Incorporating Temporal Constraints in the Planning Task of a Hybrid Intelligent IDS

Álvaro Herrero[1], Martí Navarro[2], Vicente Julián[2], and Emilio Corchado[3]

[1] Department of Civil Engineering, University of Burgos, Spain
C/ Francisco de Vitoria s/n, 09006 Burgos, Spain
ahcosio@ubu.es

[2] Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia,
Camino de Vera s/n, 46022, Valencia, Spain
mnavarro@dsic.upv.es, vinglada@dsic.upv.es

[3] Departamento de Informática y Automática, University of Salamanca,
Plaza de la Merced s/n, 37008 Salamanca, Spain
escorchado@usal.es

Abstract. Accurate and swift responses are crucial to Intrusion Detection Systems (IDSs), especially if automatic abortion mechanisms are running. In keeping with this idea, this work presents an extension of a Hybrid Intelligent IDS characterized by incorporating temporal control to facilitate real-time processing. The hybrid intelligent -IDS has been conceived as a Hybrid Artificial Intelligent System to perform Intrusion Detection in dynamic computer networks. It combines Artificial Neural Networks and Case-based Reasoning within a multi-agent system, in order to develop a more efficient computer network security architecture. Although this temporal issue was taken into account in the initial formulation of this hybrid IDS, in this upgraded version, temporal restrictions are imposed in order to perform real/execution time processing. Experimental results are presented which validate the performance of this upgraded version.

Keywords: Multiagent Systems, Hybrid Artificial Intelligent Systems, Computer Network Security, Intrusion Detection, Temporal Constraints, Time Bounded Deliberative Process.

1 Introduction and Previous Work

A wide range of Artificial Intelligence (AI) techniques and paradigms have been used to build Intrusion Detection Systems (IDSs). Previous studies have taken advantage of agents and Multiagent Systems (MASs) in the field of Intrusion Detection (ID) [1], [2], and different machine learning models – including Data Mining techniques and Artificial Neural Networks (ANN) – have successfully been applied to ID, such as [3], [4]. Moreover, AI techniques have been combined (genetic algorithms and K-Nearest Neighbor (K-NN) [5] or K-NN and ANN [6], among others), in order to approach ID from a hybrid point of view. In some cases they provide intelligence for MAS.

Other approaches involve the application of AI techniques in real-time environments to provide real-time systems with 'intelligent' methods to solve complex problems. There are various proposals to adapt AI techniques to real-time requirements; the most promising algorithms within this field being Anytime [7] and approximate processing [8]. One line of research in Real-Time AI is related to large applications or hybrid system architectures that embody real-time concerns in many components [8], such as Guardian [9], Phoenix [10], or SA-CIRCA (Self-Adaptive Cooperative Intelligent Real-Time Control Architecture) [11].

MOVICAB-IDS (MObile VIsualisation Connectionist Agent-Based IDS) has been proposed [12], [13], [14] as a novel IDS comprising a Hybrid Artificial Intelligent System (HAIS). It monitors the network activity to identify intrusive events.

This hybrid intelligent IDS combines different AI paradigms to visualise network traffic for ID at packet level. It is based on a dynamic MAS, which integrates an unsupervised neural projection model and the Case-Based Reasoning (CBR) paradigm [15] through the use of deliberative agents that are capable of learning and evolving with the environment. A dynamic multiagent architecture is proposed in this study that incorporates both reactive and deliberative (CBR-BDI agents [16]) types of agents. The proposed IDS applies an unsupervised neural projection model to extract interesting traffic dataset projections and to display them through a mobile visualisation interface.

Response time [17] is a critical issue for most of the security infrastructure components of an organization. The importance of a quick and smart response increases in the case of IDSs. Systems that require a response before a specific deadline, as determined by the system needs, make it essential to monitor execution times. Each task must be performed by the system within a predictable timeframe, within which accurate execution of the given response must be guaranteed. This is the main reason for time-bounding the analytical tasks of MOVICAB-IDS. A key step is the assignation of each pending analysis to available Analyzer agents (see section 2 for further details), which is performed by the Coordinator agent. Accordingly, temporal constraints are incorporated in the Coordinator agent that maintains its deliberative capabilities. With reference to the aforementioned works in the area of Real-Time AI, a Real-Time Agent (RTA) can be defined as an agent with temporal constraints in at least one of its responsibilities [18]. So, an agent assigned to real-time environments must accomplish its goals, responsibilities and tasks with the additional difficulty of temporal constraints. Such agents may have temporal bounded interactions, a modification that will affect all communication processes in the MAS where the RTA is located. These issues are discussed in this research in the case of the MOVICAB-IDS Coordinator agent, which has been modelled as a real-time agent.

