

**AFRL-RI-RS-TR-2008-148**  
**Final Technical Report**  
**May 2008**



**INTELLIGENT ASSISTANTS FOR DISTRIBUTED  
KNOWLEDGE ACQUISITION, INTEGRATION,  
VALIDATION, AND MAINTENANCE**

**George Mason University**

*APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.*

**STINFO COPY**

**AIR FORCE RESEARCH LABORATORY  
INFORMATION DIRECTORATE  
ROME RESEARCH SITE  
ROME, NEW YORK**

## NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the Air Force Research Laboratory Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (<http://www.dtic.mil>).

AFRL-RI-RS-TR-2008-148 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/

/s/

PETER J. ROCCI, JR.  
Work Unit Manager

JOSEPH CAMERA, Chief  
Information & Intelligence Exploitation Division  
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

# REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Service, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.

**PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

<b>1. REPORT DATE (DD-MM-YYYY)</b> MAY 2008	<b>2. REPORT TYPE</b> Final	<b>3. DATES COVERED (From - To)</b> Sep 04 – Feb 08
--	--------------------------------	--

<b>4. TITLE AND SUBTITLE</b>  INTELLIGENT ASSISTANTS FOR DISTRIBUTED KNOWLEDGE ACQUISITION, INTEGRATION, VALIDATION, AND MAINTENANCE	<b>5a. CONTRACT NUMBER</b>
	<b>5b. GRANT NUMBER</b> FA8750-04-1-0257
	<b>5c. PROGRAM ELEMENT NUMBER</b> 62702F

<b>6. AUTHOR(S)</b>  Gheorghe Tecuci Mihai Boicu	<b>5d. PROJECT NUMBER</b> SMET
	<b>5e. TASK NUMBER</b> 00
	<b>5f. WORK UNIT NUMBER</b> 01

<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b> George Mason University 4400 University Drive, MSN 4B2 Fairfax VA 22030-4422	<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>
---	---

<b>9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b> AFRL/RIED 525 Brooks Rd Rome NY 13441-4505	<b>10. SPONSOR/MONITOR'S ACRONYM(S)</b>
	<b>11. SPONSORING/MONITORING AGENCY REPORT NUMBER</b> AFRL-RI-RS-TR-2008-148

**12. DISTRIBUTION AVAILABILITY STATEMENT**  
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# WPAFB 08-3334

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**  
This research has developed an integrated set of tools, called Disciple 2008 learning agent shell, for continuous acquisition of knowledge directly from subject matter experts, and for the integration, validation, and maintenance of the acquired knowledge. During mixed-initiative problem solving sessions, Intelligent Disciple assistants supporting individual subject matter experts acquire their problem solving expertise by employing an integrated set of learning, maintenance and validation methods. The acquired knowledge is shared by the subject matter experts and their Disciple assistants in a distributed and hierarchical repository. The Disciple 2008 learning agent shell was used to develop and transition Disciple-COG, a problem solving and learning agent for center of gravity analysis. Disciple-COG is regularly used in courses at the US Army War College and was also successfully used at the Air War College. The transition of Disciple-COG is supported by lecture notes and the text book "Agent-assisted Center of Gravity Analysis" that provides both a detailed description of the developed center of gravity analysis approach, and step by step instructions for using Disciple-COG.

**15. SUBJECT TERMS**  
artificial intelligence, intelligent assistant, knowledge acquisition and maintenance, knowledge base integration, mixed-initiative reasoning, subject matter expert, knowledge engineering, center of gravity analysis

<b>16. SECURITY CLASSIFICATION OF:</b>			<b>17. LIMITATION OF ABSTRACT</b>  UU	<b>18. NUMBER OF PAGES</b>  39	<b>19a. NAME OF RESPONSIBLE PERSON</b> Peter J. Rocci
<b>a. REPORT</b> U	<b>b. ABSTRACT</b> U	<b>c. THIS PAGE</b> U			<b>19b. TELEPHONE NUMBER (Include area code)</b> N/A

# Table of Contents

1. Introduction .....	1
2. Disciple Approach to Knowledge Acquisition and Agent Development .....	2
3. Disciple 2008 Learning Agent Shell .....	4
4. Development of a Disciple Agent for a Specific Military Application .....	6
5. Continuous Knowledge Acquisition and Maintenance .....	10
6. Distributed Knowledge Acquisition and Integration .....	15
7. Knowledge Acquisition and Validation .....	18
8. Methodology for Distributed Knowledge Acquisition, Integration, Validation and Maintenance .....	19
9. Transition of Disciple-COG .....	22
10. Other Applications of Disciple Agents .....	27
11. Future Research Directions .....	28
Appendices	
A1. Publications Describing the Results of this Project .....	29
A2. Agent-Assisted Center of Gravity Analysis Text Book: Summary and Table of Contents.....	32
A3. Disciple Shell 2008 and Disciple-COG CD-ROM .....	34

## List of Figures

Figure 1: Overall architecture of the Disciple 2008 learning agent shell .....	5
Figure 2: Customization of the Disciple shell into Disciple-COG .....	7
Figure 3: Computational approach to center of gravity analysis .....	8
Figure 4: Problem reduction paradigm for center of gravity analysis problem in Figure 3 .....	9
Figure 5: Critical vulnerability analysis of a friendly COG by an enemy force.....	10
Figure 6: Integrated methods for continuous knowledge acquisition and maintenance.....	11
Figure 7: Rule learning and refinement from three examples .....	14
Figure 8: The organization of the knowledge base of a Disciple agent.....	16
Figure 9: Distributed knowledge bases development and integration .....	17
Figure 10: Interface of the situation assessment module .....	23
Figure 11: Interface of the mixed-initiative reasoner .....	25
Figure 12: Vulnerability assessment interface.....	25
Figure 13: Fragment of an automatically generated report.....	26
Figure 14: Subjective evaluation of Disciple-COG .....	27

### Acknowledgements

We gratefully acknowledge the support from the Air Force Research Laboratory and the guidance provided by Mr. Peter Rocci, the program manager for this project. We are also grateful to our collaborators from the senior service colleges and particularly to Dr. Jerome Comello and Mr. William Cleckner (from the US Army War College), Dr. Joseph Strange (from the Marine Corps War College) and Col. Jeffrey Hightaian (from the Air War College).

## **1. Introduction**

The goal of this project was to build on the Disciple learning agent technology in order to develop an integrated set of tools for continuous acquisition of knowledge directly from subject matter experts, and for the integration, validation, and maintenance of the acquired knowledge. This included the application of the tools to the development and transition of a problem solving and learning agent for center of gravity analysis.

The result of this project is a significant extension of the Disciple learning agent technology, an integrated learning agent shell for rapid development of knowledge-based learning agents for complex military problems, and an illustrative example of such an agent for center of gravity analysis. This agent, called Disciple-COG, is regularly used in courses at the US Army War College and was also successfully used at the Air War College. To further facilitate its use we have written the text book “Agent-assisted Center of Gravity Analysis” (see Appendix A2). This book provides both a detailed description of the developed center of gravity analysis approach, and step by step instructions for using Disciple-COG. The CD accompanying the book includes the Disciple-COG agent and lecture notes supporting its use in courses at senior service colleges.

The next sections present the main results of this project. Section 2 provides an overview of the Disciple approach to knowledge acquisition and agent development. Section 3 presents the Disciple 2008 learning agent shell, which is the developed integrated set of tools for knowledge acquisition, integration, validation and maintenance. Section 4 presents the methodology for developing a Disciple agent for a specific military problem, such as Disciple-COG which was developed for center of gravity analysis. Section 5 presents an overview of the methods that were developed for continuous knowledge acquisition and maintenance. Section 6 presents the developed approach for distributed knowledge acquisition and integration. Section 7 discusses validation methods. Section 8 presents an overview of the Disciple methodology for distributed

knowledge acquisition, integration, validation and maintenance. Section 9 presents the Disciple-COG agent and some results of its use at the Air War College. Finally, Section 10 discusses some limitations of the developed methods and tools and proposes several future research directions.

The Appendices include the list of publications that further describe our results, the table of content of the Disciple-COG book, and the content of the CD with the deliverables of this project.

## **2. Disciple Approach to Knowledge Acquisition and Agent Development**

The Disciple approach denotes an evolving theory and associated methodologies and tools aimed at allowing subject matter experts that do not have prior knowledge engineering experience to build knowledge-based systems by themselves, with no or very limited assistance from computer scientists or knowledge engineers. The basic idea of the Disciple approach is to develop a learning agent shell (see next section) that can be taught directly by a subject matter expert to become a knowledge-based assistant. The expert interacts directly with a Disciple agent, to teach it to solve problems in a way that is similar to how the expert would teach a student or an apprentice. For instance, the expert will formulate a specific problem and will show the agent the reasoning steps to solve it, helping the agent to understand them. Each problem solving step represents an example from which the agent learns a general rule, with the help of the subject matter expert. As Disciple learns new rules and concepts from the expert, the interaction between the expert and Disciple evolves from a teacher-student interaction, toward an interaction where both collaborate in problem-solving. During this phase, Disciple learns not only from the contributions of the expert, but also from its own successful or unsuccessful problem solving attempts, which lead to the refinement of the learned knowledge. This process is based on:

- *Mixed-initiative problem solving* (Tecuci et al., 2007b,c; Aha and Tecuci, 2005; Boicu M. et al, 2005a), where the expert solves the more creative parts of the problem and the agent solves the more routine ones.
- *Integrated learning and teaching* (Boicu C. 2006; Boicu C. et al., 2005a; Tecuci et al., 2005a), where the expert helps the agent to learn (for instance, by providing examples, hints and explanations), and the agent helps the expert to teach it (for instance, by asking relevant questions).
- *Multistrategy learning* (Tecuci et al., 2008b; Boicu C. et al, 2007; Tecuci et al., 2007b; Boicu and Tecuci, 2005c), where the agent integrates complementary strategies, such as learning from examples, learning from explanations, and learning by analogy, to learn general concepts and rules.

The Disciple agents employ a very general, divide-and-conquer, approach to problem solving, called problem-reduction/solution-synthesis (Tecuci et al., 2008a), which is applicable in a wide range of domains. In this approach, a complex problem is successively reduced to simpler and simpler problems, the solutions of the simplest problems are found, and then these solutions are successively composed, from bottom up, until the solution of the initial problem is obtained. To exhibit this type of problem solving behavior, the domain knowledge base is structured into an object ontology (which describes the objects from an application domain) and a set of problem solving rules (expressed with the objects from the ontology). One type of rule is the problem reduction rule which expresses how and under what conditions a generic problem can be reduced to simpler generic problems. Another type of rule is the solution synthesis rule which expresses how and under what conditions the solutions of generic subproblems can be combined into the solution of a generic problem. The conditions of the rules may be complex first-order logical expressions, but they are learned by the Disciple agents rather than being programmed by a knowledge engineer (as illustrated in Section 5).

