Intelligent Agent Technology in Command and Control Environment

Edward Dawidowicz[1]
U.S. Army Communications-Electronics Command (CECOM)
CECOM, RDEC, Myer Center
Command and Control Directorate
Fort Monmouth, NJ. 07703
732.427.4122

Abstract

The Intelligent Agent (IA) technology has applications in the following areas of military Command and Control (C2): logistics, combat planning and battle plan execution monitoring.

C2 information extraction should not rely on using simple database queries, since the amount of data available to the commander on the modern battlefield can be overwhelming. In order to make informed decisions; the commander must have immediate access to specific information in real time. The available data therefore, must be parsed in such a way as to extract only the specific information required by the commander. The ever-increasing volume of data in the C2 environment thus requires the use of IAs to extract relevant information for the commander in real time.

The paper describes an application of IA in assisting a decision-maker, i.e. the military commander, by extracting needed information from a large amount of data and triggering an alarm when certain critical conditions are reached. The open architecture proposed here not only allows effective IA implementation but also expansion of future IA applications, as needs demand. This paper is a continuation of earlier work [Ref. 1,2,3].

1.0 Introduction

The increase in dynamics associated with the modern battlefield adds additional levels of complexity to the perception of what constitutes a battlespace. As a rule, even the best battle plans are subject to change. In this rapidly changing environment, the commander must be notified in real time when critical situation changes occur. For this problem, the IA proves to be extremely useful. In addition to alerting the commander about a sudden encountered problem, the IA also provides the necessary information allowing the commander to resolve the problem in near-real time.

In general, IAs should exhibit the following behavior:

1. Intelligent – ability to infer, learn
2. Proactive - goal-directed
3. Reactive - respond to domain changes
4. Autonomous - operate without human involvement.

[1] This work was supported by the BV STO, CPXXI ATD(US Army CECOM) and CPoF (DARPA) programs

An IA will exhibit some if not all of the above properties. However, strong proactive and reactive behavior may also bring about a manifestation of an undesirable anarchical personality exhibited by the IAs within a particular system application. To prevent this occurrence, a Governing agent is implemented to maintain order within the system.

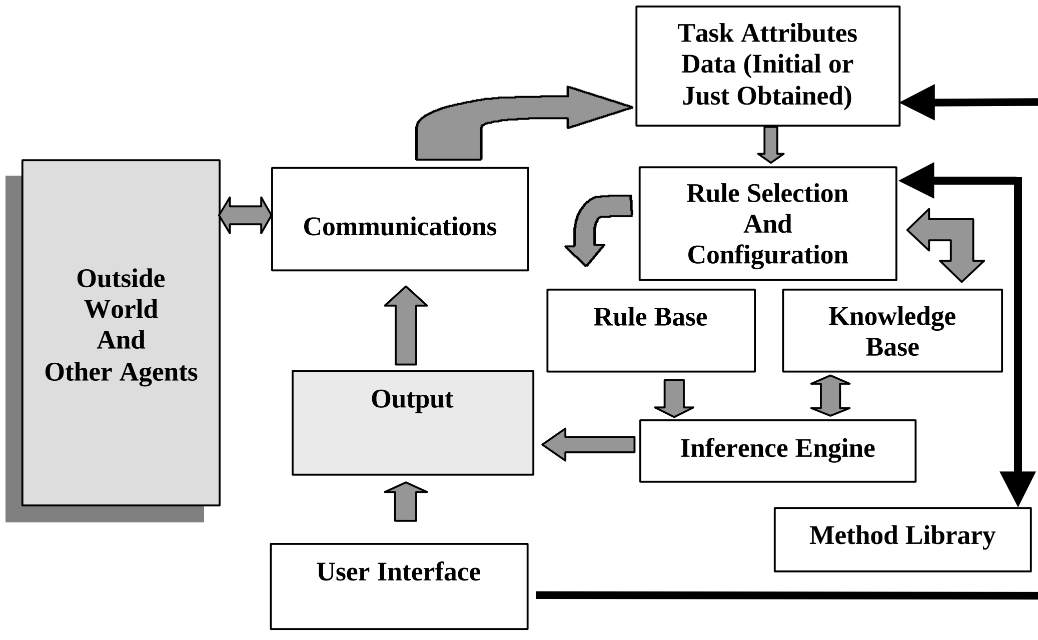
Task Attributes Data (Initial or Just Obtained)
Communications
Rule Selection And Configuration
Outside World And Other Agents
Rule Base
Knowledge Base
Output
Inference Engine
Method Library
User Interface

Figure 1
Intelligent Agent Architecture

1.1 Application

The described IA application was limited to the areas of logistics, execution monitoring, and commander's assistant. Due to time and resource limitations, but the practicality of IA technology is shown.

