In Proceedings of the 2001 International Conference on Artificial Intelligence, IC-AI'2001, pp. 122-128. June 25-28, 2001, Monte Carlo Resort, Las Vegas, Nevada, USA. IEEE Computer Society, Los Alamitos, California, 2001.

Mixed-Initiative Agent Teaching and Learning

Mihai Boicu, Gheorghe Tecuci and Bogdan Stanescu
Learning Agents Laboratory, Department of Computer Science, MSN 4A5,
George Mason University, 4400 University Dr., Fairfax, VA 22030-4444, USA
{mboicu, tecuci, bstanesc}@gmu.edu

Abstract. This paper presents a mixed-initiative teaching and learning method that allows a subject matter expert, who does not have knowledge engineering experience, to transfer his problem solving expertise to a learning agent. The expert is teaching the agent to perform domain specific tasks in a way that resembles how the expert would teach an apprentice while solving problems in cooperation. The method integrates synergistically the complementary knowledge and reasoning styles of the expert and the agent, where the expert helps the agent to learn and the agent helps the expert to teach it. This approach is illustrated with examples of teaching the Disciple-COA agent to critique courses of actions, a challenge problem defined in the DARPA's High Performance Knowledge Bases program.

1 Introduction

While significant progress has been made in the area of knowledge engineering, building knowledge bases and agents that incorporate the knowledge of a subject matter expert (SME) is still a very long and error-prone process [2]. One of the main difficulties is that the knowledge engineer who actually builds the knowledge-based agent has to become quite knowledgeable in the expert's subject matter, and the expert has to learn enough about the knowledge engineering process to properly explain his or her knowledge to the knowledge engineer. To achieve these goals requires time and effort. Therefore, there is a contradiction in the knowledge-based agent development process, where the presence of the knowledge engineer is both part of the solution and part of the problem. Moreover, as computers become more powerful and user-friendly, and the knowledge-based agents demonstrate their utility in all the segments of the information society, there appears a need for SMEs to build knowledge bases and agents by themselves. Responding to this need, DARPA has initiated the Rapid Knowledge Formation program [3], to gain a scientific understanding of how ordinary people can work with formal representation of knowledge and to develop the Artificial Intelligence technology that will allow SMEs that do not have prior knowledge engineering experience to build knowledge bases directly, with no or very limited assistance from knowledge engineers.

Our approach to this problem is to develop a powerful learning agent that can collaborate with a subject matter expert to develop its knowledge base so that it represents the problem solving knowledge of the human expert. The goal is to divide the responsibility between the expert and the agent for those elements of knowledge engineering for which they have the most knowledge and aptitude, such that together they form a complete team for knowledge base development. This requires mixed-initiative reasoning, where the human and the automated agent share representations, communicate naturally, properly divide their tasks and responsibilities, coordinate their actions, take the initiative and release the control.

When building the knowledge base of an agent, a knowledge engineer acts directly on the "brain" of the agent because he understands the "neurons and the connections." A subject matter expert cannot do this. What he or she could do is to transfer his or her knowledge in a way that is similar to how he or she would teach a human student or apprentice. It would then be the agent's responsibility to learn from the input provided by the expert to build its knowledge structures. Therefore the process of building the knowledge base of the agent is modeled by analogy with the human educational process, where the expert is the teacher and the agent is the learner. However, both teaching of the agent and learning from the expert are very complex tasks. To deal with this complexity we have developed mixed-initiative methods where the expert helps the agent to learn and the agent helps the expert to teach it.

Using this approach, we have developed a family of learning agents, called Disciple [9], that are increasingly more capable to acquire problem solving knowledge from subject matter experts. A Disciple learning agent consists of an integrated set of knowledge acquisition, learning and problem solving modules for a generic knowledge base structured into two main components:

1) an object ontology that defines the types of objects from a specific application domain, together with their properties and relationships, all represented as frames, according to the knowledge model of the Open Knowledge Base Connectivity protocol [4];
2) a set of problem solving rules expressed with these concepts. The problem solving approach of a Disciple agent is task reduction, where a task to be accomplished by the agent is successively reduced to simpler tasks until the initial task is reduced to a set of elementary tasks. Therefore, the rules from the knowledge base are task reduction rules.

