D. students that do research on some knowledge base development aspect. Each developed module needs to be evaluated, and Disciple, as a whole, needs to be evaluated in a coherent way, even though each module may be the result of the research and development of a different individual. This has led us to the development of a general evaluation framework with the following characteristic features:

- it provides a repertoire of evaluation methods that are appropriate for the evaluation of the mixed-initiative knowledge base development methods and tools;
- it provides guidelines for designing experiments based on these methods;
- it provides ready-made evaluation programs that could be used in a certain evaluation method and experimentation set-up.

Based on this framework, each researcher develops his or her own evaluation model. Conversely, each new evaluation model may contain additional methods and evaluation programs that are used to extend the evaluation framework.

Figure 2 is a representation of this evaluation framework that shows its three main dimensions.

The first dimension concerns the stage and place of the experiment. This could be either a laboratory experiment or a field experiment. Laboratory experiments are performed with preliminary prototypes in order to gather feedback from the

research team and to build baselines for future evaluations. The field experiments take place in a near operational environment with a complete prototype.

The second dimension of the evaluation framework concerns the usefulness of the system, measuring its performance and its usability.

The third dimension concerns the type of experimentation methods used, which could be subjective methods or objective methods. Both subjective and objective data need to be collected and a cross-checking between them is a must.

As mentioned above, the framework also includes general evaluation guidelines, such as:

· Construct an evaluation model;
· Use multiple iterations of laboratory experiments and field experiments;
· Experiment with multiple domains;
· Collect and compare subjective and objective data;
· Evaluate alternative KB development methods and compare their results;
· Compare evaluation results with a baseline (or model answer). If the baseline does not exist, create one in a laboratory experiment;
· Utilize scoring functions when applicable;
· Handle the sample size issue if needed;
· Minimize transfer effects;
· Deal with possible order effects;
· Utilize the existing automatic analysis and evaluation modules/utilities.

4 A Knowledge Acquisition Experiment

In this section we present a knowledge acquisition experiment that we have performed and which illustrates the evaluation framework described above.

One of the goals of the Disciple knowledge base development approach is to enable rapid acquisition of competent task reduction rules from a subject matter expert, with limited or no assistance from a knowledge engineer. To achieve this goal, we have developed a mixed-initiative rule learning and refinement method that allows the Disciple agent to learn complex task reduction rules from an easy interaction with a subject matter expert. During this interaction, the subject matter expert provides a specific example of a task reduction operation. Then it helps the Disciple agent to understand why this reduction is correct by providing hints. Based on the received hints Disciple generates a list of plausible explanations from which the expert chooses the correct ones. Then, using the example and the found explanation, Disciple generates a task reduction rule. The rules learned in this way are used by the agent to solve new problems. During this process, the expert is shown each problem-solving step (rule application), being asked to accept or to reject it. If a problem-solving step is accepted by the expert, the rule that generated it is automatically generalized by the agent to cover this new positive example. If, however, the problem-solving step is rejected as incorrect by the expert, then the agent tries to understand why the example is wrong (again with the help of the hints provided by the expert), and specializes the rule