The problem-reduction/solution-synthesis approach greatly facilitates distributed problem solving and knowledge acquisition. Indeed, various subproblems of a given problem could be solved by different users and their Disciple assistants. Then the solutions of the subproblems could be combined into the solution of the given problem. Similarly, each of the Disciple agent can learn from its user, and the acquired knowledge can then be integrated.

*Disciple aims to be a comprehensive, end-to-end approach, to the development and maintenance of adaptive knowledge-based agents that reason and learn.* Therefore, its associated methodology covers all the phases of knowledge bases and agents development and use, from the initial analysis and modeling of the reasoning processes of the subject matter experts, to ontology specification, import and development, to agent teaching, ontology learning, rule learning, to mixed-initiative and autonomous problem solving, to knowledge bases integration, validation, and maintenance, to knowledge base export, and to agent use.

### **3. Disciple 2008 Learning Agent Shell**

An expert system shell is a tool for building expert systems. It consists of a general inference engine for a certain class of problems (e.g. planning, design, diagnosis, monitoring, prediction, interpretation) and supports a representation formalism in which the knowledge base can be encoded. If the inference engine is adequate for a certain type of problems (e.g. planning), then the process of building an expert system for that type of problems is reduced to the building of the knowledge base. Unfortunately, it is the building of the knowledge base which is the most complex, time-consuming, and error-prone part of building the system. To overcome this knowledge acquisition bottleneck the concept of learning agent shell has been introduced. A learning agent shell is a new type of software tool that extends and generalizes an expert system

shell with a powerful learning engine that allows the system to learn its problem solving expertise directly from a subject matter expert.

The Disciple 2008 learning agent shell, the architecture of which is presented in Figure 1, is not only able to acquire knowledge directly from subject matter experts, but it is also much more broadly applicable than an expert system shell because its problem solving approach (problem-reduction/solution-synthesis) is not restricted to a given type of problem (such as analysis or design).

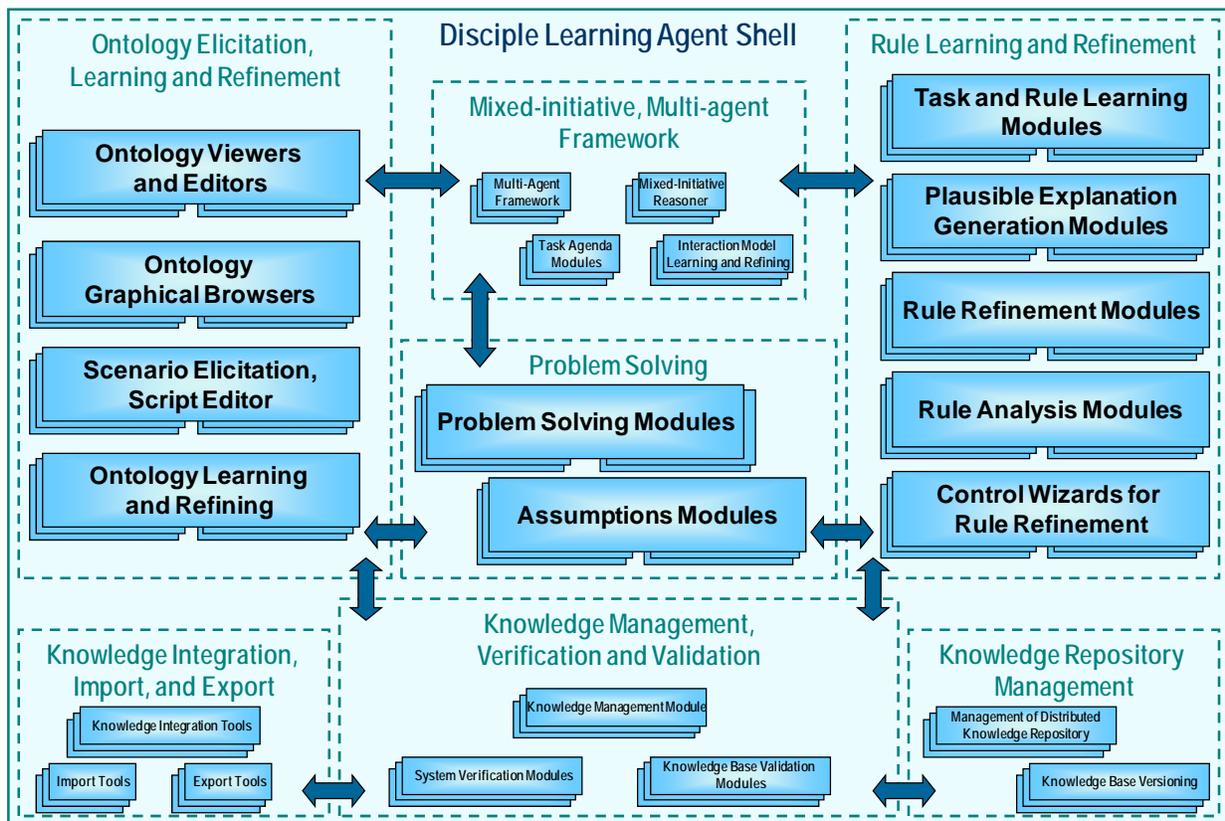


Figure 1: Overall architecture of the Disciple 2008 learning agent shell

The Disciple shell has a multi-agent architecture where its modules are implemented as agents that communicate with one-another and with the users within a mixed-initiative framework. As shown in Figure 1, there are many modules that support the many phases necessary to develop and maintain knowledge-based intelligent assistants. Some of the modules are used directly by subject matter experts, while others are used by knowledge engineers.

#### **4. Development of a Disciple Agent for a Specific Military Application**

In essence, a Disciple agent for a specific military application is developed by appropriately customizing the Disciple Shell, as illustrated in Figure 2 for the case of center of gravity analysis.

There are two main customization processes. One is the development of the domain knowledge base which consists of two main phases: 1) the development of an initial object ontology and 2) the training of the Disciple agent shell in the learning of reasoning rules and refinement of the ontology. The other process is the development of customized modules for the application domain. For example, in the case of Disciple-COG, the users needed a report generation capability that allows Disciple-COG to generate the solution of an analysis in the form of a written report (see section 9). However, all these specialized modules, once developed, become part of the general Disciple shell and can be used in other applications.

The customization will also involve the definition of different types of users (e.g. developer, knowledge engineer, subject matter expert, end-user, or student) with rights to access only certain modules of the Disciple agent.

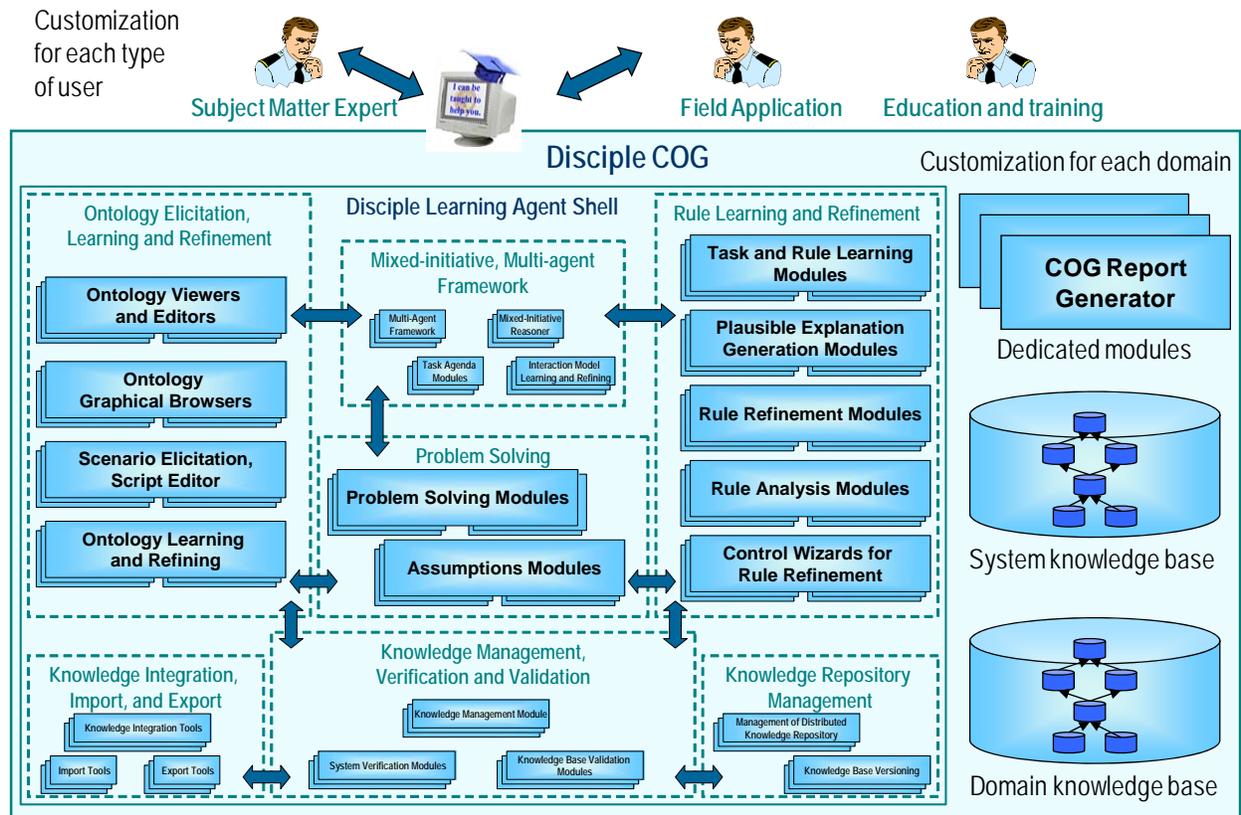
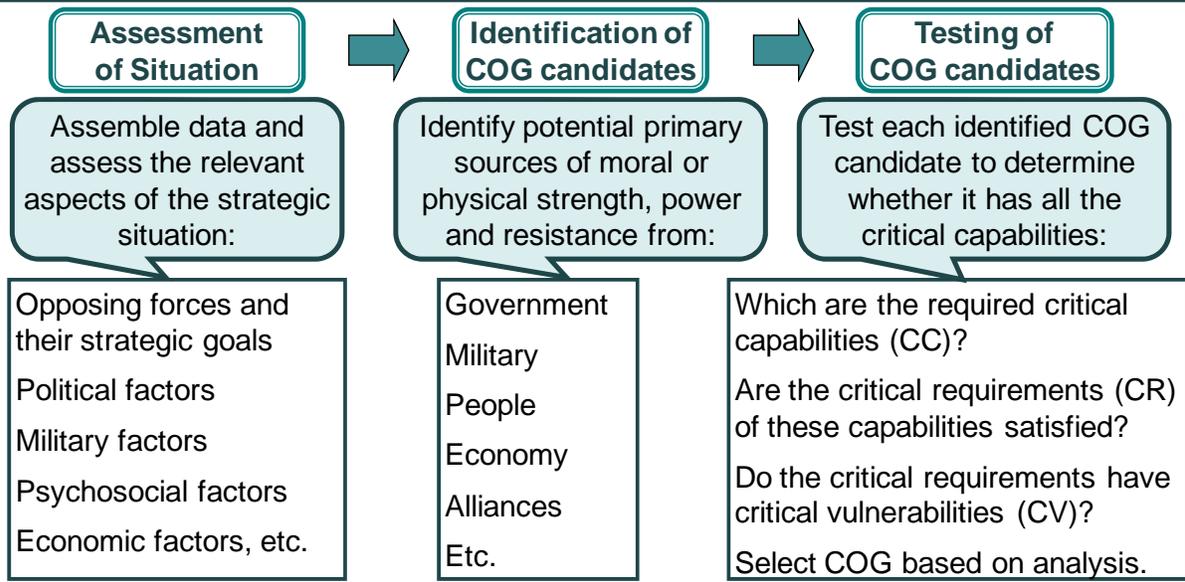