The process of building the knowledge base of a Disciple agent includes three main phases: domain modeling, ontology development, and rule development. In the domain modeling phase, the subject matter expert identifies

a set of specific tasks that constitute a representative set for the tasks that the final agent should be able to perform. Then, for each of these tasks, he or she represents the corresponding problem solving process as a sequence of task reductions (and, possibly, task composition) steps. This process also produces an informal specification of the objects needed to be represented into the agent's ontology. This specification guides the ontology development phase, where some of the object concepts are imported from existing repositories of knowledge, and others are defined by the expert (possibly assisted by a knowledge engineer). This second phase results in an initial knowledge base that contains an object ontology but no rules. The rules are developed during the third phase when the subject matter expert teaches the agent how to perform its tasks in a way that resembles how the expert would teach a human apprentice when solving problems in cooperation. As a result of this process, the agent will learn problem solving rules from the expert, and will also extend and update its ontology.

Mixed-initiative reasoning occurs during each knowledge base development activity: domain modeling, ontology import, ontology development, problem solving, rule learning and rule refinement. In this paper we discuss this approach in the context of rule learning, illustrating it with an example of developing a course of action critiquer, a challenge problem introduced in the DARPA's and AFOSR's High Performance Knowledge Bases (HPKB) program.

2 The Disciple Course of Action Critiquer

A military course of action (COA) is a preliminary outline of a plan for how a military unit might attempt to accomplish a mission. It consists of a COA sketch and a COA statement. The COA sketch is a graphical representation of the terrain, locations, compositions and missions of the friendly and enemy units. The COA statement explains in a restricted subset of English what the units will do to accomplish the assigned mission. The COA challenge problem consists of rapidly developing a knowledge-based critiquer that receives as input the description of a military course of action for ground force operations and assesses various aspects of the COA. The role of this critiquing agent is to act as an assistant to the military commander, helping the commander to choose between several COAs under consideration for a certain mission. The agent could also help students to learn to develop courses of action.

To solve this challenge problem we have developed the Disciple-COA agent and have taught it to identify the strengths and the weaknesses of a course of action with respect to the principles of war and the tenets of army operations. There are nine principles of war: objective, offensive, mass, economy of force, maneuver, unity of command, security, surprise, and simplicity. They provide general guidance for the conduct of war at the strategic, operational and tactical levels. The tenets of army operations describe the characteristics of successful operations. They are: initiative, agility, depth, synchronization and versatility [5].

The expert teaches the Disciple-COA agent how to generate such solutions while jointly performing critiquing tasks, as described in the following. First, the expert formulates the critiquing task to be performed, for instance, "Assess COA411 with respect to the Principle of Mass." To perform this assessment one needs a certain amount of information that is obtained by asking a series of questions. The answer to each question allows one to reduce the current assessment task to a more specific one, as illustrated in Figure 1.

Assess COA411 with respect to the Principle of Mass
Does COA411 identify a decisive point?
Yes, it identifies the decisive point RED-MECH-COMPANY4.

Then I have to
Assess mass for COA411 with RED-MECH-COMPANY4 as the decisive point.
Does the main effort act on RED-MECH-COMPANY4 with an adequate force ratio?
Yes, it acts with a force ratio of 10.6.

Then I have to
Assess mass for COA411 when the main effort acts on RED-MECH-COMPANY4 with the adequate force ratio of 10.6.
Does the main effort get help acting on RED-MECH-COMPANY4?
Yes, it gets help from the supporting action SUPPRESS1, which also acts on RED-MECH-COMPANY4.