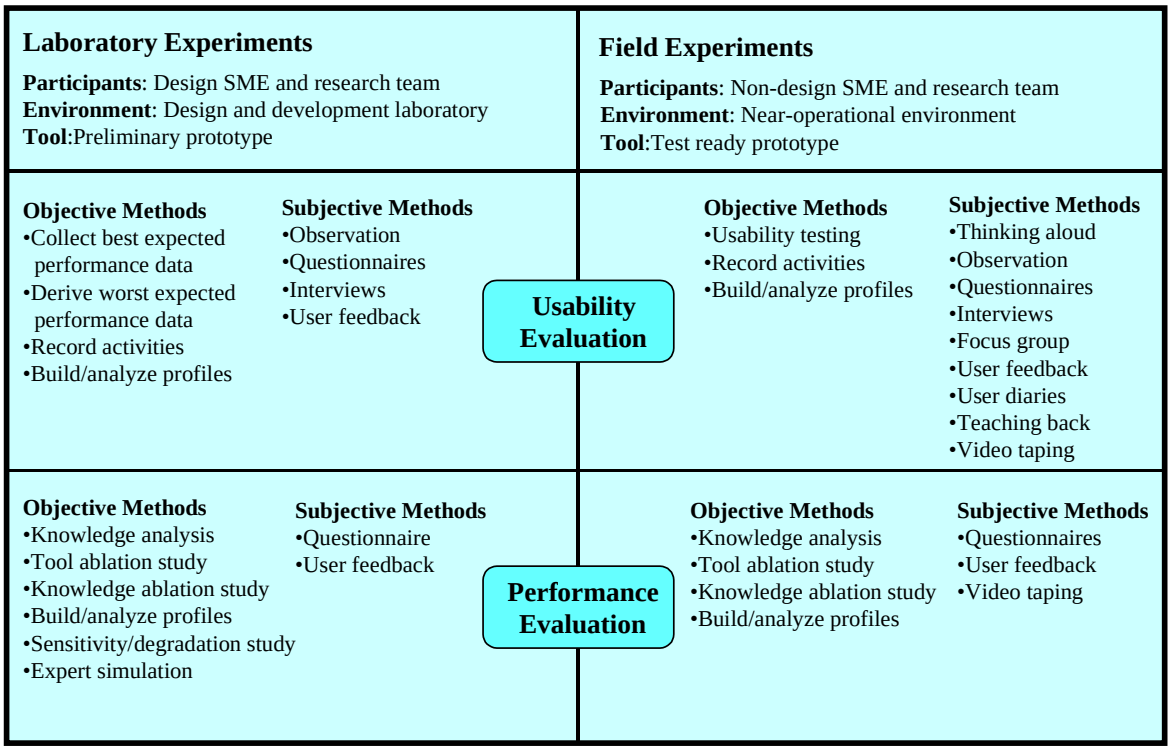

Figure 2: The evaluation framework.

accordingly. In this way the rules are improved, gradually leading to correct problem solving behavior. It is important to stress that the subject matter expert does not see or work directly with the rules learned by the agent, but only with specific problem solving examples and their explanations.

Figure 3 illustrates this knowledge acquisition task in the context of the Course of Action (COA) challenge problem used in the DARPA's High Performance Knowledge Bases program (Tecuci et al., 2001). A military course of action is a preliminary outline of a plan for how a military unit might attempt to accomplish a mission. The problem solving task addressed is to critique such courses of action. For instance, the left hand side of Figure 3 represents a sequence of three task reduction steps used to identify a strength in the course of action COA411 with respect to the Principle of Security. Each task reduction step consists of a task to be performed, a question that addresses some aspect of the performance of the task, an answer to the question, and a simpler task to which the first task is reduced, based on the answer. One could regard the task reduction steps in the left hand side of Figure 3 as an illustration of how a military expert might think when performing the top-level task. From each such task reduction step the Disciple agent learns a general task reduction rule. For instance, the rule in the right hand side of Figure 3 has been learned by the agent starting from the second task reduction step.

To test our claim that the method described above allows rapid acquisition of competent task reduction rule from a subject matter expert, we have performed a field knowledge acquisition experiment at the US Army Battle Command Battle Lab, in Ft Leavenworth, Kansas.

The subject matter experts that participated in the experiment were four Army officers that did not have any prior knowledge engineering experience.

The experiment was designed to test the following hypotheses:

Null Hypothesis: All the subject matter experts can rapidly (within one day) develop competent task reduction rules for critiquing courses of action with respect to two principles war, based on a given model of the problem solving process.

Alternative hypothesis: At least one subject matter expert cannot develop, within one day, competent task reduction rules for critiquing courses of action with respect to two principles war, based on a given model of the problem solving process.

The experiment had three phases: a joint training phase (day 1 to 3), an individual knowledge acquisition experiment (day 4), and a discussion of the experiment (day 5). The entire experiment was videotaped. The training for the experiment included a detailed presentation of Disciple's knowledge representation, problem solving and learning methods and tools. For the knowledge acquisition experiment itself, each expert received a copy of Disciple-COA with a partial knowledge base. This knowledge base contained a complete object ontology but no rules. We also provided the experts with the descriptions of three courses