Figure 2: Customization of the Disciple shell into Disciple-COG

Before the Disciple shell can be trained by subject matter experts, a knowledge engineer has to work with a subject matter expert to define a structure for the problem solving process in the problem-reduction / solution-synthesis reasoning framework. Figure 3 presents the overall computational approach to center of gravity analysis, developed with Prof. Comello from the US Army War College and implemented in Disciple-COG.

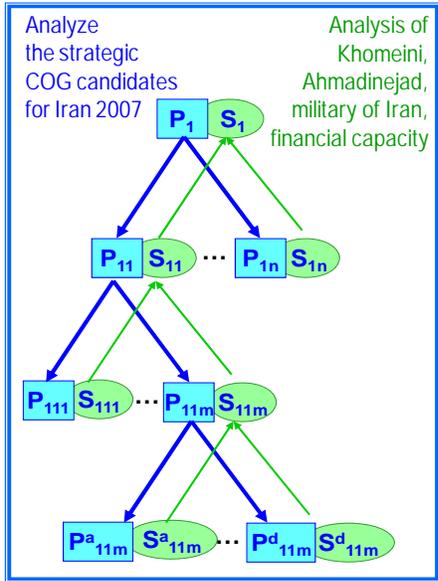
Figure 4 presents the overall formalization of the center of gravity analysis problem as task reduction and solution synthesis (see also Tecuci et al., 2008).

**Given:** A strategic situation (e.g. the invasion of Iraq by the US-led coalition in 2003).  
**Determine:** The strategic centers of gravity of the opposing forces and their critical vulnerabilities.



**Figure 3: Computational approach to center of gravity analysis**

- 1) Analyzing the strategic COG of a force is reduced to the problems of analyzing the COG candidates corresponding to its main elements of power (government, people, economy, military, etc.).
- 2) Analyzing a COG candidate is reduced to the problems of analyzing its Critical Capabilities (CC).
- 3) Analyzing a Critical Capability is reduced to analyzing its Critical Requirements (CR).
- 4) Analyzing a Critical Requirement is reduced to determining whether it has any Critical Vulnerability (CR).



**Figure 4: Problem reduction paradigm for center of gravity analysis problem in Figure 3**

Figure 5 defines the problem of critical vulnerability analysis of a friendly center of gravity by an enemy force (top), shows a systematic way of solving it (middle), and provides an example of a specific problem and its solution (bottom). This formalization was done in collaboration with Prof. Joseph Strange from the Marine Corps War College and will be used to illustrate some knowledge base maintenance operations in Section 6.

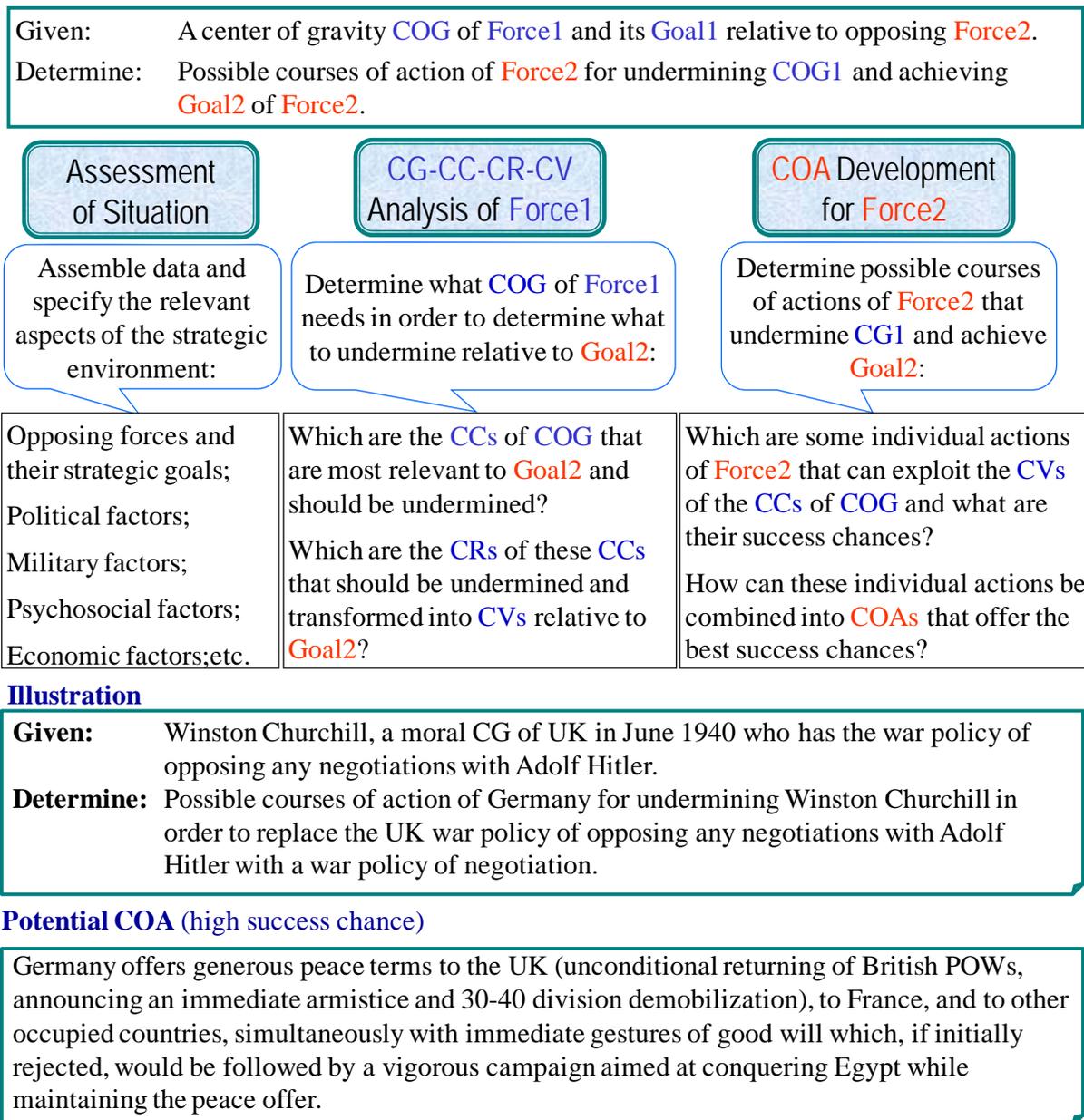
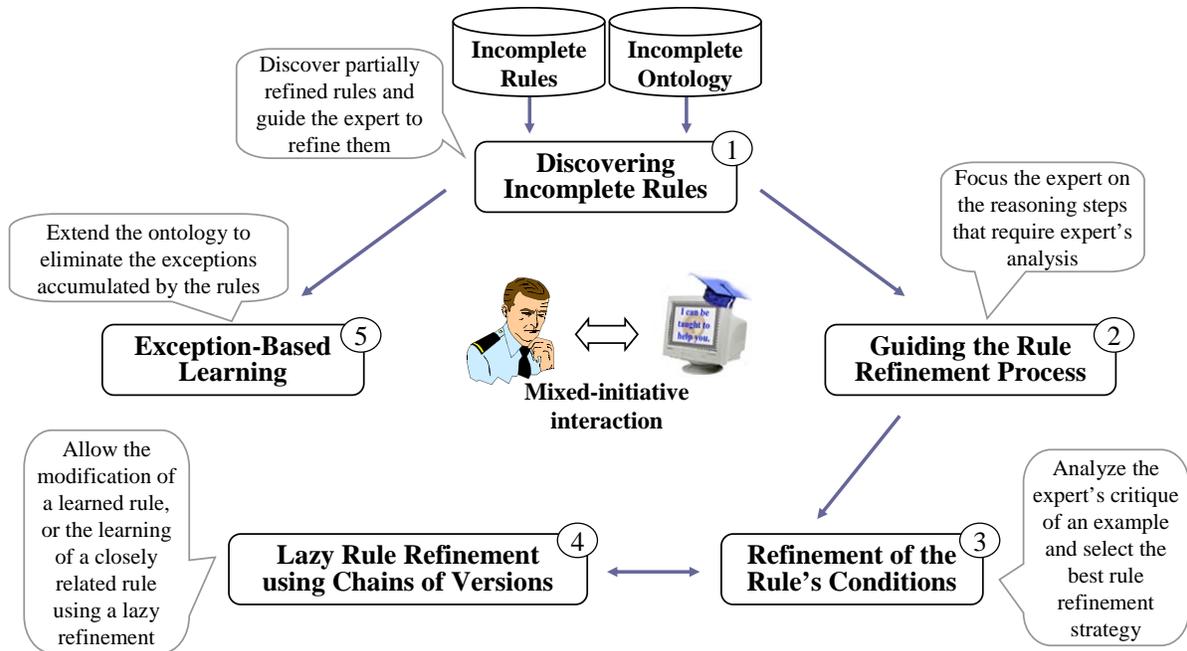


Figure 5: Critical vulnerability analysis of a friendly COG by an enemy force

## 5. Continuous Knowledge Acquisition and Maintenance

We have developed an integrated set of methods for continuous knowledge acquisition and maintenance, and we have implemented them into the Disciple Shell 2008. The most important of these methods are shown in Figure 6 (Boicu C. et al., 2007; Tecuci et al., 2007d; Boicu C., 2006; Tecuci et al., 2005a).