Then I can conclude that
There is a major strength in COA411 with respect to mass because the main effort acts on the decisive point of the COA (RED-MECH-COMPANY4) with a force ratio of 10.6, which exceeds the recommended force ratio. Additionally, the main effort is assisted by the supporting action SUPPRESS1 which also acts on the decisive point. This is good evidence of the allocation of significantly more than minimum combat power required at the decisive point and is indicative of the proper application of the principle of mass.

Figure 1: COA critiquing through task reduction.

In the beginning such task reduction steps are provided by the expert. However, from each of them Disciple learns a general task reduction rule, with the help of the expert, as will be described in the following section. As the agent learns more rules, it takes the initiative in this cooperative problem solving process, proposing

reductions by itself. The expert has to analyze each reduction proposed by the agent, deciding whether to accept it or to reject it. In each case the agent will learn from the expert, either by generalizing or by specializing the rule that generated the reduction. If the agent was not able to propose any task reduction or the proposed reduction was rejected, the expert has to provide a solution. In this case the agent learns a new rule from the example provided by the expert.

3. Mixed Initiative Rule Learning

In the following we will illustrate the mixed-initiative rule learning process. Let us consider the task reduction process illustrated in Figure 1. First, the expert formulates the critiquing task to be performed "Assess COA411 with respect to the Principle of Mass", and the agent reduces it to "Assess mass for COA411 with RED-MECH-COMPANY4 as the decisive point." Let us now assume that the agent does not know how to reduce this task. Therefore, the next reduction has to be indicated by the expert. The left hand side of Figure 2 represents the reasoning process of the expert. It is the same with the second reduction step in Figure 1. The right hand side of Figure 2 represents the same reasoning, this time in the internal representation language of the agent. Notice that the internal representation of a task is only a surface reformulation of its external representation. The internal representation consists of a task name and a set of features. The task name is an abstract phrase characterizing the task as a whole. That is, it should not contain any specific object. Each feature is another phrase that further characterizes the task, by referring to specific objects. Task formalization is a mixed-initiative reasoning process that is done during domain modeling. We will not address it in this paper. Instead, we will assume that this problem has already been solved. A remaining challenge is then to help the agent understand why the expert has performed the task reduction in Figure 2. This process is described bellow.

3.1 Explanation Generation

The Question and its Answer provide an explanation of why the top task in Figure 2 is reduced to the bottom task. While this explanation is very natural to a human expert, a learning agent cannot understand it. The explanation that would be understood by the agent is represented in the right part of Figure 2, and consists of various relations between certain elements from the agent's ontology. The first two explanation pieces are:

BLUE-TASK-FORCE1 ASSIGNMENT MAIN-EFFORT1
BLUE-TASK-FORCE1 TASK PENETRATE1 OBJECT-ACTED-ON RED-MECH-COMPANY4

They state, in Disciple's language, that the main effort acts on RED-MECH-COMPANY4: BLUE-TASK-FORCE1 is the main effort and its task is PENETRATE1, which acts on RED-MECH-COMPANY4. Similarly, the next two explanation pieces state that the main effort acts on RED-MECH-COMPANY4 with the adequate force ratio of 10.6: the actual force ratio for PENETRATE1 is 10.6, and this is greater than 3.0, the recommended force ratio.

An expert can understand these formal expressions because they actually correspond to his own explanations. However, he cannot be expected to be able to define them because he is not a knowledge engineer. For one thing, he would need to use the formal language of the agent. But this would not be enough. He would also need to know the names of the potentially many thousands of concepts and features from the agent's ontology.

While defining the formal explanation of this task reduction step is beyond the individual capabilities of the expert and the agent, it is not beyond their joint capabilities. Finding these explanation pieces is a mixed-initiative process of searching the agent's ontology, an explanation piece being a path of objects and relations in this ontology. In essence, the agent will use analogical reasoning and help from the expert to identify and propose a set of plausible explanation pieces from which the expert will have to select the correct ones.