2.0 Architecture

The IA architecture developed was designed to incorporate all four behavioral elements described earlier and is illustrated in Figure 1. The architecture allows behavioral modification and is common to agents designed to interact with the user. This model, however, does not address the architecture of a Governing Agent or Controller.

The task attributes are pre-configured to meet the anticipated user needs. The user has an option to customize the attribute settings based on requirements. The attributes determine the tasks assigned and the behavioral tendency of the agent.

The rules are selected from the Rule Base (RB) by Rule Selection and Configuration (RSC) using selected attributes, and loaded into the Inference Engine (IE). The output generated by the Inference Engine will supply information to the client/user[2].

The IE executes selected rules and informs the user if the information supplied was conclusive. If not, the IE brokers another service, agent or human for additional information. Irrespective of available information, the IE will generate an output. However, it will reflect the completeness of information by computing a confidence level of the inference made.

2.1 Open Architecture

The open architecture design increases the product's life cycle while reducing software maintenance costs. This is accomplished by replacing outdated methods with new methods, and expanding the RB with new rules defined by the user. The software maintenance is accomplished with the aid of RSC. The following example illustrates the open architecture concept, and the role RSC plays in facilitating software maintenance. In general, a rule can be formulated as:

```
If Sensor (argument list) Then
        Effector (argument list)
End if
```

Where Sensors and Effectors are the test and inference functions, respectively.

The RSC allows the user to modify the RB and Knowledge Base (KB) with a user-friendly interface. The user is presented with task pertinent lists of sensor[3] and effector methods. The lists of Sensors and Effectors are generated based on the content of the Method Library repository. These lists can consist of old or new methods. Presented with the available methods, the user can formulate the rule. If no duplicate or similar rule exists in the RB the newly configured rule is then stored in the Rule Base.

3.0 Method and Experiment

The IA application was built using the following tactical scenario to demonstrate collaboration between Commanders Aid (CA), Execution Monitoring (EM), and Situation Awareness (SA) agents.

The scenario is as follows:
A battalion size team is to secure Objective Area (OA) Alpha at 04:00 hours. At 03:35 scouts report enemy activity in the vicinity of the OA. The enemy force is reported to be at least two companies of mechanized infantry and a small number of tanks. The lead elements of the force are four kilometers away from the OA.

[2] The term client implies software, and the term user implies a human.
[3] The sensors occasionally are more complex than, for instance, a>b. The sensor may need to compute or obtain the values of a and/or b before making the required evaluation.

The information supplied by the scout triggers the EM agent. The EM agent in turn triggers the CA

friendly artillery or/and airborne platforms. Within minutes, the CA presents the commander with the information needed to devise a plan of action based on the computed weapon effectiveness (WE) shown

Target	**Asset**	
Inf. CO-1,2	**MLRS M270**	**.85**
Inf. CO-1,2	**Howitzer 155-mm**	**.73**
Armor CO-1	**Longbow**	**.87**
Armor CO-1	**MLRS M270**	**.59**

Table 1

The equation used in computing weapon effectiveness can be defined considering the events to be dependent or independent.

WE=P(A)

Where: **WE** = Weapon Effectiveness and **P** is the probability of occurrence of event **E**, $0 \leq P(A) \leq 1$. Since **A** is a union of events we can express [5]

$$\mathbf{P(A)=P(a_1+ a_2+ a_{3+\ldots}\ a_n)}$$

If $a_1+ a_2+ a_{3+\ldots}\ a_n$ are all elements in A. The probability of union of events A and B can be extended as:

$$\mathbf{P(A\cup B)=P(A+B)=P(A)+P(B)- P(A\cap B)}$$

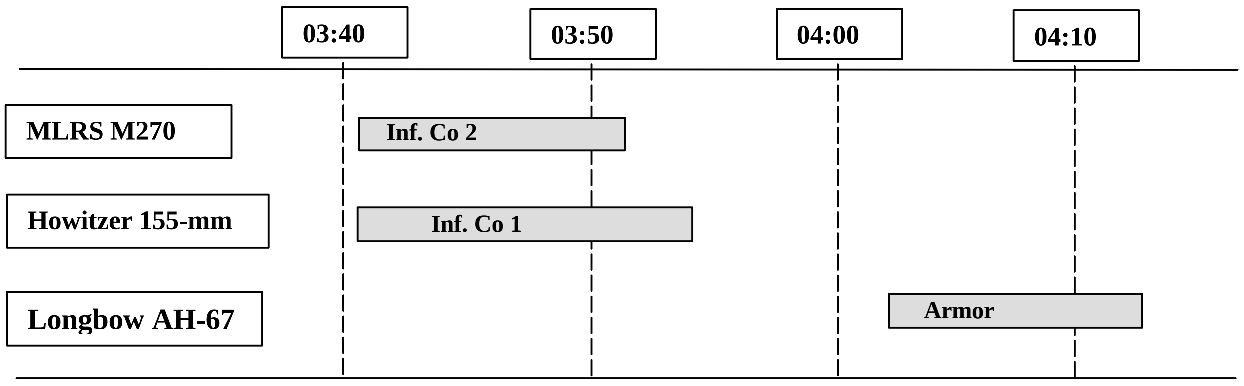

Figure 2
Task Execution Matrix

This relationship is shown in Figure 3.

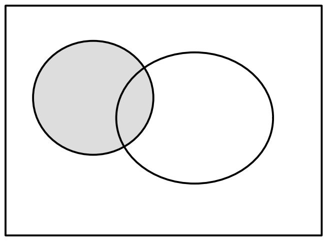

Figure 3
Venn diagram of union of events A and B

If events A and B are mutually exclusive, i.e. $\mathbf{P(A \cap B)=0}$
then:

$$\mathbf{WE= P(A \cup B)= P(A)+P(B)}$$

where

When the combined effectiveness for several weapons firing at the same target area sequentially and in a short time interval, the WE is computed as:

$$= \mathbf{1} - \prod_{i=1}^{N}(1-\mathbf{d})$$

where:

$\mathbf{d_i}$ - probabilistic contributing elements such as probability of a hit under ideal conditions, weapon effectiveness under ideal conditions.

N is the number of probabilistic contributing elements.

For example, if we use both options in Table 1, the MLRS M270 WE=.85 and the 155mm Howitzer WE=.73, then the combined WE=.96
The CA suggests a possible course of action (COA) using situation relevant rules, and computed values of WE. The results produced by the CA are displayed as a task execution matrix shown in Figure 2.

4.0 Conclusion

IA technology can add a new level of sophistication to the battlefield. The IAs can serve as the commander's intelligent assistants for instance, capable of analyzing a given situation, and suggesting elements of a possible COA. The ultimate goal of the IA is to become an intelligent help capable of anticipating user needs based on the current situation.

IAs can dramatically increase their combined capability in an environment of collaborative problem solving, regardless of their individual limitations. To optimize that potential, a good stasis between the system and the individual agents must be achieved. A well-designed architecture is of great importance in achieving this goal.

Acknowledgments

The author would like to thank the following individuals for their valuable comments and input that helped to improve and shape the final design.

- Thomas Newsome, CPXXI-ATD program manager.
- John Soos, BP-STO program manager
- Ward Page, CPoF program manager.
- Members of Battle Command Battle Labs at Fort Leavenworth; Colonel M. Heredia, Lt. Colonel J. Duquette, and Major E.F. Stockel**.**
- My colleagues Eugene Bajew and Gunther Kainz.
- William Kaplan, whose editorial comments made this paper much more palatable to the reader.
- And last but certainly not least, my dear wife Milena, whose patience, love and support were instrumental in the writing of this paper.

- **References**

1. [Dawidowicz et al, 97] Edward Dawidowicz, Lisa Tran, Richard Wong, Israel Mayk, Dirk Klose (1998), *Scenario Animator and Generator as a Command and Control Decision Making Tool for Humanitarian Assistance*, Proceedings of the Forth International Command and Control Research and Technology Symposium, Washington, DC, June 17-20, 1997.
2. [Dawidowicz et al, 96 Edward Dawidowicz, Lisa Tran, Richard Wong, Israel Mayk, Dirk Klose (1998), *Command and Control Tool: Scenario Animator and Generator*, Proceedings of the 1998 Research and Technology Symposium, pp. 504-516, NPGS, Monterey, CA, June 1996.
3. [Mayk et al, 97] Mayk I., Salton J., Dawidowicz E., Wong R., Tran L. Chamberlain S., and Brundick F., *Distributed Situation Awareness for C2 Platforms*, Proceedings of the 1997 International Conference on SMC, Orlando, FL, October 1997.
4. [Michel et al 81] Anthony N. Michel, Charles J. Herget, *Mathematical Foundations in Engineering and Science Algebra and Analysis*, Prentice-Hall, Inc, 1981.
5. [D. E. Grawoing] Dennis E. Grawoing, Decision Mathematics, McGraw-Hill Book Company, 1967, p 240.