In Proceedings of the 2001 International Conference on Tools With Artificial Intelligence, ICTAI-2001, Dallas, Texas, November 2001.

Automatic Knowledge Acquisition from Subject Matter Experts

Mihai Boicu, Gheorghe Tecuci, Bogdan Stanescu, Dorin Marcu, and Cristina Cascaval
Learning Agents Laboratory, Department of Computer Science, MS 4A5
George Mason University, 4400 University Drive, Fairfax, VA 22030-4444
{mboicu, tecuci, bstanesc, dmarcu, ccascava}@gmu.edu, http://lalab.gmu.edu

Abstract

This paper presents current results in developing a practical approach, methodology and tool, for the development of knowledge bases and agents by subject matter experts, with limited assistance from knowledge engineers. This approach is based on mixed-initiative reasoning that integrates the complementary knowledge and reasoning styles of a subject matter expert and a learning agent, and on a division of responsibilities for those elements of knowledge engineering for which they have the most aptitude. The approach was evaluated at the US Army War College, demonstrating very good results and a high potential for overcoming the knowledge acquisition bottleneck.

1. Introduction

This paper addresses the knowledge acquisition bottleneck in the development of knowledge bases and agents, bottleneck that we consider to be one of the main barriers in the generalized application of Artificial Intelligence. Traditionally, a knowledge-based system is built by a knowledge engineer (KE) who has to acquire the knowledge from a subject matter expert (SME) and to encode it into the knowledge base (KB). This is a very difficult process because the experts express their knowledge informally, using natural language, visual representations and common sense, often omitting many essential details that are considered obvious. In order to properly understand an expert's problem solving knowledge and to represent it in a formal, precise, and complete knowledge base, the knowledge engineer needs to become himself a kind of subject matter expert. Therefore this process is very difficult, error-prone, and time-consuming [2].

Our research goal is to develop a theory, methodology and tool that will allow SMEs that do not have prior knowledge engineering experience to build knowledge-based systems by themselves, with no or very limited assistance from knowledge engineers. Our approach to this problem consists of developing a very capable learning agent shell that can perform many of the functions of a KE. The SME and the agent engage into a mixed-initiative process of developing the agent's KB to incorporate the expertise of the SME. The concept of learning agent shell is an extension of the concept of expert system shell [5]. As an expert system shell, it includes a general inference engine that can be reused for multiple applications. In addition, it includes a general learning engine for building a knowledge base consisting of an object ontology that describes the entities from an application domain, and a set of problem solving rules expressed with these objects. The process of developing a knowledge-based system for a specific application relies on importing ontological knowledge from existing knowledge repositories, and on teaching the learning agent how to perform various tasks, in a way that resembles how an expert would teach a human apprentice when solving problems in cooperation.

Over the years we have developed a series of learning agent shells from the Disciple family [8], most recently as part of the "High Performance Knowledge Bases" and "Rapid Knowledge Formation" programs supported by DARPA and AFOSR [3]. These programs emphasize the use of the challenge problems to focus the research and development efforts and measure the effectiveness of alternative technical approaches to the development of knowledge-based systems. Each of the following challenge problems required the rapid development and maintenance of a KB for a different type of application, and led to the development of an extended and improved Disciple system:

1) The Workaround challenge problem - planning how a convoy of military vehicles can circumvent or overcome obstacles in their path, such as damaged bridges. To solve this challenge problem we have developed the Disciple-Workaround learning agent, demonstrating that a knowledge engineer can rapidly teach Disciple, using military engineering manuals and sample solutions provided by a subject matter expert [6, 1, 9].

2) The Course of Action (CoA) challenge problem - critiquing military courses of actions with respect to the principles of war and the tenets of army operations. To

solve this challenge problem we have developed the Disciple-CoA learning agent with which we achieved two new significant milestones. For the first time we have developed the knowledge base using an object ontology created by another group (Teknowledge and Cycorp), demonstrating both the feasibility of knowledge reuse with the Disciple approach, and the generality of the Disciple rule learning methods. Also, we have conducted a one-week knowledge acquisition experiment at the US Army Battle Command Battle Lab, in Fort Leavenworth, Kansas, where four military experts who did not have any prior knowledge engineering experience succeeded to train Disciple, by following a model of the CoA critiquing process that was provided to them [10].

3) The Center of Gravity (CoG) challenge problem - identifying strategic center of gravity candidates in military conflicts. This has resulted in the development of the Disciple-RKF/CoG learning agent and in the achievement of another significant milestone: to our knowledge, it is for the first time that subject matter experts succeeded to develop end-to-end knowledge-based agents, with very limited assistance from knowledge engineers.

This paper presents the Disciple-RKF/CoG tool, the latest version of the Disciple learning agent shell. We first introduce the Center of Gravity challenge problem that has focused the development of Disciple-RKF/CoG. Then we present the architecture of Disciple-RKF/CoG and the main stages of agent development. Each of the Disciple components is then presented in more details, following the stages of agent development. At the end we present recent experimental results and the main directions of our near future research.

2. The Center of Gravity Challenge Problem

The concept of Center of Gravity was introduced by Clausewitz in 1832, in his classical book "On War". The center of gravity of an entity (state, alliance, coalition, or group) is the foundation of capability, the hub of all power and movement, upon which everything depends, the point against which all the energies should be directed. If a combatant eliminates or influences the enemy's strategic center of gravity, then the enemy will lose control of its power and resources and will eventually fall to defeat. Similarly, if the combatant fails to adequately protect his own strategic center of gravity, he invites disaster [7]. There is a lot of emphasis on the determination and analysis of CoG in practice and in the education of strategic leaders at all the senior military service colleges. Therefore, the George Mason University Learning Agents Laboratory has teamed up with researchers and subject matter experts from the Center for Strategic Leadership of the US Army War College to develop, apply and evaluate the Disciple approach in the context of CoG candidates identification and analysis.

The developed Disciple-RKF/CoG learning agent was used and evaluated in two courses at the US Army War College, during the Winter-2001 and Spring-2001 terms, as described in section 10.

In the rest of this paper we will use examples from the Okinawa scenario, describing the planned US invasion of the island of Okinawa in 1945, to illustrate the Disciple methodology and tools.

3. The Disciple Architecture

The general architecture of the Disciple shell is shown in Figure 1. At a general level, Disciple contains three main components:

1) The intelligent user interface which allows the expert to communicate with Disciple in a manner that is similar to the way he communicates in his environment;

2) The teaching, learning and problem solving component which performs knowledge formation and problem solving, containing tools for rule learning, rule refinement, mixed initiative problem solving and autonomous problem solving;

3) The knowledge base management component that contains tools for managing the knowledge base and for importing knowledge from external knowledge repositories.