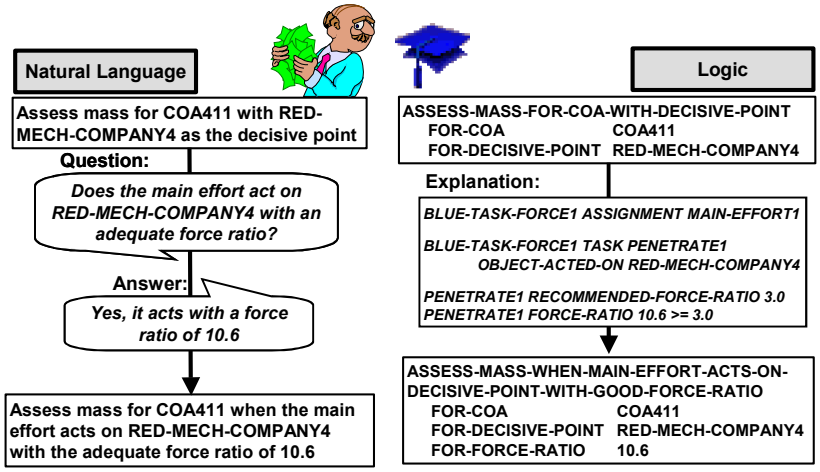

Figure 2: Language to logic translation.

One analogical reasoning heuristic is the following one:

1. Look for a rule R_k that reduces the current task (i.e. "ASSESS-MASS-FOR-COA-WITH-DECISIVE-POINT"), even though this rule is not applicable in the current situation.
2. Extract the explanations E_g from the rule R_k.
3. Look for explanations of the current task reduction (i.e. the task reduction in Figure 1) that are similar with E_g, and propose them to the expert.

This heuristic is based on the observation that the explanations of the alternative reductions of a task tend to have similar structures. The same factors are considered, but the relationships between them are different. Let us consider, for instance, that the agent has previously learned the task reduction rule in Figure 3. As one can see, the explanation pieces from the rule in Figure 3 (that correspond to a situation where the main effort acts on the decisive point, but not with an adequate force ratio), have the same structure with the explanation pieces from the right of Figure 2, except that the actual force ratio is less (not greater than) the recommended force ratio. The agent will use the structure of the explanation pieces from the rule in Figure 3 in conjunction with the current problem solving situation, and will propose to the expert the explanation pieces from Figure 2.

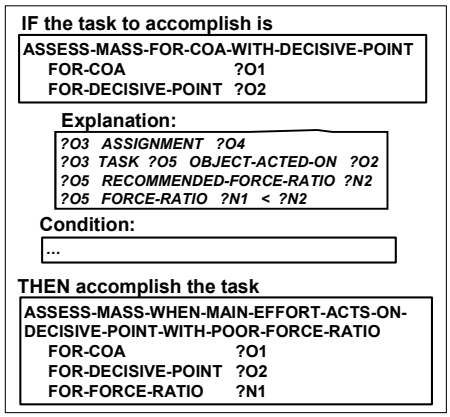

Figure 3: A previously learned task reduction rule.

Let us consider again the situation from Figure 2. To provide the solution to the agent, the expert uses the Example Editor that is already initialized with the task to be reduced. Then he has to specify the question, the answer, and the subtask(s). Notice, however, that the above heuristic relies only on the task to be reduced. Therefore, the agent can start immediately to search for plausible explanations, in parallel with the completion of the example by the expert. Nevertheless, as the completion of the example advances, the agent could extract additional hints from the expert's actions. For instance, once the expert has specified the question and the answer, the objects and the relations referred there (such as main effort, RED-MECH-COMPANY4, force ratio, 10.6) could be used as searching hints. Indeed, they tell the agent that the potentially useful explanations are those that contain these elements. After the expert has completed the definition of the example, the agent will analyze the example to determine whether this is a positive exception of an existing rule, or whether it is a genuinely new example that justifies the learning of a new rule. In the later case the agent will initiate other analogical reasoning heuristics for explanation generation that take into account the entire form of the example (not just its top task). For instance, another explanation generation heuristic is to consider the explanations from the rules that reduce a task that is similar (but not identical) with the top task in Figure 2, and the resulting task is also similar with the task from the bottom of Figure 2. A weaker (but nevertheless useful) heuristic is to also consider the rules where only the initial task is similar with the top task in Figure 2. The expert could also give explicit hints to the agent. A hint is any type of guidance of how to search for explanations, for instance to look at the possible links between two given objects. The Question and its Answer also provide hints to the agent.