This paper is organized as follows. Section 2 briefly outlines the architecture of MOVICAB-IDS. Section 3 shows the temporal bounded CBR cycle that is used as the deliberative process in some RTAs. Section 4 then describes the upgrade of the Coordinator agent in MOVICAB-IDS which works when real-time constraints are imposed. Furthermore, some results on CPU utilization and Average Execution Time are presented to show the benefits that arise from subjecting different phases of CBR to temporal constraints. Finally, the conclusions and future work are discussed in Section 5.

2 MOVICAB-IDS Architecture

MOVICAB-IDS has been designed, on the basis of Gaia methodology [19], [20], as a MAS that incorporates six agents, as can be seen in Fig. 1.

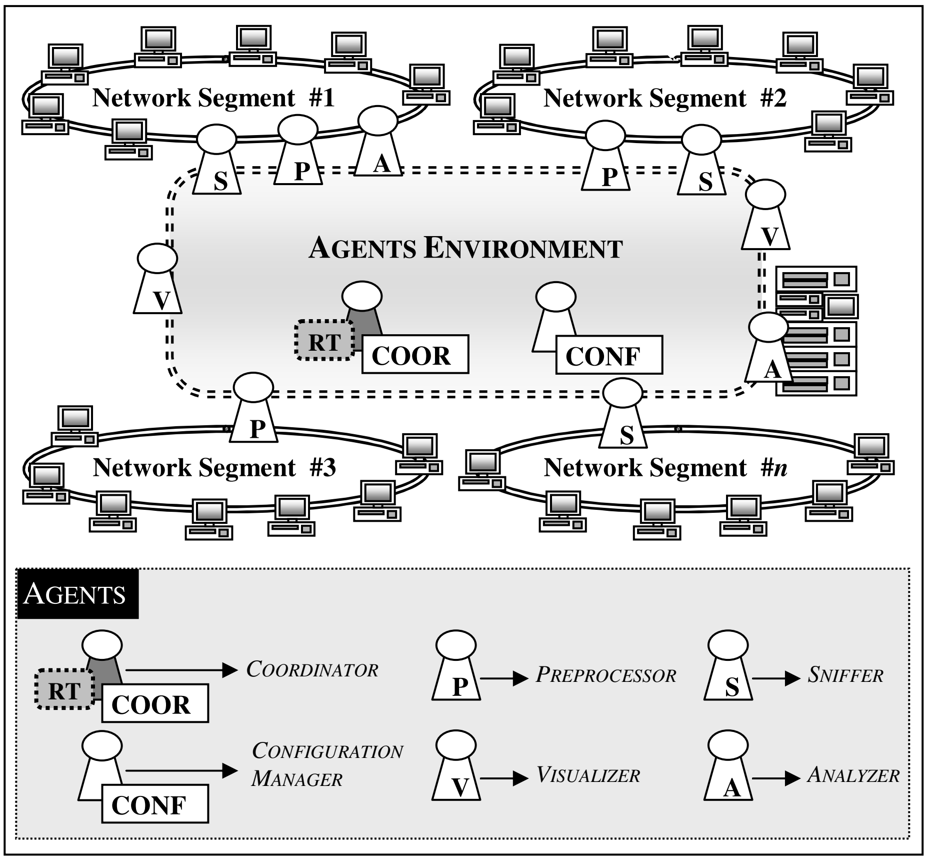

Fig. 1. MOVICAB-IDS architecture

The following agents are included in MOVICAB-IDS:

- **Sniffer:** this reactive agent is in charge of capturing traffic data. The continuous traffic flow is captured and split into segments in order to send it through the network for further processing. Then, the readiness of the data for preprocessing is communicated. One agent of this class is located in each of the network segments that the IDS has to cover (from 1 to *n*).
- **Preprocessor:** after splitting traffic data, the generated segments are preprocessed prior to their analysis. Once the data has been preprocessed, an analysis for this new piece of data is requested.