**Figure 6: Integrated methods for continuous knowledge acquisition and maintenance.**

Because the rules learned by the agent from the subject matter experts are generally incomplete, one has developed a set of methods to continuously analyze the rules, in order to discover whether they need further refinement (see #1 in Figure 6). These methods alleviate the experts' tendency of omitting implicit details in human communication, and guide them to provide more explanations of the rules' examples in order to refine these rules (Boicu C. et al., 2005c).

When a Disciple agent solves some problem, it shows the entire reasoning tree to the expert. Because this tree is generally very large for a complex application domain (such as center of gravity analysis), methods have been developed to guide the expert in understanding, navigating and refining a complex reasoning tree, by focusing the expert on those steps of the agent's reasoning process that require expert's analysis (see #2 in Figure 6). These are steps generated by using highly incomplete rules, or the plausible upper bound conditions of partially incomplete rules, which need to be refined (Boicu C., 2006).

After the expert selects a problem reduction step to analyze, the agent interacts with him or her to critique the step and then updates the rule accordingly. Next the agent guides the expert to analyze the other reduction steps of the current reasoning tree that were generated by the same rule, leading to the further refinement of the rule's applicability conditions (see #3 in Figure 6; Boicu C., 2006).

Often the refinement of a rule requires the verification of the rule's previous examples which may not be natural or even possible to do at that time. We have developed a lazy rule refinement method that allows the modification of a learned rule, or the learning of a closely related rule, without requiring the expert to perform an analysis of the rule's representative examples at the time of the modification. Instead, this analysis is postponed until the agent applies the rule in problem solving (see #4 in Figure 6; Boicu C. et al, 2006).

The incompleteness of the representation language results in the learning of rules with exceptions. Generally, these exceptions are due to missing or partially represented ontological knowledge. We have developed a method that performs an analysis of the rules' exceptions and suggests extensions to the agent's ontology, improving the rules by eliminating their exceptions (see #5 in Figure 6; Boicu C., 2006).

One of the most powerful capabilities of Disciple is that it learns complex reasoning rules through an easy and natural interaction with a subject matter expert. To briefly illustrate this capability, consider the problem of critical vulnerability analysis from Figure 5 and its illustration with Winston Churchill, a moral center of Gravity of the UK who has the war policy of opposing any negotiation with Adolf Hitler. A German analyst would like to determine possible courses of action of Germany for undermining Winston Churchill in order to replace the UK's war policy of opposing any negotiations with Adolf Hitler with a war policy of negotiation.

A critical capability of Winston Churchill, which is very relevant to this goal and needs to be undermined, is that of maintaining the support of the population of UK for his war policy. This, in turn, leads to the need to undermine the critical requirements of this capability. One such critical requirement is that the population of UK should believe that Adolf Hitler has imposed dictatorial and humiliating peace terms to the occupied countries. One strategy for Germany to undermine this critical requirement is to offer more generous peace terms to some of the occupied countries, contingent upon a peace treaty between UK and Germany.

Figure 7 illustrates how the analyst teaches Disciple how to apply this strategy. First the expert considers Norway, which is one of the occupied countries, to which Germany is offering complete withdrawal from occupied territories, but contingent on a peace treaty between Germany and the UK. This is the example from the upper left side of Figure 7. From this example Disciple learns a problem reduction rule with the main condition shown in the right hand side of Figure 7. This rule is immediately used by Disciple to suggest to the analyst that Germany may offer complete withdrawal from occupied territories also to Poland, Holland, Belgium, Czechoslovakia and Luxemburg. But the German analyst indicates that Germany cannot offer complete withdrawal from Poland. Disciple proposes as a plausible explanation (and the expert accepts) the fact that Poland is between Germany and Russia which is a future likely target for Germany. As a result, the reasoning rule from the right hand side of Figure 7 is refined by adding

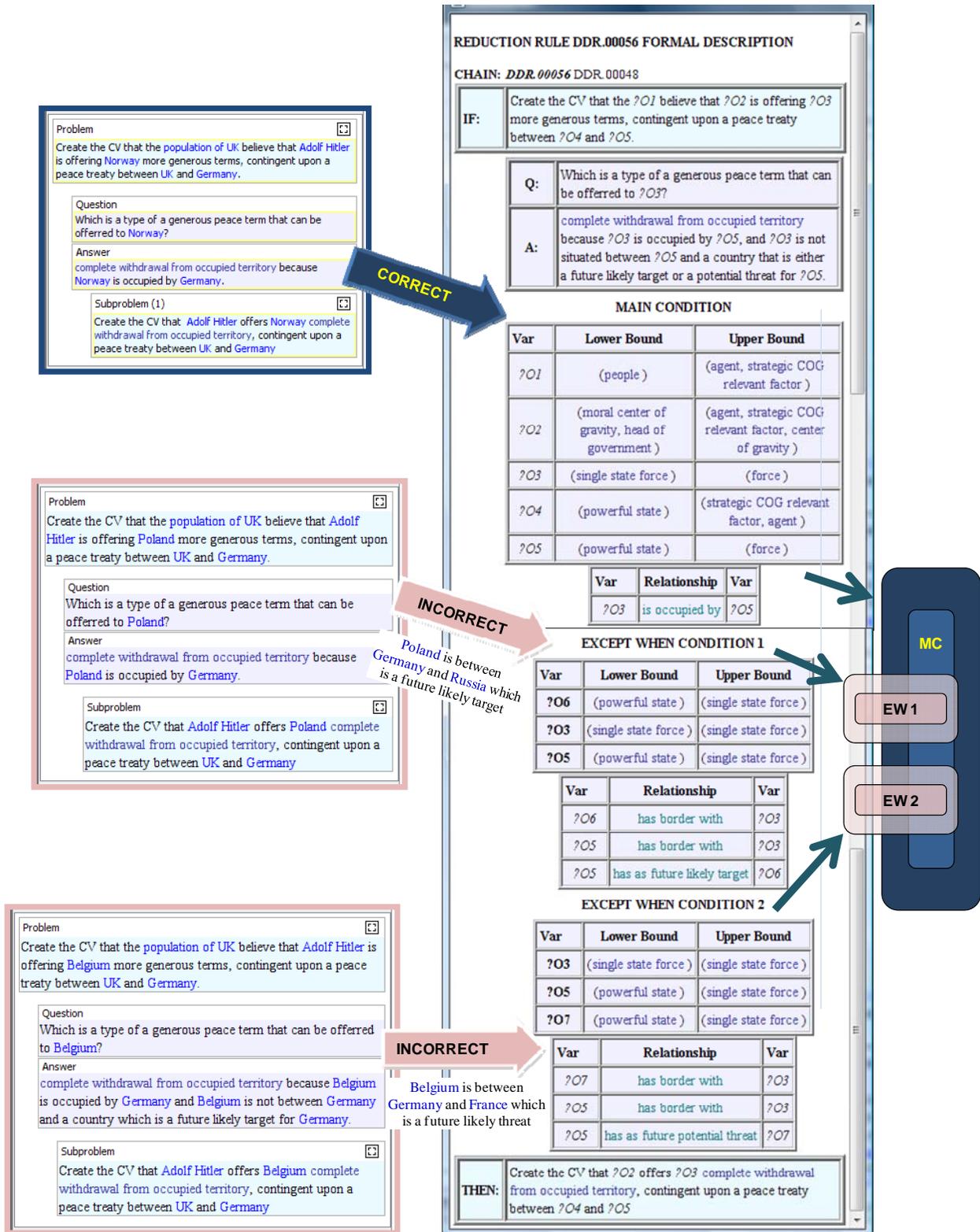


Figure 7: Rule learning and refinement from three examples

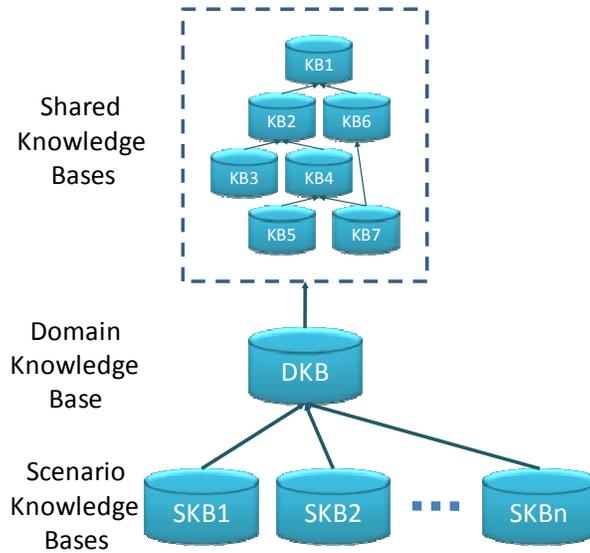
“EXCEPT WHEN CONDITION 1” to it. Also, Disciple no longer suggests to offer Poland and Czechoslovakia (another occupied country situated between Germany and Russia) complete withdrawal from the occupied territory.

Further the analyst also indicates that Belgium cannot be offered complete withdrawal from its occupied territory and provides as explanation the fact that Belgium is between Germany and France, which is a future likely threat to Germany. This causes Disciple both to learn a new feature (“has as future potential threat”) and to refine the reasoning rule with “EXCEPT WHEN CONDITION 2.” Thus, from only three examples, Disciple has learned a complex reduction rule that, in essence, states that: “As a goodwill gesture toward an opponent, the leader of a country can offer complete withdrawal from an occupied country, contingent on a peace treaty with the opponent, except when the occupied country is bordering a future likely target, and except when the occupied country is bordering a future potential threat.” The three conditions of this rule are expressed as plausible version spaces that will be further refined by Disciple.

## **6. Distributed Knowledge Acquisition and Integration**

To support distributed knowledge acquisition, the knowledge base of a Disciple agent consist of a hierarchy of knowledge bases structured into a shared knowledge base part and a local part, as indicated in Figure 8.