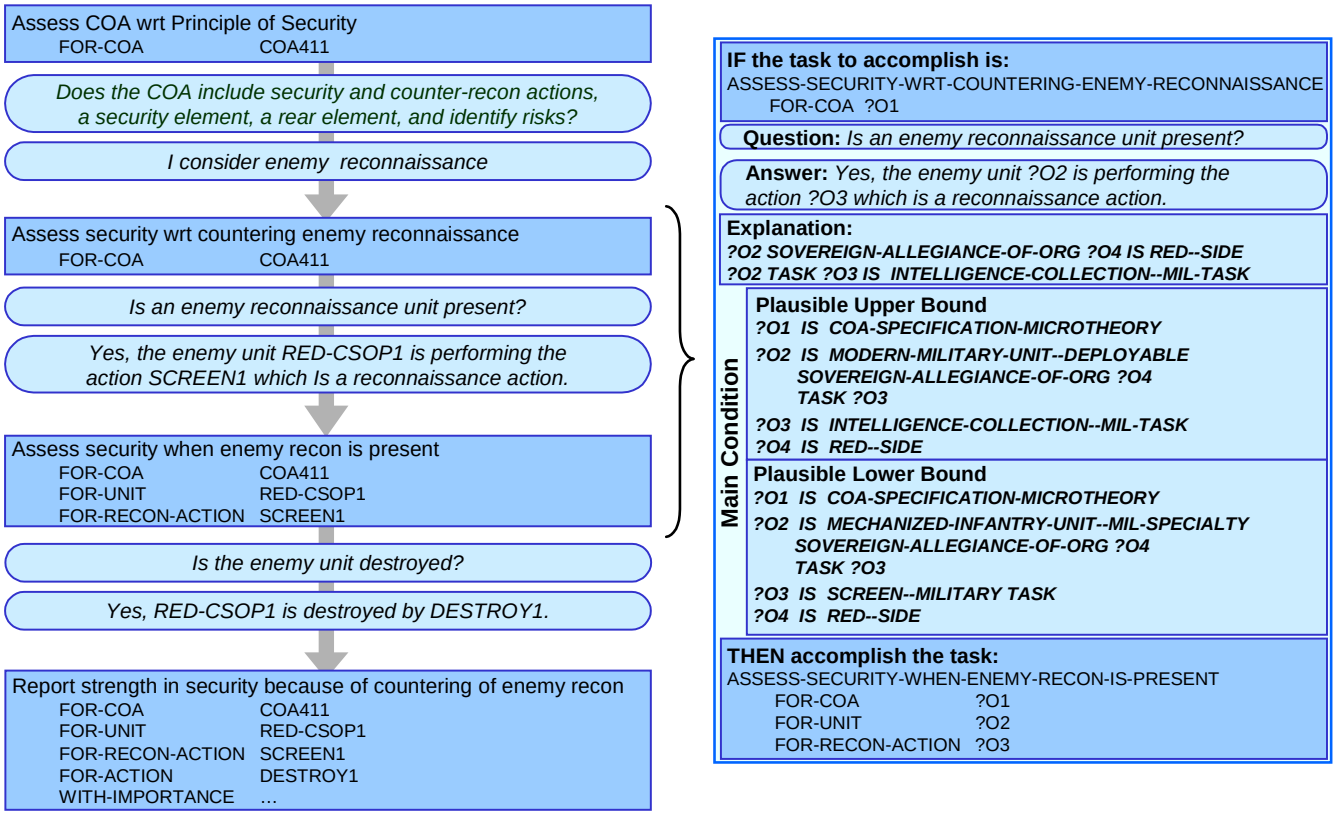

Figure 3: Illustration of the knowledge acquisition task.

of action, COA411, COA421, and COA51, to be used for training Disciple. Finally, we provided and discussed with the experts the modeling of critiquing these courses of action with respect to the principles of offensive and security. That is, we provided the experts with specific task reductions like the one from the left hand side of Figure 5, to guide them in teaching Disciple-COA. After that, each expert taught Disciple-COA independently, while being supervised by a knowledge engineer whose role was to help the expert if he would reach any impasse while operating Disciple.

During the morning of the 4th day each expert succeeded to teach a personal copy of Disciple to correctly critique the COAs with respect to the Principle of Offensive. Then, during the afternoon each of them succeeded to teach Disciple to correctly critique the COAs with respect to the Principle of Security.

The teaching of the Disciple agent proceeded as follows. First Disciple was taught to critique COA411 with respect to the Principle of Offensive. From this it learned a number of task reduction rules like the one from the right hand side of Figure 3. Then COA421 was loaded into the knowledge base and Disciple critiqued it using the learned rules. It correctly identified a strength with respect to the Principle of Offensive. However, according to the provided modeling, there was one additional strength and one weakness in COA421. Therefore the expert taught Disciple to identify them as well, and Disciple learned additional rules. Then COA51 was loaded into the knowledge base and Disciple critiqued it using the rules learned from COA411 and COA422. In this case Disciple generated two wrong critiques. The subject matter expert identified the wrong task reductions and helped Disciple to understand why they were wrong. Then Disciple refined its task reduction rules. After that COA411 was again loaded into the system and critiqued with the updated rules. All the generated solutions were correct. The same was done for COA421, and again all the generated solutions were correct. The same procedure was repeated for the Principle of Security.