The first phase in developing a Disciple agent capable of solving the CoG problem is to build a generic object ontology that defines the type of objects from the application domain (such as opposing force, state, or government), the type of features that these objects may have, and corresponding elicitation scripts that will guide the expert to define instances of these objects and features corresponding to a specific scenario. After this phase is

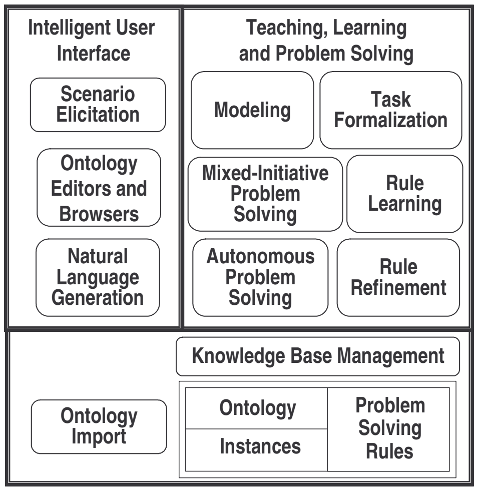

Intelligent User Interface
Scenario Elicitation
Ontology Editors and Browsers
Natural Language Generation
Teaching, Learning and Problem Solving
Modeling
Task Formalization
Mixed-Initiative Problem Solving
Rule Learning
Autonomous Problem Solving
Rule Refinement
Knowledge Base Management
Ontology Import
Ontology
Instances
Problem Solving Rules

Figure 1: The Disciple architecture.

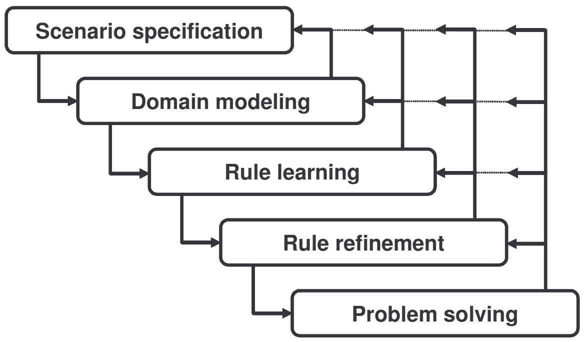


Figure 2. The main phases of agent development with Disciple-CoG.

completed by the knowledge engineer, the rest of the agent development is entirely done by the subject matter expert, with no or very limited assistance from a knowledge engineer. Figure 2 shows the phases of this agent development process. During the Scenario specification phase, the expert is guided by the Scenario Elicitation tool to specify the objects that define a strategic scenario (e.g. the planned US invasion of the island of Okinawa in 1945), as presented in the next section. In the next phase the expert describes in English how he identifies center of gravity candidates for the defined scenario, using the Modeling tool, as illustrated in Section 5. Each specific reasoning step formulated with the Modeling tool is an example from which a general rule is learned using the Rule Learning tool, as described in section 6. These rules are only partially learned and are further refined with the Rule Refinement tool presented in section 7. The Problem Solving tool, presented in section 8, is used either in a mixed-initiative mode or in an autonomous mode, to identify strategic CoG candidates for the current scenario or for other scenarios.

These agent development phases follow a normal sequence from one phase to the other, but there is also the need to return to a previous phase to complete specific actions, as indicated by the back arrows from Figure 2. For example, while the expert and Disciple are performing mixed-initiative problem solving, the expert may need to define a new reduction that requires him to perform modeling, rule learning and rule refinement.

The rest of this paper follows these agent development phases, describing in more detail the corresponding Disciple tools/components.

4. The Scenario Elicitation Tool

The initial knowledge base of Disciple-RKF/CoG contains a generic object ontology which is based on the OKBC knowledge model [4]. A fragment of this ontology is represented in the right-top part of Figure 3, marked with thick lines. It includes descriptions of generic objects or concepts (such as Scenario, Force, Opposing_force, etc.), but no specific objects or instances. Associated with certain object concepts from this ontology are elicitation scripts that specify the questions to ask the expert in order to elicit instances of that concept. They also specify how to extend the ontology with the elicited knowledge.

The first step performed by the SME is to populate this ontology with instances and relationships that describe a strategic scenario. The expert does not need to see or understand this generic ontology. Instead, the expert-agent interaction takes place as illustrated in Figure 4, being directed by the execution of the elicitation scripts. The left part of the window is a table of contents, whose elements indicate various aspects of the scenario. When one such aspect is selected by the expert, its elicitation script is executed to acquire from the expert a description of that aspect, or to update a previously specified description. Initially, the script corresponding to the concept "Scenario" is executed, asking the expert to provide a name for the scenario (e.g. Okinawa_1945), a description in natural language, and the opposing forces (e.g. US_1945 and Japan_1945). A fragment of this elicitation script and of the generated instances are shown in Figure 3. After the opposing forces are specified, they

. . .
Multiple-instances-elicitation
  Prompt: Name the opposing forces in <*Scenario*>
  Answer: <*Opposing-force-k*>
  Actions:
    <*Opposing-force-k*> instance-of Opposing_force
    <*Scenario*> has_as_opposing_force <*Opposing-force-k*>
. . .