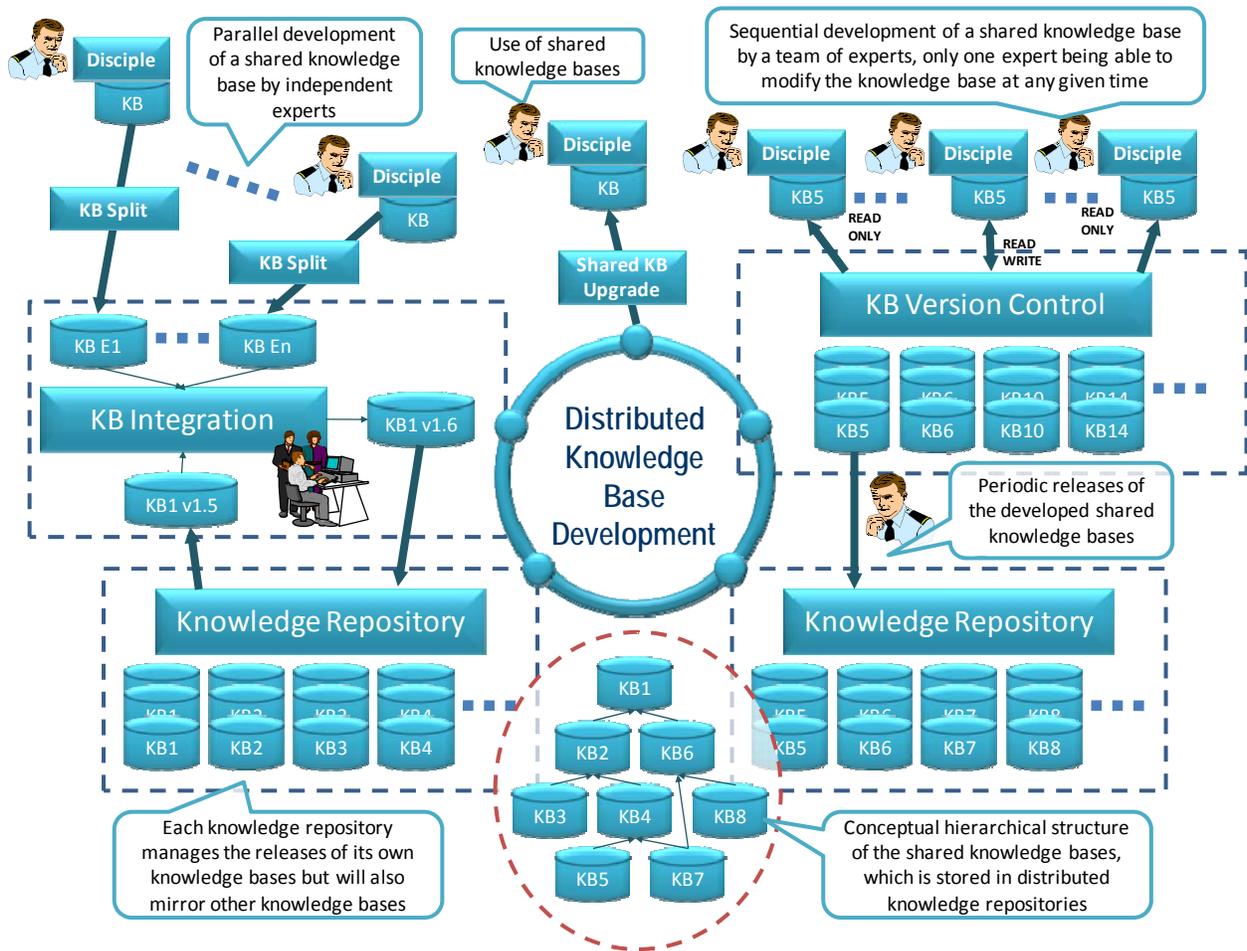
The shared knowledge base part consists of a hierarchy of knowledge bases that contain the shared object ontology, problems, solutions and rules. This shared knowledge is used both to enable the communication and collaboration between the experts, and to facilitate the development of the local knowledge bases of different Disciple agents. The local knowledge bases will inherit from some of the shared knowledge bases.



**Figure 8: The organization of the knowledge base of a Disciple agent**

The local knowledge base part is divided into a domain knowledge base, several scenario knowledge bases and, if necessary, several state knowledge bases. The domain knowledge base represents general knowledge of a domain (such as center of gravity analysis or intelligence analysis) which consists of generic objects, features, facts, problems, and reasoning rules. The domain knowledge base also inherits general knowledge (e.g. a hierarchy of units of measure) from the shared knowledge bases. Its knowledge is inherited by several scenario knowledge bases that represent specific situations (e.g. the current situation in Iraq, to perform a center of gravity analysis of it). A scenario represents instances and facts describing a situation, as well as specific problems to be solved in that situation. Under a scenario knowledge base there may be several state knowledge bases which are useful for state-based reasoning, such as planning.

Figure 9 represents the process of distributed knowledge bases development and integration that is used to develop shared knowledge bases. The Disciple shell supports two complementary strategies for the development of a shared knowledge base: parallel development by independent experts and sequential development by a team of experts.



**Figure 9: Distributed knowledge bases development and integration**

The shared knowledge bases are conceptually in a hierarchical structure. However this structure is stored in a distributed way on multiple knowledge repository servers which manage the release of new versions, the check out of specific versions, and the access rights for registered users.

In the parallel development, independent experts will work separately to develop personal knowledge bases that include a copy of the same shared knowledge base. In this process each expert will also develop his or her copy of the shared knowledge base. Periodically, they may decide to update the shared knowledge base with the new elements from their updated copy. These elements are extracted by using the knowledge base split module of Disciple. The

elements extracted from all the developed copies of the shared knowledge base and the previous version of the shared knowledge base will be integrated into a new version of the shared knowledge base by a coordinating expert and a knowledge engineer who will use the KB integration module. This new version is then used by the independent experts to upgrade their knowledge bases.

In the sequential development (which is supported by a dedicated knowledge base version control system) only one expert has write access and can modify the shared knowledge base at any given moment, the others having read-only access. When the new version of the shared knowledge base is released, it can be checked-out by another expert, who first needs to upgrade his or her local copy of the shared knowledge base to its newly released version. The upgrading process is based on a knowledge base evolution log maintained by Disciple.

## **7. Knowledge Acquisition and Validation**

Two strategies for validation have been developed. The first is to integrate knowledge validation methods in the knowledge acquisition and maintenance processes. The second is to perform independent validation of the knowledge base.

An important feature of the learning methods of Disciple is that they are tolerant to incomplete and even partial incorrect knowledge. However, identifying and correcting such situations early may significantly improve the knowledge acquisition rates and the performance of the system. Therefore, methods for checking the logical consistency of the individual knowledge base elements have been integrated with all the Disciple knowledge management methods. In particular, after each modification of the knowledge base, all the potentially affected knowledge elements are checked for consistency. If a logical inconsistency is found (e.g. trying

to delete a concept that is used in a reasoning rule), it is either automatically resolved or the modification is reversed and the user is presented with the reasons that prevented that operation.

In the Disciple approach, the knowledge-based agents perform continuous knowledge validation and maintenance. For example, during the modeling phase, the Disciple modeling assistant supports the experts in expressing their reasoning processes in a way that is consistent with the previously formalized reasoning processes (Boicu M. et al, 2005a)). After a rule is learned, the Rule Analysis method (Boicu C., 2005c) is automatically invoked to check the completeness of the rule's explanations and guide the expert in solving the discovered problems. Also, the Analyze Subtree Wizard focuses the experts on the reasoning steps that require their analysis because they have been generated with a lower level of confidence by the partially learned rules (Boicu C., 2006).

To perform an independent validation of the knowledge base, Disciple provides a validation method for the concept taxonomy and one for rules. The logical validation of the concept taxonomy checks that it is well-formed and consistent. The semantic validation requires a team of experts to answer randomly generated questions about the taxonomy. The answers are analyzed and integrated to obtain a measure of confidence in the tested elements. The rule validation method checks their logical completeness and may identify problems with no reasoning rules to solve them.

## **8. Disciple Methodology for Distributed Knowledge Acquisition, Integration, Validation and Maintenance**

The following is an overview of the main steps of the Disciple methodology.

### ***Formalization of the application domain in the problem-reduction paradigm***

A knowledge engineer and a subject matter expert will consider typical problems to be solved by the envisioned knowledge-based agent and will develop a natural divide-and-conquer approach to solving those problems, in the problem-reduction/solution-synthesis paradigm, as was discussed in Section 4.

### ***Partition of the application domain***

A knowledge engineer and a group of subject matter experts will partition the application domain, based on the areas of expertise of the subject matter experts. The knowledge bases for these parts will be developed in parallel and integrated, by groups or individual subject matter experts, with assistance from the knowledge engineer, as discussed in the following.

### ***Distributed knowledge base development***

Each individual or group of experts that develops the knowledge base for a given domain part will perform two types of tasks:

- Local development of the knowledge base specific to the assigned domain part.
- Participation in the joint development of the shared knowledge bases that include the common ontological terms, problem types and solution types.

The development of the hierarchy of shared knowledge base is performed either through sequential development supported by the Disciple version control system, or through parallel development and integration supported by the integration modules, as described in Section 6 and summarized below.

### ***Local development of individual knowledge bases***

Each individual knowledge base is developed with the tools of the Disciple shell (see Figure 2) and involves:

- The development of an initial object ontology (by importing knowledge from previously developed repositories and by using the ontology development tools of Disciple).
- The teaching of a Disciple agent by the subject matter expert, based on the modeling, problem solving and learning modules of Disciple. This will result in the learning and refinement of reasoning rules and in the further development of the ontology.

### ***Sequential development of the shared knowledge bases***

This development is supported by the Disciple KB Version Control Module which allows an expert to check out the current version of a shared knowledge base, modify and extend it, and then commit the changes back as a new version of that knowledge base. During this modification the knowledge base is locked, allowing others to download and use it but not to further develop it. The main idea is that only one expert can modify the knowledge base at any given moment. This may be difficult in the case of a large number of experts. However, it is feasible in the case where a small number of experts collaborate in the development of the same domain knowledge base or shared knowledge base.

### ***Parallel development of shared knowledge bases***

In this case each expert starts with the same shared knowledge base and develops it independently of the others. Periodically a coordinating expert and a knowledge engineer integrate the new knowledge base elements into an updated version of the shared knowledge base.

### ***Knowledge base upgrading***

After a shared knowledge base was modified (using either of the above two methods) the individual experts will need to upgrade (asynchronously) their knowledge base to the new release, by using the Shared KB Upgrade Module. This is an automatic operation, but if

inconsistencies are detected, they are signaled to be resolved manually by using the KB development tools of Disciple.

### *Continuous validation and maintenance*

During this development process both the shared and the individual knowledge bases are continuously validated and maintained, as discussed in Sections 5 and 7.