3.2 Rule generation

From the task reduction example and its explanations shown in Figure 2, the agent automatically generates the task reduction rule shown in Figure 4. The internal form of the rule is an IF-THEN structure that specifies two conditions under which the task from the IF part can be reduced to the task from the THEN part. These two conditions are the bounds of the exact (but not yet known) applicability condition of the rule. The plausible upper bound condition is, as an approximation, more general than the exact condition, and the plausible lower bound condition is less general than the exact condition. In essence, the rule in Figure 4 is obtained from the

example in Figure 2 by turning its constants (specific objects and numbers) into variables, and then restricting the possible values of these variables. The plausible lower bound condition restricts the variables to only take the values from the example in Figure 2. For instance, ?O5 is restricted to only take the value PENETRATE1, because this is the corresponding value in the example (see the explanations from the right hand side of Figure 2 and from the rule in Figure 4). The plausible upper bound condition allows a variable to take any value that does not contradict the agent's definitions of the object features and of the task features. For instance, PENETRATE1 appears as the value of the feature TASK and has the features OBJECT-ACTED-ON, FORCE-RATIO and RECOMMENDED-FORCE-RATIO. Each of these features is characterized by a domain (that indicates the set of objects that may have that feature) and a range (that indicates the possible values of this feature). Therefore ?O5 (the generalization of PENETRATE1) is restricted by the plausible upper bound condition to take values that are within the corresponding domains and ranges of these features. That is, ?O5 should be an ACTION. We should also notice that the domains and the ranges of the features are also learned by the agent. Therefore, at the time a rule is learned, the agent uses the current values of these entities, this being one reason why the rule's bounds are called plausible. During further problem solving and learning the agent will generalize or specialize the two conditions of the rule, depending on whether the result produced by the rule was correct or not, and the two bounds will converge toward one another. Let us also notice that, in addition to the two conditions, the learned rule also includes generalizations of the explanations, and of the question and its answer, which basically represent the same information at higher levels of abstraction.