Figure 4 shows the evolution of the knowledge base during the teaching process for one of the experts, being representative for all the four experts. The horizontal line shows the successive COAs used to teach Disciple, and therefore corresponds to the time. As one can see from Figure 4, Disciple initially learned more rules (and the corresponding tasks), and then the emphasis shifted to rule refinement. Therefore, the increase in the knowledge base size is greater toward the beginning of the training process for each principle. The teaching for the Principle of Offensive took 101 minutes for this expert. During this time Disciple learned 14 tasks and 14 rules (147 simple axioms equivalent). The teaching for security consisted of 72 minutes of expert-Disciple interactions. During this time Disciple learned 14 tasks and 12 rules (136 simple axioms equivalent). There was very limited assistance from the knowledge engineer and consisted primarily of help with the operation of Disciple. The knowledge acquisition rate obtained was very high (~ 9 tasks and 8 rules / hour expert, or 98 simple axioms equivalent/hour). At the end of this training process, Disciple-COA was able to correctly identify 17 strengths and weakness of the 3 COAs with respect to the principles of offensive and security.

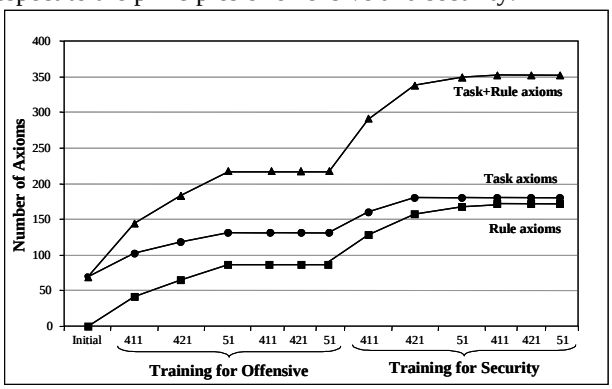

Figure 4: Evolution of the knowledge base.

After the experiment, each expert was asked to fill in a detailed questionnaire designed to collect subjective data for usability evaluation. The questionnaire was organized into three major sections: a) 6 overall questions, b) 40 detail questions, and c) comments and recommendations. The questions addressed three main dimensions of evaluation: effect on task performance, system usability, and system fit, being based on the Multi-Attribute Utility Assessment approach (Adelman and Riedel, 1997). Each dimension considered various criteria (e.g. system fit with the user or system fit with the organization) and even sub-criteria. All answers took into account that Disciple-COA was a research prototype and not a commercial product. They were rated based on a scale of agreement with the question from 1 to 5, with 1 denoting no agreement at all and 5 denoting full agreement. For illustration, Figure 5 shows three of the most relevant questions and the answers provided by the four experts. The results of processing these answers are summarized in Figure 6, where the numeric answers were translated into corresponding percentages (5 to 100% and 1 to 0%). Notice the very high score of 80.8% for fitness to the user and the organization. This shows that the problems addressed in this experiment and their solutions were of realistic complexity, and not toy ones.

Given the fact that each of the four experts succeeded to rapidly (within around 3 hours) develop a knowledge base

Questions	Answers
Do you think that Disciple is a useful tool for Knowledge Acquisition?	• Rating 5. Absolutely! The potential use of this tool by domain experts is only limited by their imagination - not their AI programming skills. • 5 • 4 • Yes, it allowed me to be consistent with logical thought.
Do you think that Disciple is a useful tool for Problem Solving?	• Rating 5. Yes. • 5 (absolutely) • 4 • Yes. As it develops and becomes tailored to the user, it will simplify the tedious tasks.
Were the procedures/ processes used in Disciple compatible with Army doctrine and/or decision making processes?	• Rating 5. As a minimum yes, as a maximum—better! • This again was done very well. • 4 • 4

Figure 5: Sample questions answered by the experts.

for correctly critiquing the three courses of action with respect to the principles of offensive and surprise, we can conclude that the null hypothesis is true.

This experiment demonstrated that the Disciple approach is a very promising solution to the knowledge acquisition bottleneck. It also created a baseline for evaluating future versions of this approach.