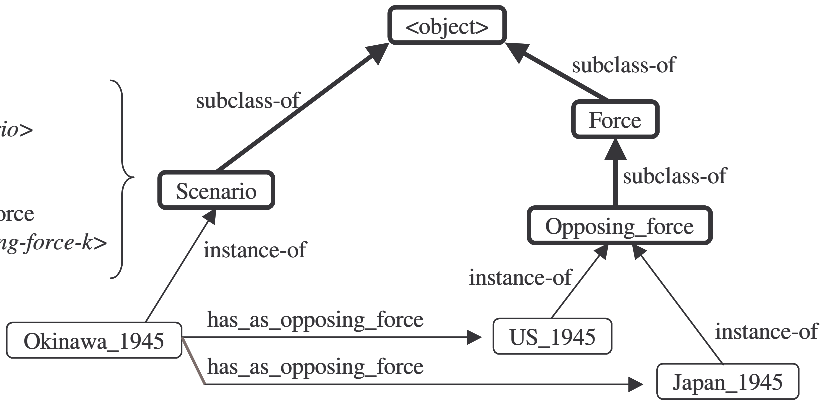
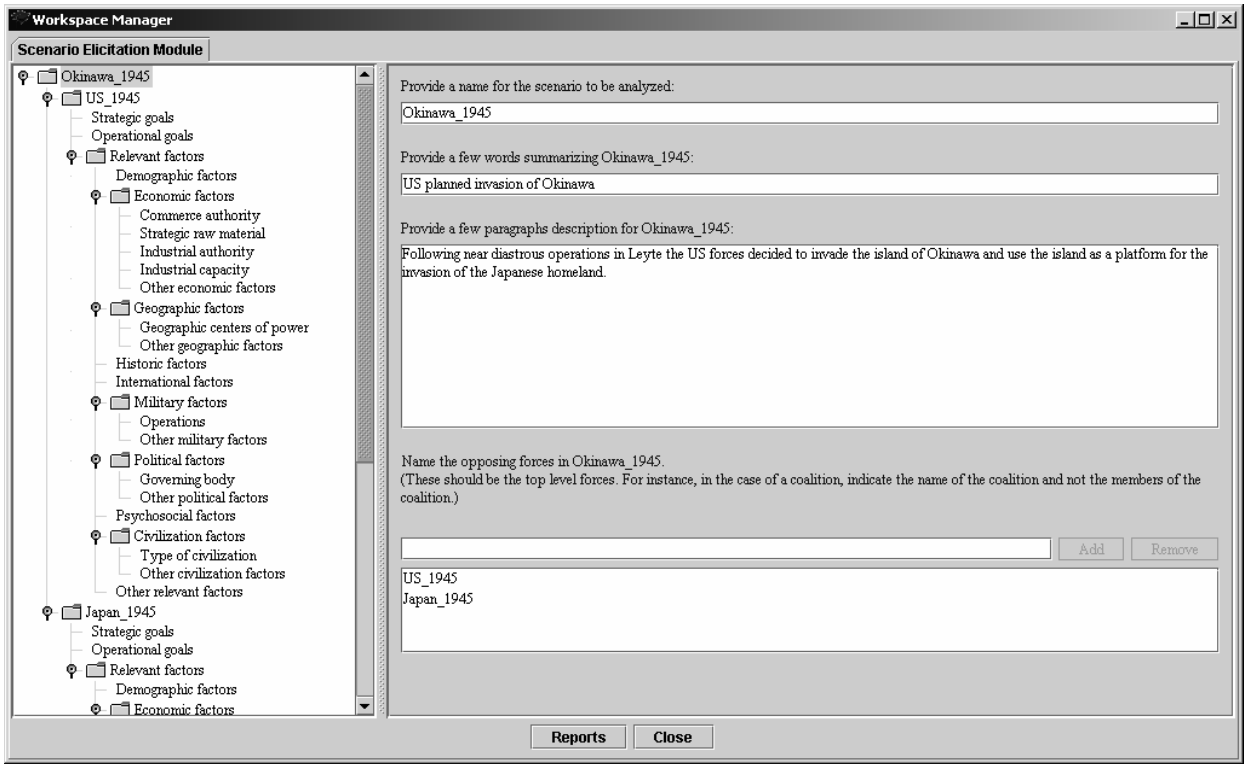


Figure 3: Fragment of the elicitation script and of the object ontology.



Provide a name for the scenario to be analyzed:
Okinawa_1945

Provide a few words summarizing Okinawa_1945:
US planned invasion of Okinawa

Provide a few paragraphs description for Okinawa_1945:
Following near diastrous operations in Leyte the US forces decided to invade the island of Okinawa and use the island as a platform for the invasion of the Japanese homeland.

Name the opposing forces in Okinawa_1945.
(These should be the top level forces. For instance, in the case of a coalition, indicate the name of the coalition and not the members of the coalition.)

US_1945
Japan_1945

Figure 4: The interface of the Scenario Elicitation tool.

also appear as titles in the table of contents, together with the characteristics that need to be elicited for them. If the expert clicks on any of these characteristics, its elicitation script is executed, guiding the expert to define the corresponding characteristic.

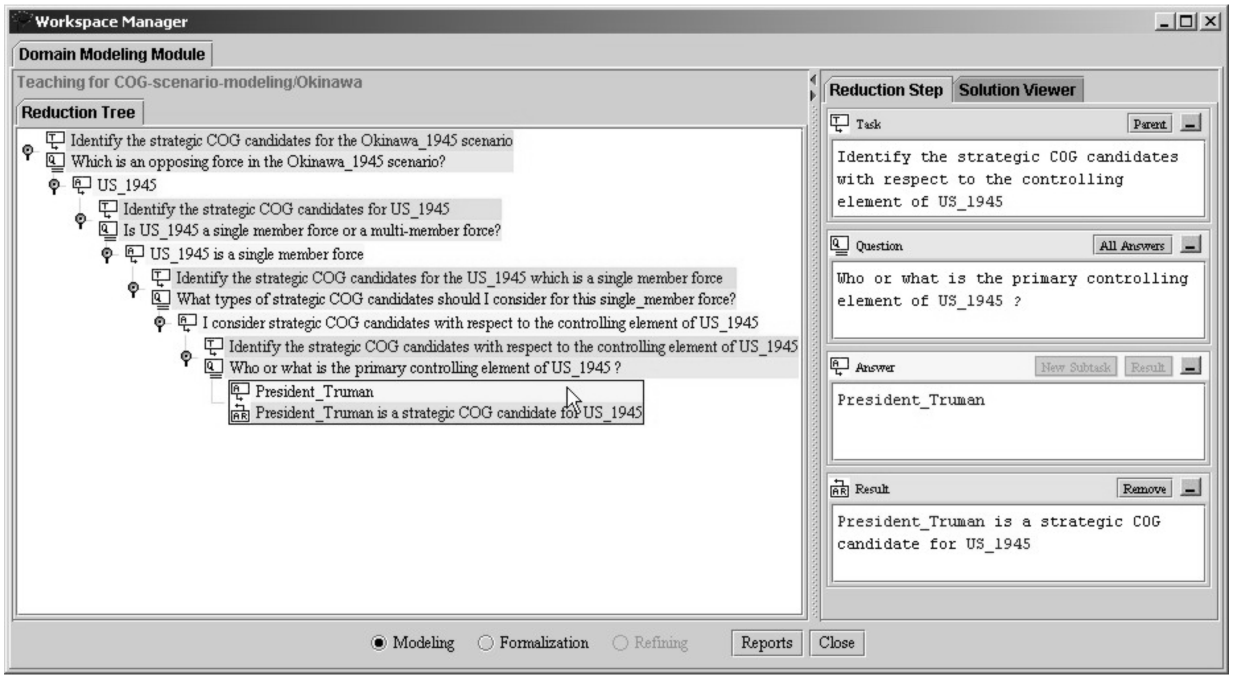
5. The Modeling Tool

After the expert has specified the Okinawa_1945 scenario using the Scenario Elicitation tool, he can start to teach the agent how to identify the strategic CoG candidates for this particular scenario. The Disciple problem-solving approach is based on the general task-reduction / solution-synthesis paradigm, being applicable in a wide variety of domains. In this paradigm, a task to be performed is successively reduced to simpler tasks until the tasks are simple enough to have immediate solutions. These solutions are then successively combined until the solution of the initial task is obtained. Therefore, the expert needs to express the process of identifying strategic CoG candidates for the Okinawa_1945 scenario, using this task reduction paradigm. We consider this to be the most difficult activity for the expert. The expert uses the Modeling Tool to express his reasoning in English, as illustrated in Figure 5. The left hand side of Figure 5 shows the task reduction steps that successively reduce the initial (top level) task to a solution. Regard it as a representation of how the expert might think aloud while solving this problem:

I need to
Identify the strategic COG candidates for the Okinawa_1945 scenario
    Which is an opposing force in the Okinawa_1945 scenario?
    US_1945
Therefore I need to
Identify the strategic COG candidates for US_1945
...
Therefore I conclude that
President_Truman is a strategic COG candidate for US_1945

Each reduction step from the left hand side is defined by filling in the panes from the right hand side. The top pane indicates the current task that needs to be reduced. The expert has to define a question that is relevant to the reduction of this task, then he has to answer the question, and finally, he has to reduce the task to a simpler one that incorporates the information from the answer. In the simple example illustrated in the right hand side of Figure 5, the expert has enough information to reduce the current task to a solution, which is also a solution of the top task from the left pane in Figure 5.