## **9. Transition of Disciple-COG**

As indicated above, Disciple-COG is an intelligent agent developed with the Disciple shell. Disciple-COG assists a military leader to analyze a strategic situation to determine the strategic center of gravity candidates of the opposing forces.

Disciple-COG has been trained to perform center of gravity analysis based on the analyses of specific historical situations by military experts. As a result, Disciple-COG has learned general analysis strategies that allow it to analyze new situations. Moreover, the resulting analysis is similar to the analysis that would have been performed by the training experts. This makes Disciple-COG exceptionally useful in the education and training of military personnel who, by using it, can learn to follow a systematic approach to center of gravity analysis.

Successive versions of Disciple-COG have been used successfully in courses at the US Army War College and the US Air War College to describe and analyze historic situations (e.g. World War II in Europe in 1943), current situations (e.g. Iraq) and even future hypothetical situations (e.g. Iran 2016).

The following is a brief description of the use of Disciple-COG. A detailed description is provided in (Tecuci et al. 2008; See Appendix A2).

First Disciple-COG guides the user to identify, assess and describe the aspects of the strategic situation that are relevant to its center of gravity analysis. An example of such a situation is the Insurgence conflict between the Lord's Resistance Army and the Government of Uganda since 1986 (situation analyzed by an Air War College international fellow from Uganda).

The user-agent interaction is very easy and natural for the user, taking place as illustrated in Figure 10. The left part of the window is a table of contents whose elements indicate various important aspects of the situation. When the user selects one such aspect, Disciple-COG asks specific questions intended to acquire a description and/or assessment of that aspect, or to update a previously specified description. The user's answers lead to the generation of new items in the left hand side of the window, and trigger new questions from the agent.

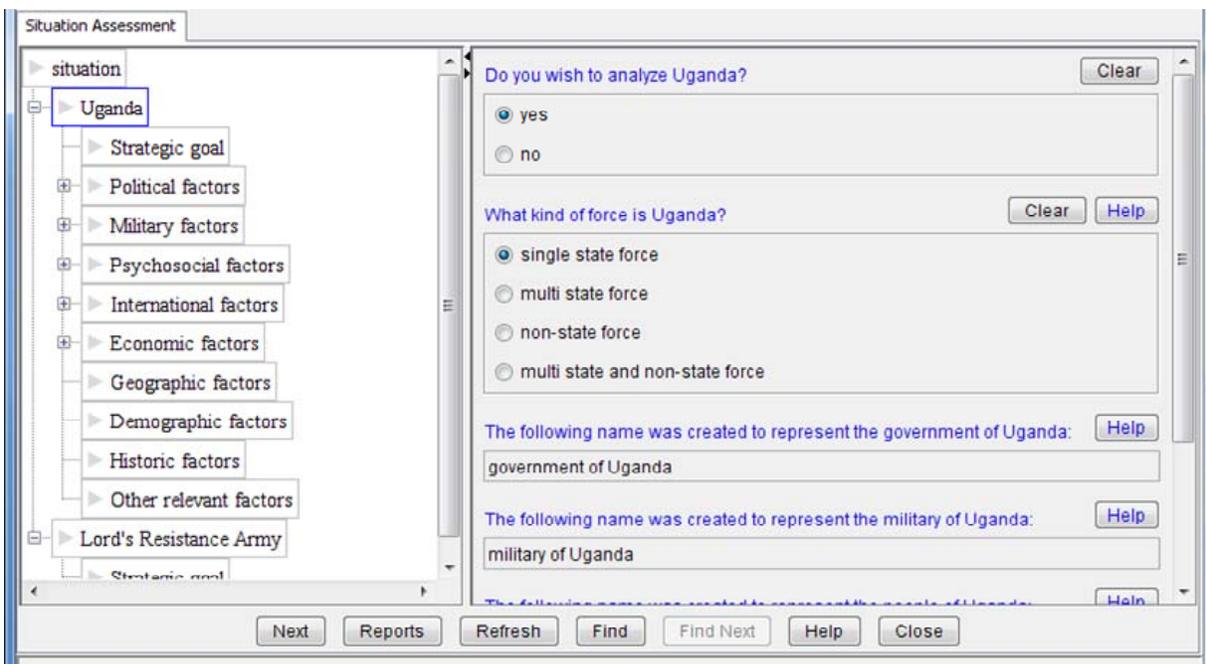


Figure 10: Interface of the situation assessment module

For instance, Disciple-COG asks for the opposing forces of the current situation, the user names them Uganda and Lord's Resistance Army, and Disciple-COG includes them into the table of contents. Then, when the user clicks on one of them (e.g. Uganda), Disciple-COG asks for their characteristics, as indicated in the right hand side of Figure 10. The user characterized Uganda as a single state force which caused Disciple-COG to extend the table of contents with the relevant aspects of a single state force (i.e. strategic goals, political factors, military factors, etc.). The user can now click on any such aspect and will be asked specific questions by Disciple-COG.

The user is not required to answer all the questions and Disciple-COG can be asked, at any time, to identify and test the strategic center of gravity candidates for the current description of the situation. Figure 11 shows the interface of the mixed-initiative reasoner that performs the analysis. The left hand side shows a classification of the various center of gravity candidates identified by Disciple-COG (e.g. will of the people of Uganda, President Yoweri Museveni, etc.) and their components (e.g. their critical capabilities). When the user selects a center of gravity candidate in the left hand-side (e.g. "Candidate: President Yoweri Museveni" in Figure 11), the right hand side of the interface shows the result of its analysis: "President Yoweri Museveni is a strategic COG candidate that can be eliminated because President Yoweri Museveni does not have all the necessary critical capabilities (e.g. be irreplaceable)."

Under this result are the results of the analyses for the individual critical capabilities that appear under President Yoweri Museveni in the left hand side of the screen.

Thus, Disciple-COG guides the user through a detailed analysis of a situation that identifies a set of center of gravity candidates, a set of critical capabilities for each candidate and a set of critical requirements for each critical capability. Also, for the identified critical requirements, Disciple-COG points the user to assess their critical vulnerabilities (if any) and to justify them, as illustrated in Figure 12.