- **Analyzer:** this is a CBR-BDI agent. It has a connectionist model embedded in the adaptation stage of its CBR system that helps to analyze the preprocessed traffic data. The connectionist model is called Cooperative Maximum Likelihood Hebbian Learning (CMLHL) [21]. This agent generates a solution (or achieves its goals) by retrieving a case and analyzing the new one using a CMLHL network.
- **ConfigurationManager:** the configuration information is important as data capture, data splitting, preprocessing and analysis depend on the values of several parameters, such as packets to capture, segment length, features to extract... All this information is managed by the ConfigurationManager reactive agent, which is in charge of providing this information to the Sniffer, Preprocessor, and Analyzer agents.
- **Coordinator:** There can be several Analyzer agents (from 1 to *m*) but only one Coordinator: the latter being in charge of distributing the analyses among the former. In order to improve the efficiency and perform real-time processing, the preprocessed data must be dynamically and optimally assigned. This assignment is performed taking into account both the capabilities of the machines where the Analyzer agents are located and the analysis demands (amount and volume of data to be analysed). As is well known, the CBR life cycle consists of four steps: retrieval, reuse, revision and retention [15]. The techniques and tools used by the Analyzer agent to implement these steps are described in section 4.
- **Visualizer:** This is an interface agent. At the very end of the process, the analyzed data is presented to the network administrator (or the person in charge of the network) by means of a functional, mobile visualization interface. To improve the accessibility of the system, the administrator may visualize the results on a mobile device, enabling informed decisions to be taken anywhere and at any time.

3 Temporal Constraints for CBR-Based Agents

CBR-BDI agents [22], [23] integrate the BDI (Belief-Desire-Intention) software model and the Case-Based Reasoning (CBR) paradigm. They use CBR systems [15] as their reasoning mechanism, which enables them to learn from initial knowledge, to interact autonomously with the environment, users and other agents within the system, and which gives them a large capacity for adaptation to the needs of its surroundings. These agents may incorporate different identification or projection algorithms depending on their goals. In this case, an ANN will be embedded in such agents to perform ID in computer networks.

Although plenty of investigative effort has gone into these deliberative agents, only a few of the existing approaches cope with the application to MASs with real-time constraints [24], [25]. To apply the CBR paradigm as a reasoning mechanism in RTAs, it is necessary to adapt the techniques to be executed so that they satisfy real-time requirements. In real-time environments, the CBR stages must be temporal bounded to ensure that the solutions are produced on time; giving the system a temporal bounded deliberative case-based behaviour. Thus, a Temporal Bounded CBR (TB-CBR) mechanism [26] is suitable as the basis of the deliberative reasoning of RTAs.

The TB-CBR cycle starts at the learning stage, which entails checking whether previous cases are awaiting revision and could be stored in the case-base. The plans provided at the end of the deliberative stage are stored in a solutions list while feedback on their utility is received. This list is accessed when each new TB-CBP cycle begins. If there is sufficient time, the learning stage is implemented for cases where solution feedback has recently been received. If the list is empty, this process is omitted.

The next stage to be implemented is the deliberative stage. The retrieval algorithm is used to search the case-base and chose a case that is similar to the current case (i.e. the one that characterizes the problem to be solved). Each time a similar case is found, it is sent to the reuse phase where it is transformed into a suitable plan for the current problem by using a reuse algorithm. Therefore, at the end of each iteration in the deliberative stage, the TB-CBR method is able to provide a solution to the problem at hand, which may be improved in subsequent iterations if there is any time remaining at the deliberative stage.

The temporal cost of executing the cognitive task is greater than or equal to the sum of the execution times of the learning and the deliberative stages (as shown in equation 1):

$$\begin{aligned} t_{cognitiveTask} &\geq t_{learning} + t_{deliberative} \\ t_{learning} &\geq (t_{revise} + t_{retain}) * n \\ t_{deliberative} &\geq (t_{retrieve} + t_{reuse}) * m \end{aligned} \tag{1}$$

where $t_{cognitiveTask}$ is the maximum time available for the agent to provide a response; $t_{learning}$ and $t_{deliberative}$ are respectively the total execution times of the learning and the deliberative stages; t_x is the execution time of phase *x*; and *n* and *m* are the number of iterations of the learning and deliberative stages, respectively.