As mentioned above, in the modeling phase the expert uses natural language to express his reasoning process. The only restriction is that he has to use the exact



Identify the strategic COG candidates for the Okinawa_1945 scenario
Which is an opposing force in the Okinawa_1945 scenario?
US_1945
Identify the strategic COG candidates for US_1945
Is US_1945 a single member force or a multi-member force?
US_1945 is a single member force
Identify the strategic COG candidates for the US_1945 which is a single member force
What types of strategic COG candidates should I consider for this single_member force?
I consider strategic COG candidates with respect to the controlling element of US_1945
Identify the strategic COG candidates with respect to the controlling element of US_1945
Who or what is the primary controlling element of US_1945 ?
President_Truman
President_Truman is a strategic COG candidate for US_1945

Task: Identify the strategic COG candidates with respect to the controlling element of US_1945

Question: Who or what is the primary controlling element of US_1945 ?

Answer: President_Truman

Result: President_Truman is a strategic COG candidate for US_1945

Figure 5: The Modeling tool interface.

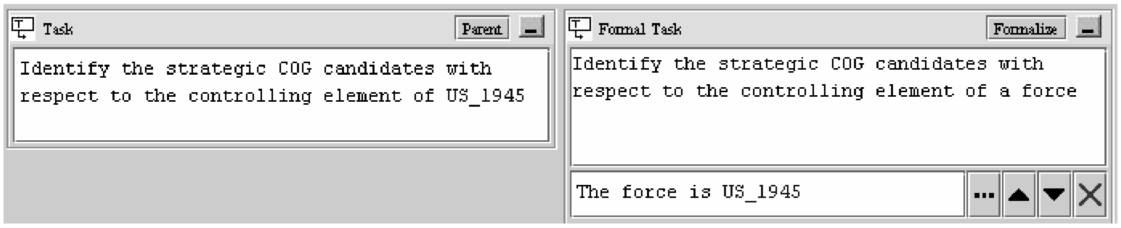
names of the instances from the knowledge base whenever he needs to refer to them. This is not a problem because of the name completion facilities of Disciple that only require the expert to provide a fragment of the name. Then the expert may choose from the different names suggested by Disciple.

6. The Rule Learning Tool

Each task reduction step like the one from the right hand side of Figure 5 represents an example from which Disciple will learn a general task reduction rule. This is achieved in three steps. First the expert and Disciple collaborate in creating a formalized representation for the tasks from the example. Then they cooperate in translating the question and the answer from natural language into formal explanation pieces. Finally, Disciple generalizes the formalized tasks and the explanation pieces into a plausible version-space rule that can be applied in future situations.

Figure 6 shows a fragment of the task formalization interface, where the informal task defined during domain modeling appears in the left hand side, and its formalization appears in the right hand side. The formalized task consists of an abstract phrase that represents the task name, and a set of specific phrases that represent the task features. The task name is generated by replacing each specific instance from the informal task with a general concept. Each task feature is generated by providing the identity of that general concept. Disciple can automatically propose the formalization of a task by using a general abstraction rule and the formalization of the parent task.

Figure 7 illustrates the interface of the Explanation Generation process during which the expert and Disciple find a set of formal explanation pieces that express the reason why the task reduction is done, reason that is informally expressed by the question/answer pair.

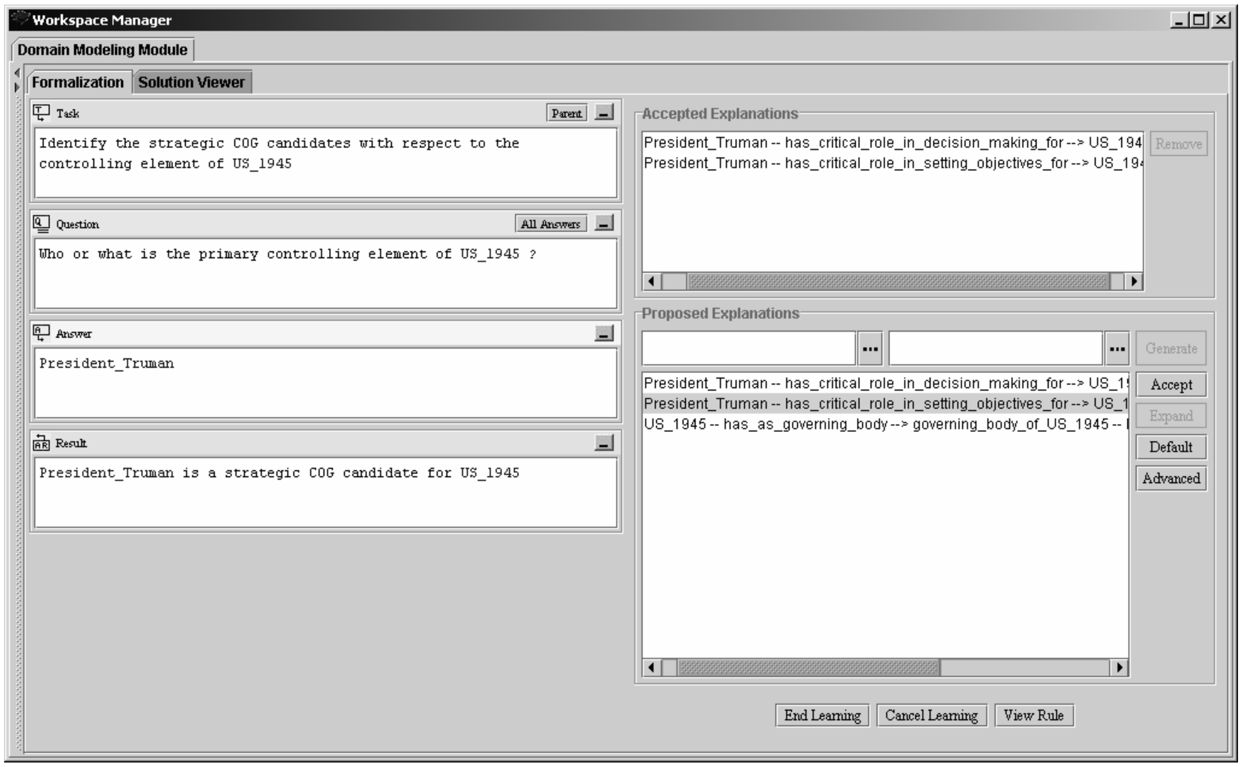


Task: Identify the strategic COG candidates with respect to the controlling element of US_1945

Formal Task: Identify the strategic COG candidates with respect to the controlling element of a force

The force is US_1945

Figure 6: A fragment of the task formalization interface.