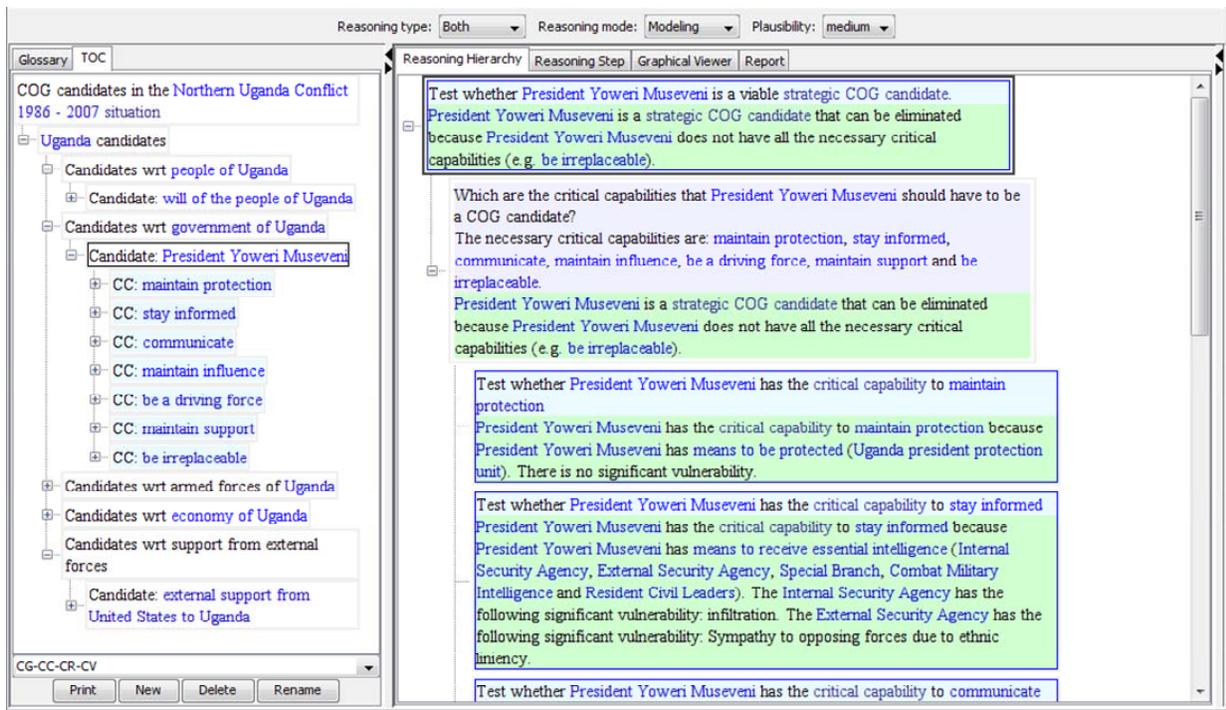


Figure 11: Interface of the mixed-initiative reasoner

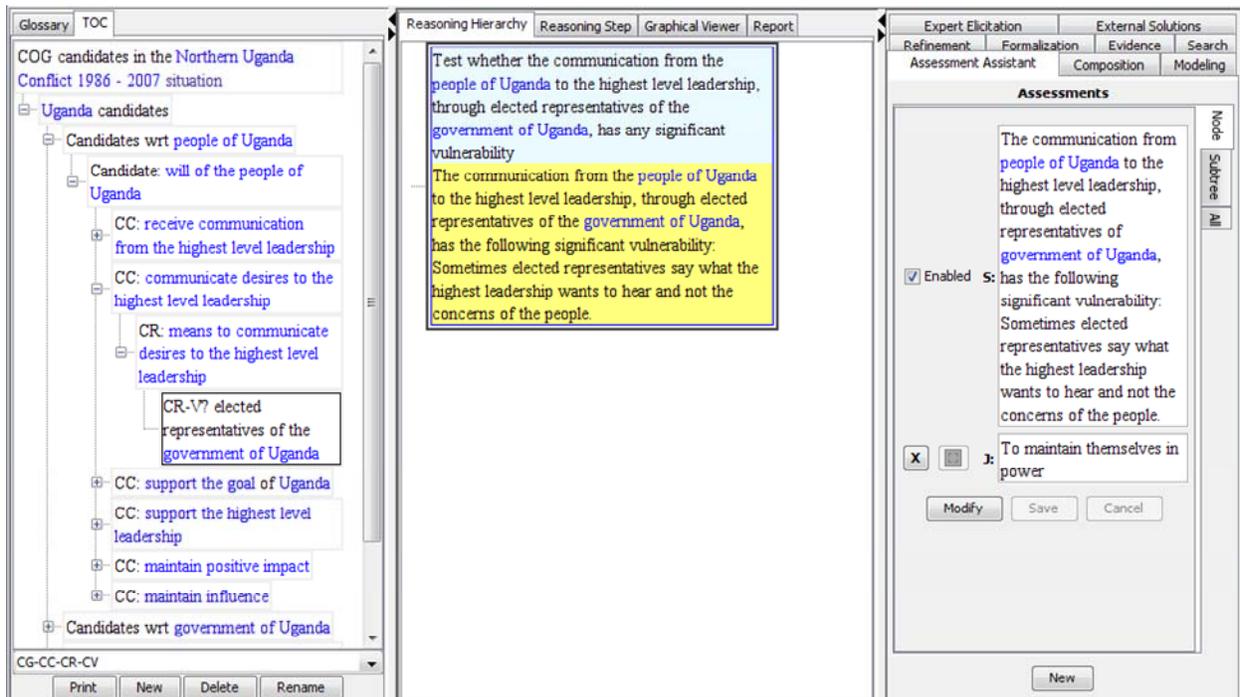


Figure 12: Vulnerability assessment interface

At the end of the analysis, Disciple-COG generates a draft analysis report, a fragment of which is shown in Figure 13. The first part of this report contains a description of the strategic situation, which is generated from the information provided and assessed by the user, as illustrated in Figure 10. The second part of the report includes all the center of gravity candidates identified by Disciple, together with their analyses, as discussed above. The user may now finalize this report by examining the analysis of each center of gravity candidate, completing, correcting, or even rejecting it and providing a different analysis.

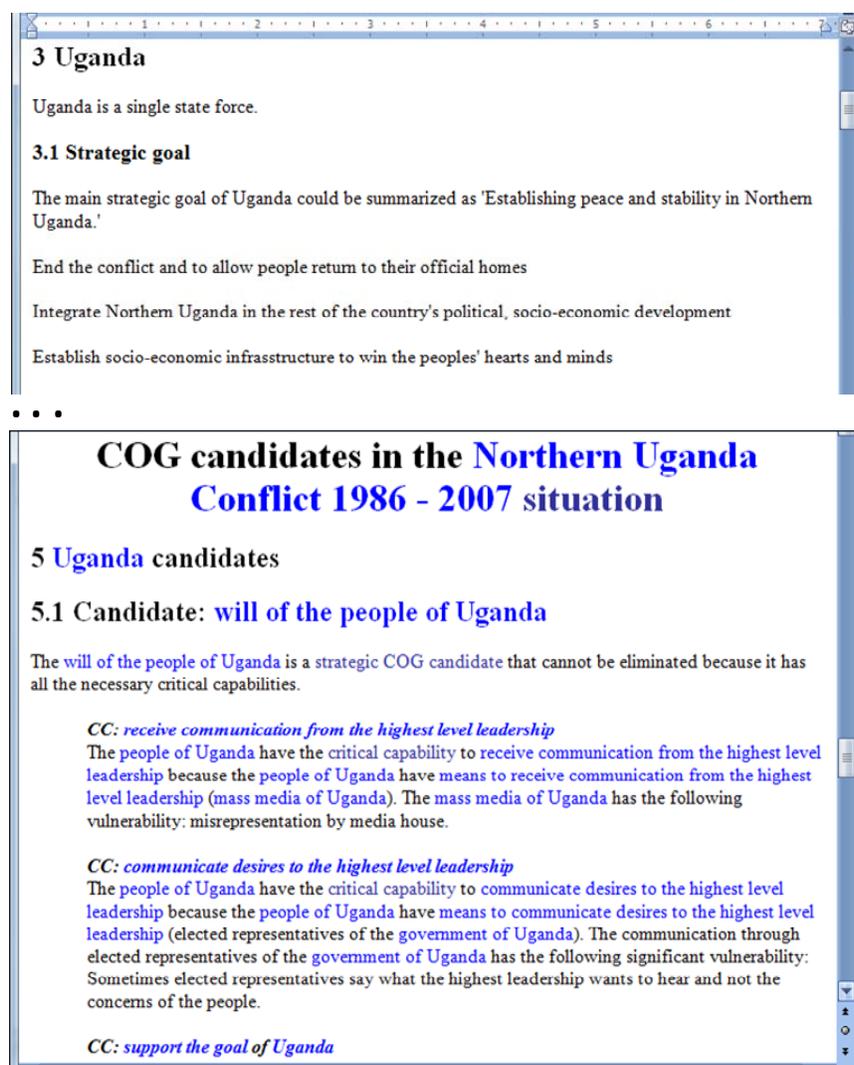


Figure 13: Fragment of an automatically generated report

Figure 14 shows some global experimental results on the use of Disciple-COG in the Spring 2007 session of the Air War College enrichment elective entitled “Center of Gravity Analysis.” In this course, ten military experts (at the rank of lieutenant colonel and above) experimented with personal copies of Disciple which guided each them to specify a scenario of interest, captured critical vulnerabilities, and performed a strategic center of gravity analysis.

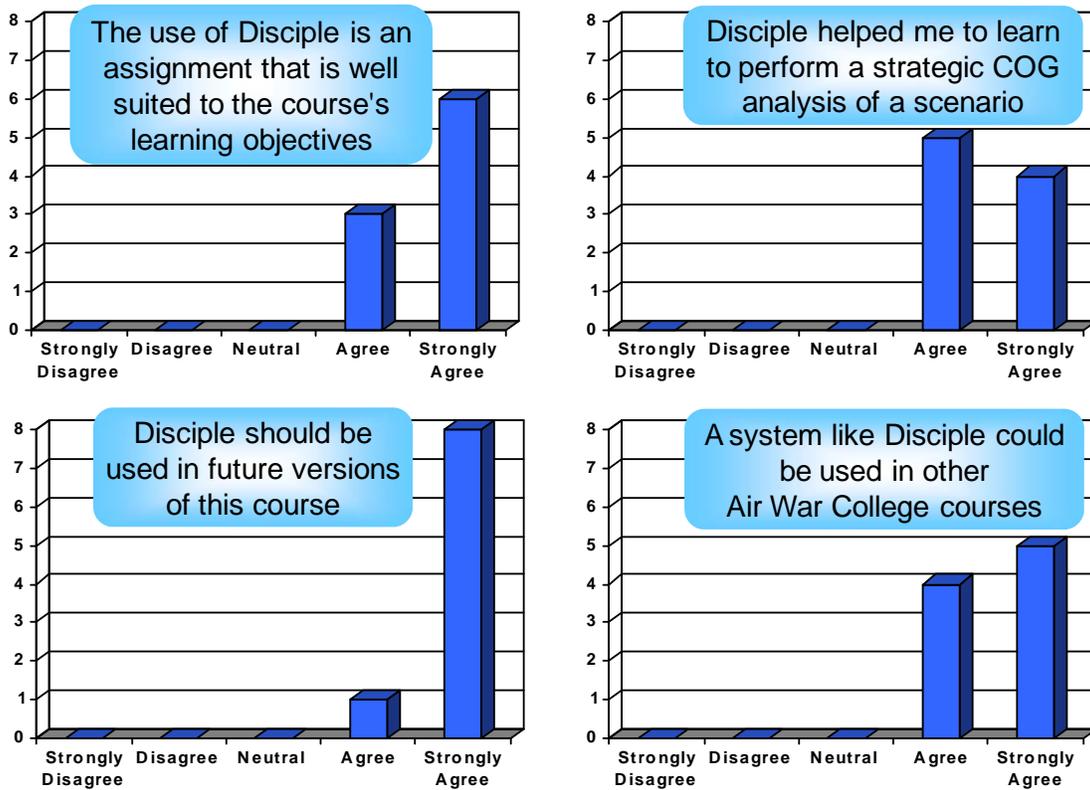


Figure 14: Subjective evaluation of Disciple-COG

## 10. Other Applications of Disciple Agents

Although this project has primarily developed a knowledge base of center of gravity analysis, the Disciple shell is applicable to a wide range of military and civilian domains. For example,

knowledge based agents have been developed for intelligence analysis, for guiding a student to select a PhD advisor, and for analyzing the financial transactions of a financial institution.

## **11. Future Research Directions**

Experience with the process of acquiring, validating, and maintaining Disciple-based agents has inspired several breakthrough developments that would significantly facilitate the use of this technology by subject matter experts, and would allow the capture of more complex problem solving knowledge. These breakthrough developments include:

- Abstraction-based knowledge capture that will guide the experts to make explicit their reasoning processes by reusing reasoning patterns at higher levels of abstraction.
- Mixed-initiative learning of solution synthesis rules directly from the subject matter experts.
- Validation-driven knowledge refinement that will allow the military experts to easily improve the reasoning of the system and its knowledge base during system's verification and validation process.

The presented knowledge base integration methods provide basic capabilities that need to be further developed to facilitate this complex process. Moreover, to further facilitate collaborative problem solving and knowledge capture from multiple subject matter experts, it would be very useful to develop web-based interfaces for selected tools and offer knowledge capture and reasoning capabilities as web services in a service-oriented architecture.

## Appendices

### A1. Publications Describing the Results of this Project

Tecuci G., Boicu M., and Comello J., *Agent-Assisted Center of Gravity Analysis*, GMU Press, 2008a (to appear).

Le V., Tecuci G., Boicu M., Agent Shell for the Development of Tutoring Systems for Expert Problem Solving Knowledge. In Proceedings of the 9<sup>th</sup> International Conference on Intelligent Tutoring Systems (ITS2008), Springer Verlag, 2008.

Tecuci G., Boicu M., Marcu D., Barbulescu M., Boicu C., Le V., Hajduk T., Teaching Virtual Experts for Multi-Domain Collaborative Planning, *Journal of Software*, Volume 3, Number 3, pp. 38-59, March 2008b.

Le V., Abstraction of Reasoning for Problem Solving and Tutoring Assistants. Ph.D. Dissertation in Information Technology. Learning Agents Center, Volgenau School of IT&E, George Mason University, Spring 2008.

Boicu C., Tecuci G., Boicu M., Learning Complex Problem Solving Expertise from Failures, in *Proceedings of the Sixth International Conference on Machine Learning and Applications (ICMLA'07)*, 6 pages, Cincinnati, Ohio, December 13-15, 2007.

Tecuci G., Marcu D., Boicu M., Le V., Mixed-Initiative Assumption-Based Reasoning for Complex Decision-Making, *Studies in Informatics and Control*, Volume 16, Number 4, December 2007a.