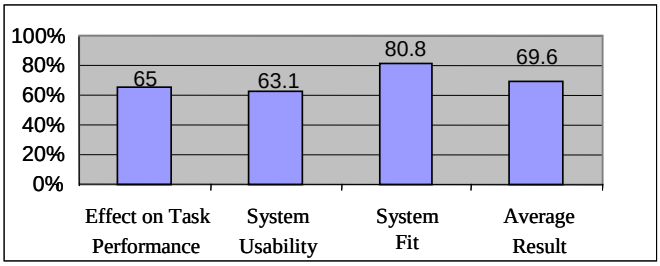

Figure 6: Overall System Utility

7 Conclusions

We have presented a general framework for the empirical evaluation of mixed-initiative knowledge base development methods and tools.

An important aspect of this framework is that the evaluation is an integral part of system design and development. This makes the framework very appropriate for the typical Ph.D. research projects where successive versions of a method have to be evaluated and compared to one another, and the reasons for the obtained behavior clearly elucidated.

Another important aspect is the extensibility of this framework with new evaluation methods, experimentation set-ups and reusable evaluation utilities.

However, an yet unsolved research issue, is how to evaluate the evaluation framework itself, beyond its repeated application.

Acknowledgements

This research was done in the GMU Learning Agents Laboratory (LALAB). Research of the LALAB is sponsored by the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, Air Force Material Command, USAF, under agreement number F30602-00-2-0546, and by the Air Force Office of Scientific Research (AFOSR) under grant no. F49620-00-1-0072. Dorin Marcu, Michael Bowman, Bogdan Stanescu, Catalin Balan, Elena Popovici, Cristina Cascaval, and other members of the LALAB contributed to successive versions of Disciple. The anonymous reviewers of this paper provided insightful comments that helped us to improve it.

References

[Adelman and Riedel, 1997] Leonard Adelman and Sharon L. Riedel. *Handbook for Evaluating Knowledge-based Systems*, Kluwer Academic Publishers, Boston, MA., 1997.

[Boicu *et al*. 2000] Mihai Boicu, Gheorghe Tecuci, Michael Bowman, Ping Shyr, Florin Ciucu, and Cristian Levcovici. Disciple-COA: From Agent Programming to Agent Teaching. In *Proceedings of the Seventeenth International Conference on Machine Learnin*g, Stanford, CA: Morgan Kaufmann, 2000.

[Cohen, 1995] Paul Cohen. *Empirical Methods for Artificial Intelligence*, The MIT Press, Boston, MA., 1995.

[Cohen *et al.*, 1998] Paul Cohen, Robert Schrag, Eric Jones, Adam Pease, Albert Lin, Barbara Starr, David Gunning and Murray Burke. The DARPA High-Performance Know-ledge Bases Project, *AI Magazine*, 19(4), 25-49, 1998.

[Kim and Gil, 1999]. Jihie Kim and Yolanda Gil. Deriving Expectations to Guide Knowledge Base Creation. In *Proc. of the Sixteenth National Conference on Artificial Intelligence*,235-241, Menlo Park, CA: AAAI Press, 1999.

[Lenat, 1995] Douglas B. Lenat. CYC: A Large-scale investment in knowledge infrastructure. *Communications of the ACM* 38(11):33-38, 1995.

[MacGregor, 1999] Robert MacGregor. *Retrospective on LOOM.* Available online: http://www.isi.edu/isd/ LOOM/ papers/macgregor/Loom_Retrospective.html, 1999.

[Menzies, 1998] Tim Menzies. Evaluation issues for problem solving methods. In *Proc. 11th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW ‘‘98)*, Banff, Canada, April 18-23, 1998.

[Tecuci, 1998] Gheorghe Tecuci. *Building Intelligent Agents: An Apprenticeship Multistrategy Learning Theory, Methodology, Tool and Case Studies.* London, England: Academic Press, 1998.

[Tecuci, 2000] Gheorghe Tecuci, Michael Bowman, Dorin Marcu, Ping Shyr, and Cristina Cascaval. 2000. An Experiment in Agent Teaching by Subject Matter Experts. *International Journal of Human-Computer Studies* 53: 583-610.

[Tecuci, 2001] Gheorghe Tecuci, Mihai Boicu, Michael Bowman, and Dorin Marcu, 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, 2001.

[Shadbolt *et al*. 1999] Nigel Shadbolt, Kieron O'Hara, and Louise Crow. The Experimental Evaluation of Knowledge Acquisition Techniques and Methods: History, Problems and New Directions. *International Journal of Human-Computer Studies* , vol. 51, 729-755, 1999.

[Webb *et al*. 1999] Geoffrey I. Webb, Jason Wells, and Zijian Zheng. An Experimental Evaluation of Integrating Machine Learning with Knowledge Acquisition. *Machine Learning*, vol. 35, 5-23, 1999.