The TB-CBR algorithm can be launched by the RTA when needed and if there is sufficient time to execute it. The maximum time available to complete the execution cycle ($t_{\max}$, where $t_{\max} \geq t_{cognitiveTask}$) must be stated. $t_{\max}$ must be split into the learning and the deliberative stages to guarantee the execution of each stage. The $timeManager(t_{\max})$ function is in charge of completing this task. Through this function, the designer specifies how the agent acts in the environment. The designer can assign more time to the learning stage, if an agent with a greater learning capacity is required. Otherwise, the function can allocate more time to the deliberation stage. Regardless of the type of agent, the $timeManage\ r(t_{\max})$ function should allow sufficient time for each deliberative stage to ensure that at least one answer will be given before it ends. Naturally, the greater the time allocated to the deliberative stages, the better the response, using an anytime algorithm that enables RTAs to refine the result of each iteration. The anytime behaviour of the TB-CBR mechanism is achieved through the use of two loop control sequences. The loop condition is built by using the *enoughTime* function, which determines if a new iteration can be performed according to the total time allocated to each stage of theTB-CBR.

The first phase of the algorithm executes the learning stage if the agent has the solutions from previous executions stored in the *solutionQueue*. The solutions are stored just after the end of the deliberative stage. The deliberative stage is only launched if there is a problem in the *problemQueue* that the agent cannot solve. This configuration allows the agent to launch the TB-CBR so that it only learns (no solution is needed and the agent has enough time to reason previous decisions), only deliberates (there are no previous solutions to consider and there is a new problem to solve) or so that it performs both functions.

4 Time-Bounding MOVICAB-IDS Coordinator Agent

The MOVICAB-IDS Coordinator agent, in charge of assigning the pending analyses to the available Analyzer agents, is defined as a Case-Based Planning (CBP-BDI) agent [27]. CBP [28] attempts to solve new planning problems by reusing past successful plans [29]. The Coordinator agent plans to allocate an analysis to one of the available Analyzer agents based on the following criteria:

- **Location**. Analyzer agents located in the network segment where the Visualizer or Pre-processor agents are placed would be prioritised.
- **Available resources** of the computer where each Analyzer agent is running. The computing resources and their rate of use all have to be taken into account. Thus, the work load of the computers is measured.
- **Analysis demands**. The amount and volume of data to be analysed are key issues to be considered.
- **Analyser agents behaviour**. These agents behave in a "learning" or "exploitation" mode. Learning behaviour causes an Analyzer agent to spend more time over an analysis than exploitation behaviour does.

Table 1. Coordinator agent - representation of case features. Classes: P (problem description attribute) and S (solution description attribute).

Class	Feature	Type	Description
P	#packets	Integer	Total number of packets contained in the dataset to be analysed.
P	Analyzers / location	Array	An array (of variable length depending on the number of available Analyzer agents) indicating the network segment where the Analyzer agent is located.
P	Analyzers / features	Array	An array (of variable length depending on the number of available Analyzer agents) containing information about the resources, their availability, and pending tasks.
P	Analyzers / failures	Array	An array (of variable length depending on the number of available Analyzer agents) containing information about the number of times each Analyzer agent has stopped working in the recent past (execution failures).
S	Analyzers / plans	Array	An array (of variable length depending on the number of available Analyzer agents) containing the analyses assigned to each Analyzer agent.

As a computer network is an unstable environment, the availability of Analyzer agents changes dynamically. Network links may stop working from time to time, so the Coordinator agent must be able to re-assign the analyses previously sent to the Analyzer agents located in the network segment that may be down at any one time. These issues are included in the representation of cases, as indicated in Table 1.

As previously explained, this agent is upgraded to become a Temporal Bounded Case-Based Planning (TB-CBP) BDI agent, bringing MOVICAB-IDS closer to real-time ID.

In this environment, analysis planning must be completed within a maximum time. For this reason, RTAs, which provide the necessary control mechanisms to carry out this task, are used to complete the analysis on time. Therefore, when a new segment is ready for analysis, the Coordinator agent, which is an RTA, has a limited amount of time to assign the pending analysis to the available Analyzer agents, which have to provide an answer as soon as possible. Therefore, a temporal constraint on the process (starting with a new generated segment and ending with the Analyzer agent generating the projection) is essential to ensure prompt execution. To perform this temporal control, all the steps in the process must be known and must be temporal bounded. Additionally, the system has to be deterministic. To guarantee these conditions, the Coordinator agent takes advantage of the TB-CBP to assign the pending analysis.