Task: Identify the strategic COG candidates with respect to the controlling element of US_1945

Question: Who or what is the primary controlling element of US_1945 ?

Answer: President_Truman

Result: President_Truman is a strategic COG candidate for US_1945

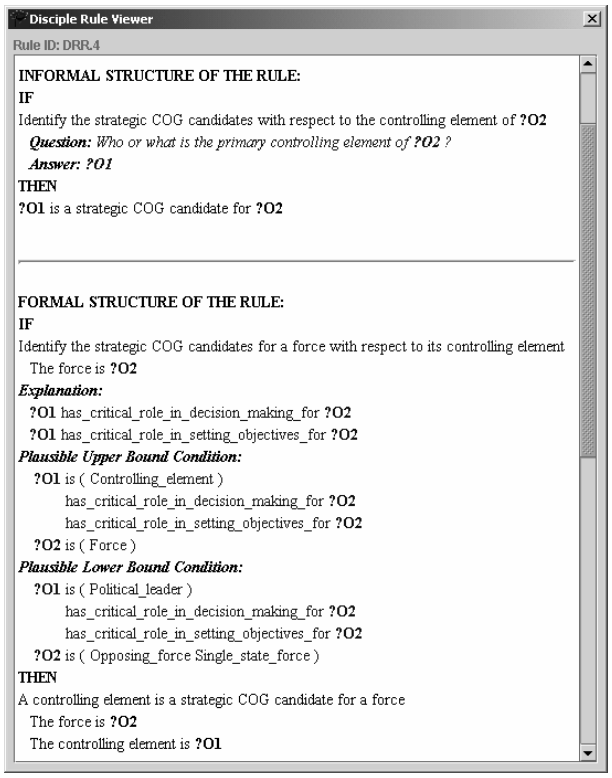
Accepted Explanations:
President_Truman -- has_critical_role_in_decision_making_for --> US_194
President_Truman -- has_critical_role_in_setting_objectives_for --> US_19

Proposed Explanations:
President_Truman -- has_critical_role_in_decision_making_for --> US_1
President_Truman -- has_critical_role_in_setting_objectives_for --> US_1
US_1945 -- has_as_governing_body --> governing_body_of_US_1945 --

Figure 7: The explanation generation dialog for the example from Figure 5.

Disciple uses several strategies to automatically propose explanation pieces, and to order them based on their plausibility. These strategies include analogy with previously learned rules, and hints from the question/answer pair. For instance, Disciple automatically proposed the three explanation pieces from the bottom right of Figure 7. The expert needs only to understand them and to select the correct ones. He can also guide the search for explanation pieces by giving additional hints.

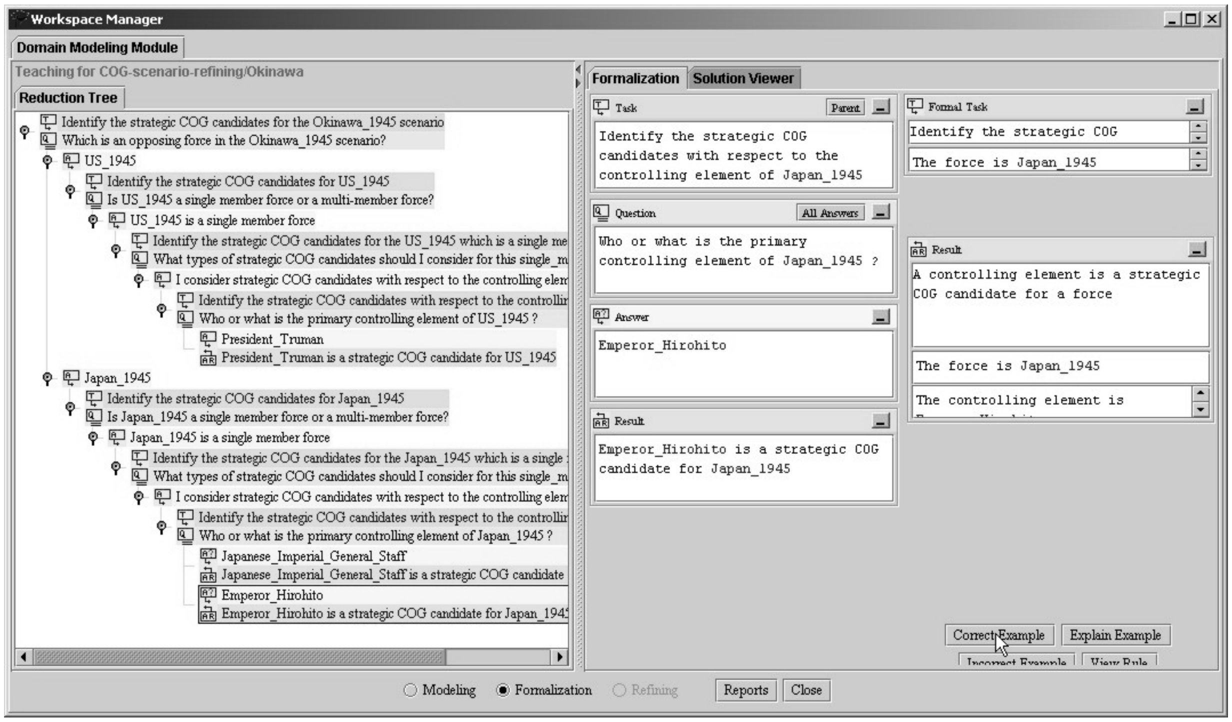
Using a form of generalization based on analogy, the agent generalizes the example and the two selected explanations pieces from Figure 7 into the IF-THEN task reduction rule shown in Figure 8. The top part of Figure 8 shows the informal structure of the rule that is obtained by simply replacing the instances from the example with variables. This form is used to communicate with the user. The bottom part of Figure 8 shows the formal structure of the rule which is used for reasoning. Notice that the formal structure of the rule has a plausible version space for the applicability condition of the rule. The plausible lower bound condition is the minimal generalization of the example and the explanation, generalization that is also constrained by the definitions of the relevant object features from the explanation. For instance, ?O1 has the features "has_critical_role_in_decision_making_for" and "has_critical_role_ in_setting_objectives_for". Therefore,



INFORMAL STRUCTURE OF THE RULE:
IF
Identify the strategic COG candidates with respect to the controlling element of ?O2
Question: Who or what is the primary controlling element of ?O2 ?
Answer: ?O1
THEN
?O1 is a strategic COG candidate for ?O2