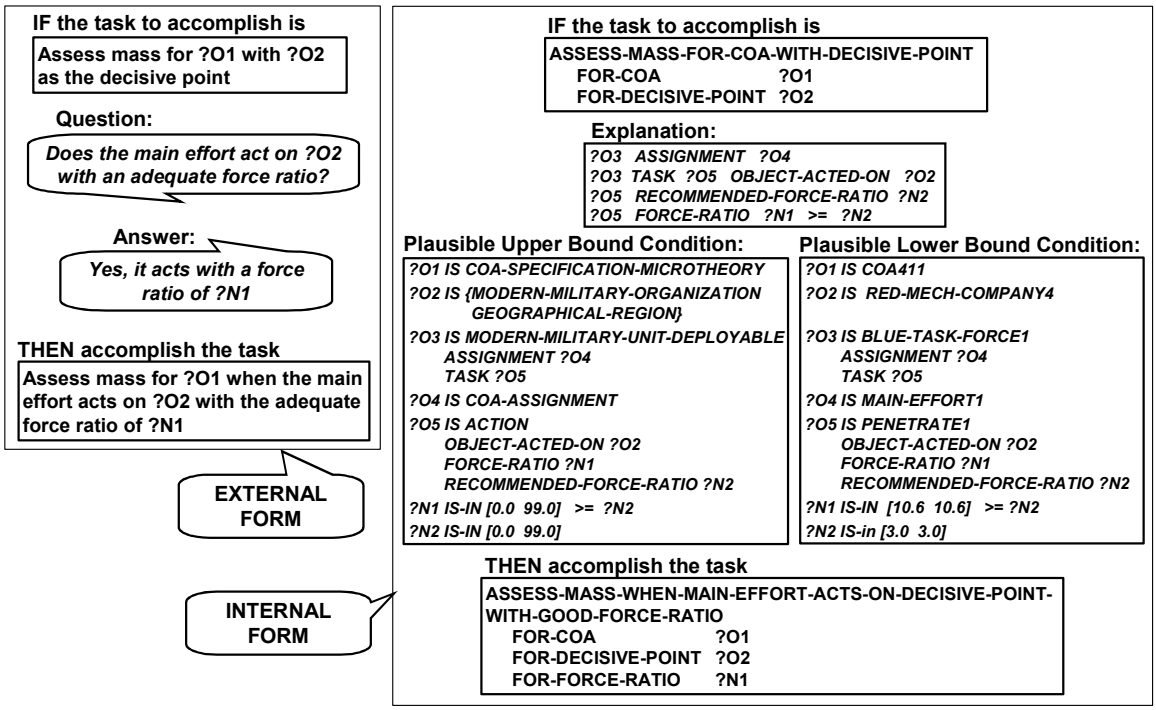

Figure 4: Learned plausible version space rule.

4 Mixed-Initiative Description Language

We have defined a mixed-initiative description (MID) language to formally represent the tasks to be performed by the expert and the agent through mixed-initiative reasoning. This is a rule-based language where each rule describes the execution of a higher level task in terms of simpler subtasks, their preconditions, the messages that could be exchanged between them, different possible flows of execution of these tasks, their possible results, and who has to execute them (the expert or the agent, or any of them). For example, Figure 5 is a graphical representation of the MID rule corresponding to the rule learning task described in the previous section. According to this MID rule, the LEARN_RULE task is decomposed into four subtasks: DEFINE_EXAMPLE, ANALIZE_EXAMPLE, EXPLAIN_EXAMPLE and CREATE_RULE. The first three subtasks are non elementary and are decomposed by other MID rules into simpler tasks, some to be executed by the agent and some by the expert. Each MID rule also represents the control and the communication flow between the component tasks. The solid arrows represent the flow of control and the messages that initiate or end a task. The dashed arrows represent the messages that are sent or received by a task during its execution. The MID rule also represents the interaction of the LEARN_RULE task with its parent task, from which it may receive some messages and to which it will send "END-WITH" messages.

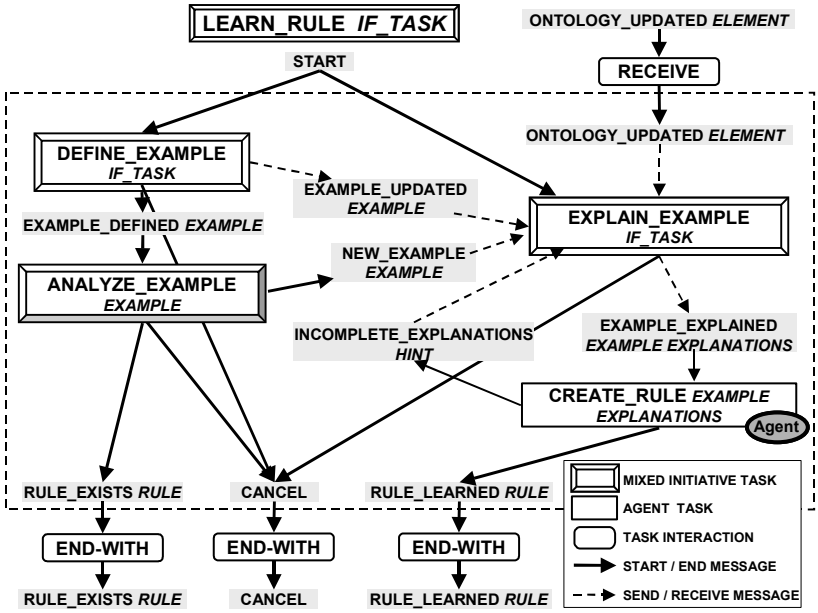

Figure 5: Mixed-initiative description language.