The four phases of the TB-CBP cycle of the Coordinator agent are re-defined to comply with the temporal constraints. As a solution must be provided within the limited time, the retrieval and reuse stages are initially performed. When a solution for the new problem is obtained, if there is no pending analysis, the coordinator agent executes the revise and retain stages. Consequently, the four phases are defined as follows:

- **(Plan) Retrieve:** when a new pre-processed dataset is ready, an analysis is requested from the Coordinator agent. The most similar plan is obtained by associative retrieval, taking into account the case/plan description shown in Table 1. As the time required to extract a case is predictable, the real-time agent knows how long it takes to get a first solution. Moreover, in the case of the coordinator agent having more time to analysis the problem, it will attempt to improve this first solution within the available time.
- **Reuse:** the retrieved plan is adapted to the new planning problem. The only restriction is that the analyses running at that time (the results of which have not yet been reported) cannot be reassigned. The others (pending) can be reassigned in order to optimize overall performance. This phase is also temporal bounded. The Coordinator agent knows when it will finish the adaptation of the cases to the new planning problem. In this phase, as the Coordinator agent calculates when the analysis agents will finish their tasks, it either knows that it can continue building the plan, because the Analyzer agents will still be executing pending analyses when this phase is completed. Thus, the assignment of an analysis to an Analyzer Agent depends on its work load at that particular point in time.
- **Revise:** the plan revision consists of a two-fold analysis. On the one hand, planning failures are identified by finding under-exploited resources. As an example, the following hypothetical situation is identified as a planning failure: one of the Analyzer agents is not busy performing an analysis while the other ones have a list

of pending analyses. On the other hand, execution failures are detected when communication with Analyzer agents has been interrupted. Information on these failures is stored in the case base (as shown in Table 1) for future consideration. When an execution failure is detected, the CBP cycle is run from the beginning, which renews the analysis request.

- **Retain:** when a plan is adopted, the Coordinator agent stores a new case containing the dataset-descriptor and the solution (see Table 1).

The main advantages of using the TB-CBP with regard to using a CBP without temporal constraints are the maximization of CPU utilization and minimization of the average execution time of the analysis function. A set of tests were performed to validate this claim, the results of which are shown in Table 2.

Table 2. TB-CBP vs. CBP

	CPU utilization	Average Execution Time
TB-CBP	97 %	1.6 ms
CBP	72 %	2.4 ms

5 Conclusions and Future Work

This paper has presented a novel version of MOVICAB-IDS which imposes temporal constraints on the deliberative agents within a CBR architecture, which enables them to respond to events in real (both hard or soft) time. In this case, the Deliberative Coordinator agent, working at a high level with Belief-Desire-Intention (BDI) concepts, is temporal bounded by redefining the four stages of its CBP cycle. As a consequence, the Coordinator agent will always give a solution within the available time, thereby maximizing CPU utilization.

Acknowledgments. This research is funded through the Junta of Castilla and León (BU006A08); the Spanish Ministry of Education and Innovation (CIT-020000-2008-2 and CIT-020000-2009-12); the Spanish government (TIN2005-03395 and TIN2006-14630-C03-01), FEDER and CONSOLIDER-INGENIO (2010 CSD2007-00022). The authors would also like to thank the vehicle interior manufacturer, Grupo Antolin Ingenieria S.A. for supporting the project through the MAGNO2008 - 1028.- CENIT Project funded by the Spanish Ministry of Science and Innovation.

References

1. Spafford, E.H., Zamboni, D.: Intrusion Detection Using Autonomous Agents. Computer Networks: The International Journal of Computer and Telecommunications Networking 34(4), 547–570 (2000)
2. Dasgupta, D., Gonzalez, F., Yallapu, K., Gomez, J., Yarramsettii, R.: CIDS: An Agent-based Intrusion Detection System. Computers & Security 24(5), 387–398 (2005)
3. Liao, Y.H., Vemuri, V.R.: Use of K-Nearest Neighbor Classifier for Intrusion Detection. Computers & Security 21(5), 439–448 (2002)