FORMAL STRUCTURE OF THE RULE:
IF
Identify the strategic COG candidates for a force with respect to its controlling element
The force is ?O2
Explanation:
?O1 has_critical_role_in_decision_making_for ?O2
?O1 has_critical_role_in_setting_objectives_for ?O2
Plausible Upper Bound Condition:
?O1 is (Controlling_element)
has_critical_role_in_decision_making_for ?O2
has_critical_role_in_setting_objectives_for ?O2
?O2 is (Force)
Plausible Lower Bound Condition:
?O1 is (Political_leader)
has_critical_role_in_decision_making_for ?O2
has_critical_role_in_setting_objectives_for ?O2
?O2 is (Opposing_force Single_state_force)
THEN
A controlling element is a strategic COG candidate for a force
The force is ?O2
The controlling element is ?O1

Figure 8: The learned rule.



```
Identify the strategic COG candidates for the Okinawa_1945 scenario
Which is an opposing force in the Okinawa_1945 scenario?
  US_1945
    Identify the strategic COG candidates for US_1945
    Is US_1945 a single member force or a multi-member force?
      US_1945 is a single member force
        Identify the strategic COG candidates for the US_1945 which is a single me
        What types of strategic COG candidates should I consider for this single_m
          I consider strategic COG candidates with respect to the controlling elem
            Identify the strategic COG candidates with respect to the controllir
            Who or what is the primary controlling element of US_1945 ?
              President_Truman
              President_Truman is a strategic COG candidate for US_1945
  Japan_1945
    Identify the strategic COG candidates for Japan_1945
    Is Japan_1945 a single member force or a multi-member force?
      Japan_1945 is a single member force
        Identify the strategic COG candidates for the Japan_1945 which is a single
        What types of strategic COG candidates should I consider for this single_m
          I consider strategic COG candidates with respect to the controlling elem
            Identify the strategic COG candidates with respect to the controllir
            Who or what is the primary controlling element of Japan_1945 ?
              Japanese_Imperial_General_Staff
              Japanese_Imperial_General_Staff is a strategic COG candidate
              Emperor_Hirohito
              Emperor_Hirohito is a strategic COG candidate for Japan_194
```

Task: Identify the strategic COG candidates with respect to the controlling element of Japan_1945

Question: Who or what is the primary controlling element of Japan_1945 ?

Answer: Emperor_Hirohito

Result: Emperor_Hirohito is a strategic COG candidate for Japan_1945

Formal Task: Identify the strategic COG / The force is Japan_1945

Result: A controlling element is a strategic COG candidate for a force / The force is Japan_1945 / The controlling element is

Figure 9: Rule refinement during cooperative problem solving.

the lower bound of ?O1 has to be included into the lower bounds of these features' domains (where the domain of a feature is the set of possible values for that feature). Additional constraints are imposed by the ranges (i.e. the set of possible values) of the features. Similarly, the upper bound is the maximal generalization of the example and the explanation.

During rule refinement, the lower bound will be generalized and the upper bound will be specialized, converging toward one another, as discussed in the next section.

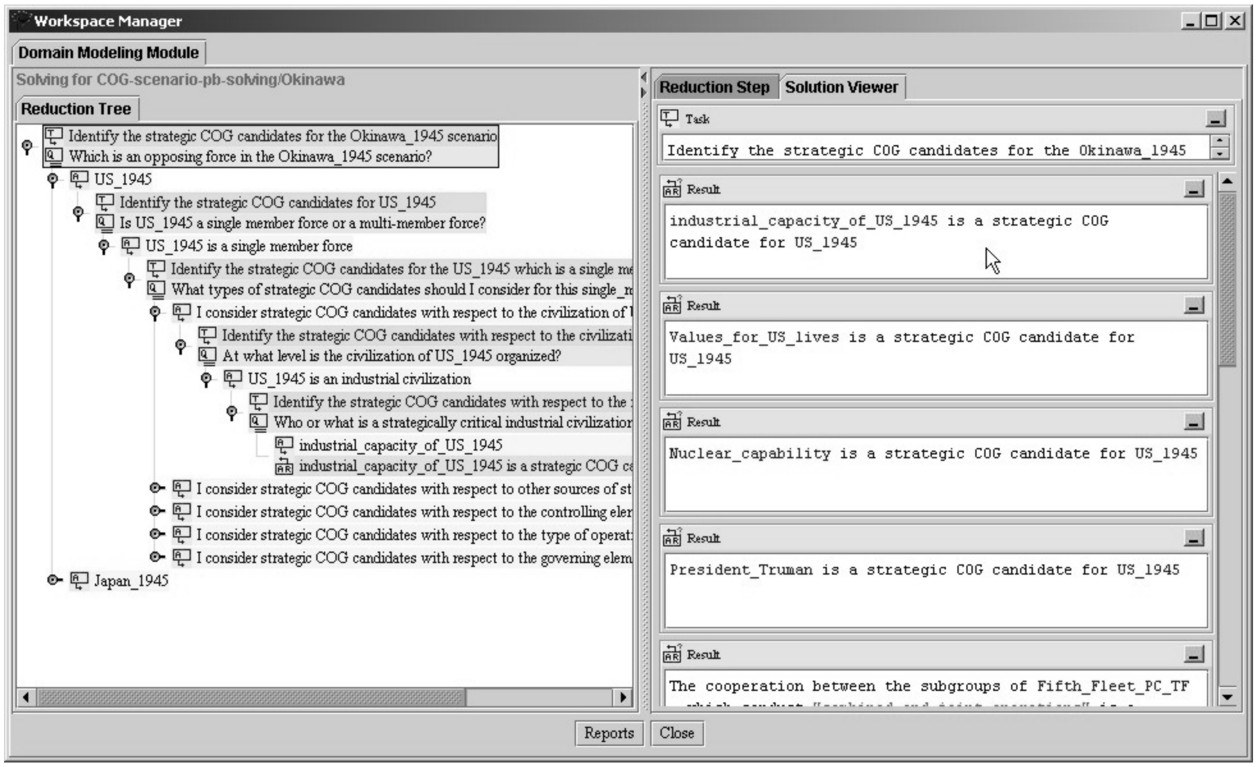
7. The Rule Refinement Tool

There are several types of interactions between the expert and Disciple. Initially the expert teaches Disciple how to solve problems and Disciple generates partially learned rules from the examples provided and explained by the expert, as indicated in the previous section. As Disciple learns from the expert, the interaction between the expert and Disciple evolves from a teacher-student interaction, toward an interaction where they collaborate in solving a problem and Disciple learns, not only from the contributions of the expert, but also from its own successful or unsuccessful problem solving attempts.

For instance, Figure 9 shows an interface of the Rule Refinement tool. The first half of left part of the Figure shows a line of reasoning that was indicated by the expert. From each task reduction step (represented by a task, a question, an answer and its subtask), Disciple has learned a plausible version space rule. These rules allowed Disciple to propose the line of reasoning from the bottom half of Figure 9 that has led to two additional solutions of the initial task. The expert has to analyze this reasoning process proposed by Disciple and to confirm or reject it. When the expert selects a reasoning step in the left hand side pane, the details of this step are shown in the right hand side pane. The expert has various options. He can simply accept it, by pressing the "Accept" button. In this case the plausible lower bound of the corresponding rule is empirically generalized to cover this example.