The MID rule from Figure 5 is fired with an *IF_TASK* like the one from the top of Figure 2 (i.e. ASSESS-MASS-FOR-COA-WITH-DECISIVE-POINT). This is a domain task that the expert will reduce. The expert will also help the agent to understand the reduction and to learn a general task reduction rule, as explained above. A Start message is immediately generated, triggering the parallel execution of the DEFINE_EXAMPLE task and of the EXPLAIN_EXAMPLE task.

The EXPLAIN_EXAMPLE task is a mixed-initiative task that has been intuitively presented above with the help of Figure 2. In essence, the expert and the agent collaborate in finding the formal explanations from the right of Figure 2. Because this task is computationally expansive, it is started immediately, even before the example reduction from Figure 2 has been completely defined. This is possible because some of the explanation generation heuristic methods (such as the one illustrated in Section 3.1) are only based on the task to be reduced. However, the EXPLAIN_EXAMPLE task receives continuously updated forms of the partial example from the DEFINE_EXAMPLE task, which triggers new explanation generation heuristics (such as those based on the hints from the question and the answer in the left hand side of Figure 2). Therefore by the time the example is completely defined and the expert directs his attention to its explanation, the agent has already computed plausible explanations and may propose them to the expert, ordered by their plausibility. The expert may select some of the explanations generated by the agent and may provide hints that will guide the agent in looking for additional explanations. Hints for the explanations generation may come from other tasks as well. For instance, during example definition the expert may need to extend the ontology of the agent with new terms needed in the example, and with their description. As indicated in the top right part of Figure 5, the EXPLAIN_EXAMPLE task will receive the new or updated elements as hints from the Ontology learning module.

The DEFINE_EXAMPLE is itself a complex mixed-initiative task the goal of which is to complete the reduction step, as illustrated in Figure 2. The end of this task initiates the ANALIZE_EXAMPLE task, another mixed-initiative task the role of which is to compare the defined example with the existing rules. If the new example does not structurally match any of the existing rules, then a new rule is created by the CREATE_RULE task, based on the example and its explanations. The CREATE_RULE task also analyzes the generated rule and may identify the need for additional explanations, for instance, when any of the rule's variables is under-constrained.

As can be seen, this rule description language allows breaking down complex tasks that need to be jointly performed by the expert and by the agent, into simpler subtasks, and to divide the responsibility between the expert and the agent for those of these tasks for which they have the most aptitude. For instance, DEFINE_EXAMPLE falls mostly under the responsibility of the expert, but CREATE_RULE is entirely under the responsibility of the agent. The rule expresses also the potential shift in initiative and control. For instance, the agent may take the initiative of looking for explanations even before the example has been completely defined. This is important because, as mentioned above, explanation generation is a computationally intensive process and the example definition does not require significant resources.

The interpreter of the MID language provides a mixed-initiative control of task execution. An important component of this interpreter is a task agenda that displays all the tasks relevant to the user at different levels of abstraction, providing an increasing efficiency of interaction as the human expert becomes more experienced in collaborating with the agent. The task agenda also highlights the tasks that are currently being executed and

those that can be scheduled for execution at that time, to keep the expert informed with both the current and the anticipated status of the execution. This task execution framework allows the agent to reason about the current tasks performed by the expert and to exhibit proactive behavior by executing support tasks in advance, in order to facilitate the initiation or completion of an important expert task that will follow.

8 Evaluation Results and Final Remarks

Simpler versions of the mixed initiative teaching and learning method presented in this paper were applied for the development of several Disciple agents, most recently as part of the HPKB program. In this program, the Disciple-Workaround agent, the Disciple-COA agent, and other competing knowledge base development approaches (such as CYC [Lenat, 1995], EXPECT [Kim and Gil, 1999] and Loom/PowerLoom [MacGregor, 1999]) have been evaluated by Alphatech in several intensive studies requiring the rapid development and maintenance of knowledge bases for solving the workaround challenge problem and the course of action challenge problem. In these experiments all the approaches demonstrated very good results and relative technology strengths. However, the Disciple approach has shown higher rates of knowledge acquisition and better problem solving performance, while the generated solutions and justifications where judged as being very intelligible [10], [11]. We give credit for these successes to the extensive use of mixed-initiative teaching and learning methods.