4. Sarasamma, S.T., Zhu, Q.M.A., Huff, J.: Hierarchical Kohonenen Net for Anomaly Detection in Network Security. IEEE Transactions on Systems Man and Cybernetics, Part B 35(2), 302–312 (2005)
5. Middlemiss, M., Dick, G.: Feature Selection of Intrusion Detection Data Using a Hybrid Genetic Algorithm/KNN Approach. In: Design and Application of Hybrid Intelligent Systems, pp. 519–527. IOS Press, Amsterdam (2003)
6. Kholfi, S., Habib, M., Aljahdali, S.: Best Hybrid Classifiers for Intrusion Detection. Journal of Computational Methods in Science and Engineering 6(2), 299–307 (2006)
7. Dean, T., Boddy, M.: An Analysis of Time-dependent Planning. In: 7th National Conference on Artificial Intelligence, pp. 49–54 (1988)
8. Garvey, A., Lesser, V.: A Survey of Research in Deliberative Real-time Artificial Intelligence. Real-Time Systems 6(3), 317–347 (1994)
9. Hayes-Roth, B., Washington, R., Ash, D., Collinot, A., Vina, A., Seiver, A.: Guardian: A Prototype Intensive-care Monitoring Agent. Artificial Intelligence in Medicine 4, 165–185 (1992)
10. Howe, A.E., Hart, D.M., Cohen, P.R.: Addressing Real-time Constraints in the Design of Autonomous Agents. Real-Time Systems 2(1), 81–97 (1990)
11. Musliner, D.J., Durfee, E.H., Shin, K.G.: CIRCA: A Cooperative Intelligent Real-time Control Architecture. IEEE Transactions on Systems, Man, and Cybernetics 23(6), 1561–1574 (1993)
12. Herrero, Á., Corchado, E., Sáiz, J.M.: MOVICAB-IDS: Visual Analysis of Network Traffic Data Streams for Intrusion Detection. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 1424–1433. Springer, Heidelberg (2006)
13. Herrero, Á., Corchado, E.: Mining Network Traffic Data for Attacks through MOVICAB-IDS. In: Foundations of Computational Intelligence. Studies in Computational Intelligence, vol. 4, pp. 377–394. Springer, Heidelberg (2009)
14. Corchado, E., Herrero, Á.: Neural Visualization of Network Traffic Data for Intrusion Detection. Applied Soft Computing Accepted with changes (2010)
15. Aamodt, A., Plaza, E.: Case-Based Reasoning - Foundational Issues, Methodological Variations, and System Approaches. AI Communications 7(1), 39–59 (1994)
16. Carrascosa, C., Bajo, J., Julián, V., Corchado, J.M., Botti, V.: Hybrid Multi-agent Architecture as a Real-Time Problem-Solving Model. Expert Systems with Applications. An International Journal 34(1), 2–17 (2008)
17. Kopetz, H.: Real-time Systems: Design Principles for Distributed Embedded Applications. Kluwer Academic Publishers, Dordrecht (1997)
18. Julian, V., Botti, V.: Developing Real-time Multi-agent Systems. Integrated Computer-Aided Engineering 11(2), 135–149 (2004)
19. Zambonelli, F., Jennings, N.R., Wooldridge, M.: Developing Multiagent Systems: the Gaia Methodology. ACM Transactions on Software Engineering and Methodology 12(3), 317–370 (2003)
20. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia Methodology for Agent-Oriented Analysis and Design. Autonomous Agents and Multi-Agent Systems 3(3), 285–312 (2000)
21. Corchado, E., Fyfe, C.: Connectionist Techniques for the Identification and Suppression of Interfering Underlying Factors. International Journal of Pattern Recognition and Artificial Intelligence 17(8), 1447–1466 (2003)
22. Corchado, J.M., Laza, R.: Constructing Deliberative Agents with Case-Based Reasoning Technology. International Journal of Intelligent Systems 18(12), 1227–1241 (2003)
23. Pellicer, M.A., Corchado, J.M.: Development of CBR-BDI Agents. International Journal of Computer Science and Applications 2(1), 25–32 (2005)
24. Carrascosa, C., Terrasa, A., García-Fornes, A., Espinosa, A., Botti, V.: A Meta-Reasoning Model for Hard Real-Time Agents. In: Marín, R., Onaindía, E., Bugarín, A., Santos, J. (eds.) CAEPIA 2005. LNCS (LNAI), vol. 4177, pp. 42–51. Springer, Heidelberg (2006)

25. Surka, D.M., Brito, M.C., Harvey, C.G.: The Real-time ObjectAgent Software Architecture for Distributed Satellite Systems. In: IEEE Aerospace Conference 2001, vol. 6, pp. 2731–2741 (2001)
26. Navarro, M., Heras, S., Julián, V.: Guidelines to Apply CBR in Real-Time Multi-Agent Systems. Journal of Physical Agents 3(3), 39–43 (2009)
27. Bajo, J., Corchado, J., Rodríguez, S.: Intelligent Guidance and Suggestions Using Case-Based Planning. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS (LNAI), vol. 4626, pp. 389–403. Springer, Heidelberg (2007)
28. Hammond, K.J.: Case-based Planning: Viewing Planning as a Memory Task. Academic Press Professional, Inc., London (1989)
29. Spalzzi, L.: A Survey on Case-Based Planning. Artificial Intelligence Review 16(1), 3–36 (2001)