Alternatively, the expert may reject the example. Then the corresponding rule will have to be specialized in order to no longer generate such an incorrect example. Disciple may perform empirical specializations of the conditions, if no failure explanations are identified. However, if the expert and Disciple identify some failure explanation, then the rule is automatically specialized by Disciple, either by specializing the conditions, or by adding an Except-when plausible version space condition. An Except-when condition is a condition that should not be satisfied in order for the rule to be applicable. Such refinements lead to complex task reduction rules.

We have discussed above the case of a local rule refinement. However, there are other strategies that can be used by the Rule Refinement tool. For instance, the expert may select and accept an entire line of reasoning



Solving for COG-scenario-pb-solving/Okinawa

```
Identify the strategic COG candidates for the Okinawa_1945 scenario
Which is an opposing force in the Okinawa_1945 scenario?
  US_1945
    Identify the strategic COG candidates for US_1945
    Is US_1945 a single member force or a multi-member force?
      US_1945 is a single member force
        Identify the strategic COG candidates for the US_1945 which is a single me
        What types of strategic COG candidates should I consider for this single_m
          I consider strategic COG candidates with respect to the civilization of U
            Identify the strategic COG candidates with respect to the civilizati
            At what level is the civilization of US_1945 organized?
              US_1945 is an industrial civilization
                Identify the strategic COG candidates with respect to the i
                Who or what is a strategically critical industrial civilization
                  industrial_capacity_of_US_1945
                  industrial_capacity_of_US_1945 is a strategic COG ca
          I consider strategic COG candidates with respect to other sources of st
          I consider strategic COG candidates with respect to the controlling eler
          I consider strategic COG candidates with respect to the type of operat
          I consider strategic COG candidates with respect to the governing elem
  Japan_1945
```

```
Task
Identify the strategic COG candidates for the Okinawa_1945

Result
industrial_capacity_of_US_1945 is a strategic COG
candidate for US_1945

Result
Values_for_US_lives is a strategic COG candidate for
US_1945

Result
Nuclear_capability is a strategic COG candidate for US_1945

Result
President_Truman is a strategic COG candidate for US_1945

Result
The cooperation between the subgroups of Fifth_Fleet_PC_TF
```

Figure 10: The interface of the problem solving tool.

generated by the agent, leading to the automatic refinement of all the applied rules.

One should notice that, as in the case of rule learning, the expert does not and will not work with the rules, but only with specific examples. Therefore, the complex knowledge engineering operations of defining and debugging problem solving rules are replaced in the Disciple approach with the much simpler operations of defining and critiquing specific examples.

8. The Problem Solving Tool

After the Disciple agent has been trained, it can be used in autonomous problem-solving mode, either by the expert who has taught it, or by a non-expert user. Figure 10 shows the interface of the Problem Solving tool. The right hand side pane shows a task and all its solutions. In the left hand side pane the expert can see the reasoning steps that have led to each of these solutions.

9. Synergistic Integration of the Modeling, Learning and Problem Solving Tools

The interaction between the expert and Disciple is based on mixed-initiative reasoning and a dynamic shift of initiative and control that result from a division of responsibility between them for the operations that need to be performed in order to develop the knowledge base. Each party is responsible for performing those operations that are easier for that party, while also receiving help from the other party. Such a complex behavior was accomplished by a synergistic integration of the tools that allow a natural move from one tool to another. This integration was done not only at the level of the interface but also at the more difficult level of knowledge representation and management. The interfaces of these tools have the same look and feel, with only minor variations to allow for the specific functionality of each tool. There is also the capability to easily update modifications from one tool to another.

10. Experimental Results

Disciple-RKF/CoG has been experimentally used and evaluated in two courses at the US Army War College (USAWC). During two successive sections of the course "Case Studies in Center of Gravity Determination," the students (senior military officers) have used the Scenario Elicitation and Modeling tools. While the students have been taught how to use these Disciple tools, they have not been provided with any other knowledge of Artificial Intelligence. At the end of the course they have filled in a questionnaire for each of these modules. On a 5-point scale, from strongly disagree to strongly agree, 8 out of 10 students of the Winter-2001 term section agreed that the Scenario Elicitation tool is easy to learn and easy to use. Similarly, 3 out of 3 students of the Spring term section agreed or strongly agreed with this statement. Another significant result is that 6 out of 8 students (in the Winter section) agreed that SMEs who are not computer scientists can learn to express their reasoning process using the task reduction paradigm. In the Spring section, 3 out of 3 students agreed or strongly agreed with this statement.

A more complete evaluation of Disciple-RKF/CoG was performed as part of the "Military Applications of Artificial Intelligence" course, during the Spring-2001 term. In this course the students have been given a general overview of Artificial Intelligence, and have been taught to use all the tools of the Disciple-RKF/CoG shell. The students have been organized in five two-student teams, each team being given the project of using Disciple to develop an intelligent agent for CoG identification, based on a different strategic scenario. The scenarios, carried forward from the previous CoG class, were the following ones: 1) the Falklands war between Argentina and Britain in 1982; 2) the OECS stabilization mission in the Grenada Island in 1982; 3) the Inchon landing during the Korean War in 1950; 4) the capture of the Leyte Island by the US forces in 1944; and 5) the US invasion of Panama in December 1989. Each team succeeded to develop a CoG agent that no only generated CoG candidates for its own training scenario, but also for the other scenarios.

In the last two 3-hour class sessions, all the teams have participated into a controlled agent development experiment that was entirely videotaped. Each team was provided with a copy of Disciple-RKF/CoG that contained a generic object ontology, but no specific instances, no tasks and no rules. They have also been given a 7-page report describing the Okinawa scenario, and have been asked to train their Disciple shell to identify center of gravity candidates, based on that scenario. After each significant phase of agent training and knowledge base development (e.g. scenario specification, modeling, rule learning, and rule refinement) their work was reviewed by a knowledge engineer, and any necessary corrections were performed by them under the guidance of the knowledge engineer. Each team succeeded to develop an end-to-end agent, in a very short time, as indicated in Figure 11.

The top part of Figure 11 shows the size of the initial generic object ontology. Each team used the Scenario Elicitation tool to populate this ontology with different instances and features, Figure 11 indicating both the time spent by each team, and the number of knowledge elements defined during this time. On average they defined 85.40 instances and 93.80 feature values in 1hour and 6 minutes. After that, each team has taught Disciple to identify CoG candidates in the Okinawa scenario. The number of rules learned from each team, together with the corresponding time, are also indicated in Figure 11. The average number of rules per team was 18.80, and the average time interval was 4 hours and 7 minutes. While obviously incompletely developed (both because of the use of a single training scenario, and of an incomplete training for that scenario), the knowledge bases were good enough for identifying correct CoG candidates for the Okinawa scenario, and even for identifying reasonable CoG candidates for the project scenarios.