To test the claim that domain experts can teach a Disciple agent, we conducted a knowledge acquisition experiment at the US Army Battle Command Battle Lab, in Fort Leavenworth, Kansas. In this experiment, four military experts that did not have any prior knowledge engineering experience received around 16 hours of training in Artificial Intelligence and the use of Disciple-COA. They then succeeded in training Disciple to critique COAs with respect to the Principle of Offensive and the Principle of Security, in about three hours, following a modeling of the critiquing process that was discussed with them at the beginning of the experiment. At the end of the experiment they completed a detailed questionnaire that revealed high scores for the perceived usefulness and usability of Disciple [11].

Currently we are continuing to develop the Disciple approach and we are applying it to the determination and analysis of the strategic center of gravity, in cooperation with the US Army War College, as part of the DARPA's Rapid Knowledge Formation program. These results support the long-term vision of the Learning Agents Laboratory (http://lalab.gmu.edu) of developing the Disciple approach to a point where typical computer users can build and maintain knowledge bases and agents as easily as they currently use personal computers for text processing.

Acknowledgments: This research was supported by the Defense Advanced Research Projects Agency, 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 under grant number F49620-00-1-0072. Dorin Marcu, Michael Bowman, Catalin Balan, Elena Popovici, Cristina Cascaval, and other members of the LALAB contributed to successive versions of Disciple.

References

1. Boicu M., Tecuci G., Marcu D., Bowman M., Shyr P., Ciucu F., Levcovici C.: Disciple-COA: From Agent Programming to Agent Teaching. In: Proceedings of the Seventeenth International Conference on Machine Learning. Morgan Kaufman, Stanford (2000) 73-80
2. Buchanan, B. G., Wilkins, D. C. (eds.): Readings in Knowledge Acquisition and Learning: Automating the Construction and Improvement of Expert Systems. Morgan Kaufmann, San Mateo (1993)
3. Burke, M.: Rapid Knowledge Formation - Program Description, http://web-ext2.darpa.mil/ iso/rkf/RKF_PIP.htm. (1999)
4. Chaudhri, V.K., Farquhar, A., Fikes, R., Park, P.D., Rice, J.P.: OKBC: A Programmatic Foundation for Knowledge Base Interoperability. In: AAAI-98 Proceedings. AAAI Press, Menlo Park (1998) 600–607
5. FM-105: US Army Field Manual 100-5, Operations, Headquarters. Department of the Army. (1993)
6. Kim, J., and Gil, Y.: 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)
7. Lenat, D.B.: CYC: A Large-scale investment in knowledge infrastructure. Communications of the ACM 38(11), (1995) 33-38
8. MacGregor, R.: Retrospective on LOOM. Available online: http://www.isi.edu/isd/LOOM/papers/macgregor/ Loom_Retrospective.html, (1999).
9. Tecuci G.: Building Intelligent Agents: An Apprenticeship Multistrategy Learning Theory, Methodology, Tool and Case Studies. Academic Press, London (1998)
10. Tecuci G., Boicu M., Wright K., Lee S.W., Marcu D., Bowman M.: An Integrated Shell and Methodology for Rapid Development of Knowledge-Based Agents. In: Proceedings of the Sixteenth National Conference on Artificial Intelligence. AAAI Press, Menlo Park (1999) 250-257
11. Tecuci G., Boicu M., Marcu D., Bowman M., Ciucu F., Levcovici C.: Rapid Development of a High Performance Knowledge Base for Course of Action Critiquing. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence and the Twelfth Conference on Innovative Application of Artificial Intelligence. AAAI Press, Menlo Park (2000) 1046-1053