Tecuci G., Boicu M., Cox M.T., (Guest Editors) *AI Magazine*, Special Issue on Mixed-Initiative Assistants, Volume 28, Number 2, Summer 2007b.

Tecuci G., Boicu M., Cox M.T., Seven Aspects of Mixed-Initiative Reasoning: An Introduction to the Special Issue on Mixed-Initiative Assistants, *AI Magazine*, Volume 28, Number 2, pp. 11-18, Summer 2007c.

Tecuci G., Boicu M., Marcu D., Boicu C., Barbulescu M., Ayers C., Cammons D., Cognitive Assistants for Analysts, *Journal of Intelligence Community Research and Development (JICRD)*, 2007. Also published in John Auger, William Wimbish (eds.), *Proteus Futures Digest: A Compilation of Selected Works Derived from the 2006 Proteus Workshop*, pp. 303-329, Joint publication of the National Intelligence University, Office of the Director of National Intelligence, and US Army War College Center for Strategic Leadership, 2007d.

Tecuci G., Boicu M., Hajduk T., Marcu D., Barbulescu M., Boicu C., Le V., A Tool for Training and Assistance in Emergency Response Planning, in *Proceedings of the Hawaii International Conference on System Sciences*, 10 pages, HICSS-40, Hawaii, January 3-6, 2007e.

Boicu C., Tecuci G., Boicu M., Lazy Rule Refinement by Knowledge-Based Agents, in *Proceedings of the Fifth International Conference on Machine Learning and Applications (ICMLA'06)*, Orlando, Florida, USA, 14-16 December 2006.

Boicu C., An Integrated Approach to Rule Refinement for Instructable Knowledge-Based Agents. PhD Thesis in Computer Science, Learning Agents Center, Volgenau School of Information Technology and Engineering, George Mason University, Fairfax, Virginia, Fall 2006.

Fowler C., Ayers C., Cammons D., Tecuci G., Boicu M., Le V., Marcu D., Boicu C., Barbulescu M., Cognitive Assistants for the Analysts: Experiments in the Classroom, in *Proceedings of the INFORMS Military Applications Society Conference "Homeland Security for the 21st Century,"* Mystic, Connecticut, July 24-26, 2006.

Tecuci G., Boicu M., Ayers C., Cammons D., Cognitive Assistants for Analysts, in *Proceedings of the 4th Annual Analysis & Production's Analysis Conference "The Future of Analysis"* NSA, April 2-4 2006.

Aha D., Tecuci G. (eds.), *Proceedings of the AAAI-2005 Fall Symposium on Mixed-Initiative Intelligent Systems*, November 4-6, 2005, Arlington, Virginia. AAAI Press Technical Report, 145p, Available also online: <http://lac.gmu.edu/MIPSA/default.htm#Proceeding>

Boicu C., Tecuci G., Boicu M., A Mixed-Initiative Approach to Rule Refinement for Knowledge-Based Agents, *Proceedings of the AAAI-05 Fall Symposium on Mixed-Initiative Problem-Solving Assistants*, pp. 1-6, Arlington, Virginia, November 4-6, 2005a.

Boicu M., Tecuci G., Marcu D., Mixed-Initiative Assistant for Modeling Expert's Reasoning, *Proceedings of the AAAI-05 Fall Symposium on Mixed-Initiative Problem-Solving Assistants*, pp.7-13, Arlington, Virginia, November 4-6, 2005a.

Tecuci G., Boicu M., Boicu C., Marcu D., Stanescu B., Barbulescu M., The Disciple-RKF Learning and Reasoning Agent, *Computational Intelligence*, Vol.21, No.4, pp 462-479, 2005a.

Boicu M., Tecuci G., Ayers C., Marcu D., Boicu C., Barbulescu M., Stanescu B., Wagner W., Le V., Apostolova D., Ciubotariu A., A Learning and Reasoning System for Intelligence Analysis, *Proceedings of the Twentieth National Conference on Artificial Intelligence*, AAAI-05, Pittsburgh, Pennsylvania, USA, July 9-13 2005b.

Boicu C., Tecuci G., Boicu M., Rule Refinement by Domain Experts in Complex Knowledge Bases, *Proceedings of the Twentieth National Conference on Artificial Intelligence*, AAAI-05, Pittsburgh, Pennsylvania, USA, July 9-13 2005b.

Boicu C., Tecuci G., Boicu M., Improving Agent Learning through Rule Analysis, *Proceedings of the International Conference on Artificial Intelligence*, ICAI-05, Las Vegas, USA, June 27-30, 2005c.

Boicu M., Tecuci G., Learning Best Concept Approximations from Examples, *International Journal of Computational Intelligence Research*, Vol.1, No.1, pp. 85-97, 2005c.

Tecuci G., Boicu M., Ayers C., Cammons D., Personal Cognitive Assistants for Military Intelligence Analysis: Mixed-Initiative Learning, Tutoring, and Problem Solving, *Proceedings of the First International Conference on Intelligence Analysis*, McLean, VA, 2-6 May, 2005b.

Bowman M., Tecuci G., Boicu M., Comello J., Information Age Warfare – Intelligent Agents in the Classroom and the Strategic Analysis Center, 8 pages, 24th Army Science Conference, November 29-December 2, 2004, Orlando, Florida. This is Army's premier Science and Technology Conference. The theme of the 2004 conference was Transformational Science and Technology for the Current and Future Force.

Tecuci G., Boicu M., Marcu D., Stanescu B., Boicu C., Barbulescu M., "Parallel Knowledge Base Development by Subject Matter Experts" in *Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management*, EKAW 2004, 5-8th October 2004 - Whittlebury Hall, Northamptonshire, UK, Springer-Verlag, 2004a.

Tecuci G., Boicu M., Marcu D., Stanescu B., Boicu C., Barbulescu M., "A Learning Agent Shell for Building Knowledge-Based Agents," in *Proceedings of the Technology Demonstration Session of the 14th International Conference on Knowledge Engineering and Knowledge Management*, EKAW 2004, 5-8th October 2004 - Whittlebury Hall, Northamptonshire, UK, 2004b.

## **A2. Agent-assisted Center of Gravity Analysis Text Book: Summary and Table of Contents**

### **SUMMARY**

This volume describes a systematic approach to strategic center of gravity analysis and a decision-support software agent, called Disciple-COG, which incorporates this approach. Disciple-COG assists a military leader to analyze a strategic situation, such as Operation Enduring Freedom - Afghanistan 2001-2002, to determine the strategic center of gravity candidates of the opposing forces. Disciple-COG is an intelligent agent that has been trained to perform center of gravity analysis based on the analyses of specific historical situations by a military expert. As a result, Disciple-COG has learned general analysis strategies that allow it to analyze new situations. Moreover, the resulting analysis is similar to the analysis that would have been performed by the training expert. This makes Disciple-COG exceptionally useful in the education and training of military personnel who, by using it, can learn to follow a systematic approach to center of gravity analysis. Successive versions of Disciple-COG have been used successfully in courses at the US Army War College and the US Air War College to describe and analyze historic situations (e.g. World War II in Europe in 1943), current situations (e.g. Iraq) and future hypothetical situations. This volume provides both a detailed description of the developed center of gravity analysis approach, and step by step instructions for using Disciple-COG. The accompanying CD includes the Disciple-COG agent and lecture notes supporting its use in courses at senior service colleges.

### **1. INTRODUCTION**

#### **2. COMPUTATIONAL APPROACH TO CENTER OF GRAVITY ANALYSIS USING AGENT TECHNOLOGY**

2.1 CENTER OF GRAVITY

2.2 CENTER OF GRAVITY ANALYSIS

2.3 DISCIPLE-COG: AN AGENT FOR CENTER OF GRAVITY ANALYSIS

#### **3. ASSESSMENT OF A STRATEGIC SITUATION**

3.1 INTRODUCTION

3.2 SAMPLE STRATEGIC SITUATIONS

3.3 OPPOSING FORCES AND THEIR GOALS

3.4 INTERNATIONAL FACTORS

3.5 POLITICAL FACTORS

3.6 MILITARY FACTORS

3.7 PSYCHOSOCIAL FACTORS

3.8 ECONOMIC FACTORS

3.9 GEOGRAPHIC FACTORS

3.10 DEMOGRAPHIC FACTORS

3.11 HISTORIC FACTORS

3.12 OTHER RELEVANT FACTORS

#### **4. TYPICAL STRATEGIC CENTERS OF GRAVITY**

4.1 NATIONAL LEADER

4.2 WILL OF THE PEOPLE

4.3 MILITARY

4.4 INDUSTRIAL CAPACITY

4.5 FINANCIAL CAPACITY

4.6 IDEOLOGY AND ITS PROPONENTS

4.7 EXTERNAL SUPPORT

4.8 WILL OF MULTI-MEMBER FORCE

#### **5 CENTER OF GRAVITY ANALYSIS THROUGH PROBLEM REDUCTION**

5.1 THE PROBLEM REDUCTION PARADIGM OF PROBLEM SOLVING

5.2 IDENTIFICATION OF CENTER OF GRAVITY CANDIDATES

5.3 TESTING OF CENTER OF GRAVITY CANDIDATES

5.4 ASSESSMENT OF CRITICAL VULNERABILITIES

5.5 DISPLAY OF THE ANALYSIS REPORT

#### **6. REPORT GENERATION**

#### **7. LECTURE NOTES: CENTER OF GRAVITY ANALYSIS WITH DISCIPLE-COG**

LECTURE 1. INTELLIGENT AGENT FOR COG ANALYSIS.

HANDS-ON: SITUATION ASSESSMENT.

LECTURE 2. INTELLIGENT AGENTS RESEARCH.

HANDS-ON: SITUATION ASSESSMENT.

LECTURE 3. COG ANALYSIS THROUGH PROBLEM REDUCTION.

HANDS-ON: COG ANALYSIS AND EXPERTISE CAPTURE.

#### **8. DISCIPLE-COG CD**

#### **9. CONCLUSIONS**

#### **ACKNOWLEDGMENTS**

#### **REFERENCES**

#### **APPENDIX: DISCIPLE-COG OPERATION NOTES**

### **A3. Disciple Shell 2008 and Disciple-COG CD-ROM**

The Disciple Shell 2008 and Disciple-COG CD-ROM includes the following:

- The source and executable code of Disciple Shell 2008, including code documentation and user help files for the Disciple modules.
- The customized Disciple-COG agent with sample knowledge bases.
- The book “Agent-assisted Center of Gravity Analysis”.
- Lecture Notes for using Disciple-COG at senior service colleges.
- Papers on the Disciple learning agent technology.
- The quarterly reports submitted during the period of performance of this project.