At the end of this experiment, the students have also completed a detailed questionnaire, containing questions about the main components of Disciple. One of the most

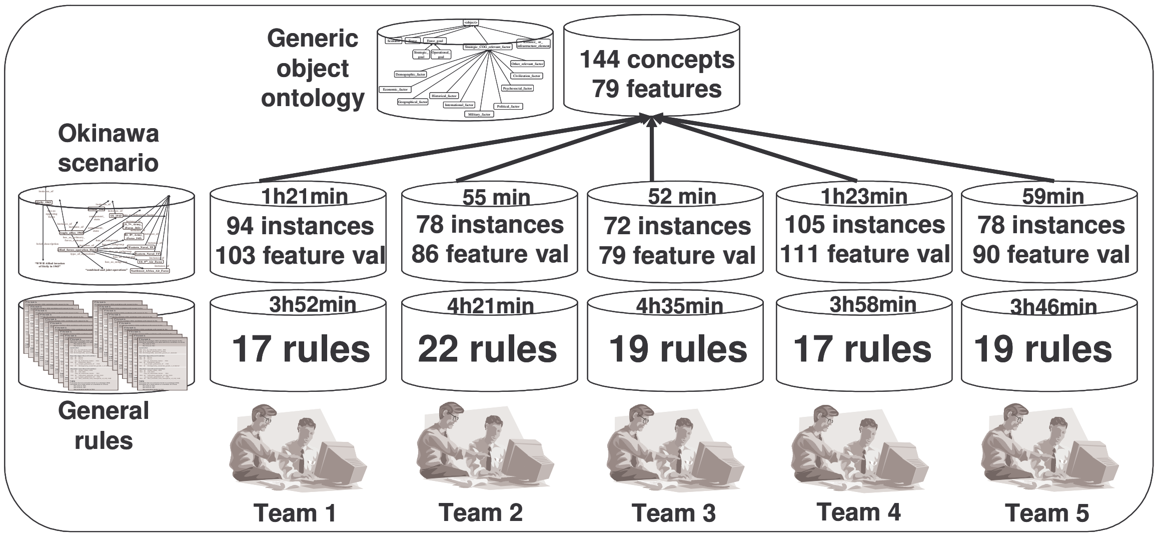


Figure 11: Knowledge bases developed during the experiment.

significant results was that 7 out of the 10 experts agreed, 1 expert strongly agreed and 2 experts were neutral with respect to the statement: "I think that a subject matter expert can use Disciple to build an agent, with limited assistance from a knowledge engineer."

We consider this experiment as being a very significant success. Indeed, to our knowledge, this is the first time that subject matter experts have developed end-to-end knowledge-based agents, with very limited assistance from a knowledge engineer.

11. Conclusions and Future Developments

Successive versions of the Disciple approach have been evaluated in several intensive studies requiring the rapid development and maintenance of KBs for solving the workaround challenge problem, the CoA challenge problem and the CoG challenge problem.

For the CoG challenge problem, SMEs with little or no knowledge engineering experience developed complete knowledge bases and agents by themselves, with very limited assistance from knowledge engineers. The successful use of Disciple in the two courses held at the US Army War College and the results of the performed experiments have led to the decision to continue this activity during the next academic year. In addition, we plan to develop a Disciple-based tutoring system. This Disciple agent will be taught by CoG experts from the US Army War College and will teach the students about the CoG identification process in a way that is similar to how they have been taught by the CoG experts.

Future planned developments of Disciple include a tool for scripts elicitation, extensions of the Scenario Elicitation tool to allow the expert to define not only instances but also new concepts and general features, more powerful analogical methods for explanation generation, the use of natural language processing methods for extracting better hints from the question and the answer, and a simpler interface that will provide more help and guidance to the expert.

To conclude, our long term vision for Disciple, that guides our future work, is to evolve it to a point where it will allow typical computer users to build and maintain knowledge bases and agents, as easily as they now use personal computers for text processing.

Acknowledgements

This research has been performed in the GMU Learning Agents Laboratory and was sponsored by DARPA, AFRL, AFMC, USAF, under agreement number F30602-00-2-0546, by the AFOSR under grant no. F49620-00-1-0072, and by the US Army. Michael Bowman, Catalin Balan, Xianjun Hao and other members of the LALAB have contributed to successive versions of Disciple. Tony Lopez and Jim Donlon, from the US Army War College Center for Strategic Leadership have contributed to the application of Disciple to the COG challenge problem.

References

1) Boicu M., Wright K., Marcu D., Lee S.W., Bowman M. and Tecuci G. (1999). The Disciple Integrated Shell and Methodology for Rapid Development of Knowledge-Based Agents. In AAAI-99/IAAI-99 Proceedings, pp. 900–901, Menlo Park, CA: AAAI Press.
2) Buchanan, B. G. and Wilkins, D. C. (editors), (1993). Readings in Knowledge Acquisition and Learning: Automating the Construction and Improvement of Expert Systems, Morgan Kaufmann, San Mateo, CA.
3) Burke, M., Rapid Knowledge Formation (RKF) Program Description, http://www.darpa.mil/ito/research/rkf/
4) Chaudhri, V. K., Farquhar, A., Fikes, R., Park, P. D., and Rice, J. P. (1998). OKBC: A Programmatic Foundation for Knowledge Base Interoperability. In Proceedings of the Fifteenth National Conference on Artificial Intelligence, 600–607, Menlo Park, CA: AAAI Press.
5) Clancey, W. J. (1984). NEOMYCIN: Reconfiguring a rule-based system with application to teaching. In Clancey W. J. and Shortliffe, E. H., eds. Readings in Medical Artificial Intelligence, pp.361-381. Reading, MA: Addison-Wesley.
6) Cohen P., Schrag R., Jones E., Pease A., Lin A., Starr B., Gunning D., and Burke M. (1998). The DARPA High-Performance Knowledge Bases Project, AI Magazine, 19(4), 25-49
7) Giles, P.K., and Galvin, T.P. (1996). Center of Gravity: Determination, Analysis and Application. CSL, U.S. Army War College, PA: Carlisle Barracks.
8) Tecuci, G. (1998). Building Intelligent Agents: An Apprenticeship Multistrategy Learning Theory, Methodology, Tool and Case Studies. London, England: Academic Press.
9) Tecuci, G., Boicu, M., Wright, K., Lee, S. W., Marcu, D. and Bowman, M. (1999). An Integrated Shell and Methodology for Rapid Development of Knowledge-Based Agents. In Proceedings of the Sixteenth National Conference on Artificial Intelligence, 250-257, Menlo Park, CA: AAAI Press.
10) Tecuci G., Boicu M., Bowman M., and Marcu D., with a commentary by Burke M., (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.