cONCIENS: Organizational Awareness in Real-Time Strategy Games

Sergio ALVAREZ-NAPAGAO, Ignasi GÓMEZ-SEBASTIÀ, Javier VÁZQUEZ-SALCEDA [a,1] and Fernando KOCH [b,2]

[a] *Knowledge Engineering and Machine Learning Group*

[b] *Institute of Information and Computing Sciences*

Abstract. The implementation of AI in commercial games is usually based on low level designs that makes the control predictable, unadaptive, and non reusable. Recent algorithms such as HTN or GOAP prove that higher levels of abstraction can be applied for better performance. We propose that approaches based on Organizational Theory can help providing a sound alternative for these implementations. In this paper we present cONCIENS, an integration of the ALIVE organizational framework into commercial games. We introduce a proof-of-concept implementation based on the integration to *Warcraft III*.

Keywords. artificial intelligence, organizational theory, games

Introduction

Artificial Intelligence (AI) in commercial games provides the means to enhance the two-way communication with the human player by delivering the illusion of "intelligence" in the non-player characters' (NPCs) behavior [11]. Although some specific types of AI algorithms, such as pathfinding or collision detection, have evolved to a mature state, the implementation of behavioral or strategical reasoning is, in most of the cases, still far from aligned with academic AI.

The current issues of commercial games AI are related to high-level concepts of gaming such as realistic virtual actors, automatic content and storyline generation, dynamic learning, or social behavior. Tackling these issues could represent a qualitative improvement on gaming experience from the player perspective and academic research on AI has good opportunities to provide solutions to these challenges[4,12].

We argue that the following issues are a consequence of the use of domain-dependent low-level approaches [3]:

- Blind specifications: the NPCs are programmed on *how* to act in reaction to environmental and/or other players conditions, but not *why* to act in a given manner; hence, there is no real purpose behind actions taken and, in most cases, these actions do not look "natural" from the human player's perception.

[1]Departament de Llenguatges i Sistemes Informàtics. Universitat Politècnica de Catalunya (UPC) C/Jordi Girona 1-3, E-08034 , Barcelona, Spain. E-Mail:**{salvarez,igomez, jvazquez}@lsi.upc.edu**

[2]Utrecht University, PO Box 80089, 3508TB Utrecht, The Netherlands. E-Mail:**fkoch@cs.uu.nl**

- Lack of flexibility/adaptiveness: the rule-based actions are limited and reactive to external conditions, unable to evolve, and providing reduced pro-activeness.
- Strange behavior: the behavior of the NPCs do no reflect the aspects of sociability and "participating in a whole", leading to unnatural actions from the human player's perception.
- Predictable behavior: NPCs' tactics are easily discoverable by the human player and, after some time, predictable, leading to negative perception.
- Low reusability, as the solutions are commonly tailored to specific scenario domains and, therefore, not re-usable through different games even if they belong to the same genre.

As discussed on [3], our hypothesis is that it is possible to create elaborate solutions for the issues of both individual behavior control and collective strategy techniques by integrating models based on Organization Theoretical methods to control NPCs' behavior. This theory contributes to the systematic study of how actors behave within organizations. Hence, the actors in a game are described as an organization which behavior is based on specific roles, norms, dependencies, and capabilities.

In this paper we present cONCIENS: a set of tools for Game AI developers to model gaming scenarios using social structures. In Section 1 we show how our proposal relates to existing state-of-the-art. Section 2 introduces our proposed architecture and how to integrate to existing games. Section 3 provides details on one of our proof-of-concept case studies, and Section 4 describes our proposal of validation. The paper concludes with Section 5, where we discuss our achievements and propose future work.

1. Related Work

There are already examples showing that higher levels of abstraction can be successfully used in commercial games' AI. Actually, some recent important commercial games such as *F.E.A.R*[14] or *Fallout 3*, have started to apply more complex cognitive patterns by using *GOAP* (Goal-Oriented Action Planning), a simplified and optimized version of *STRIPS* that allows for real-time planning of actions with pre- and post-conditions, even outperforming *Finite State Machine*-based algorithms in some scenarios[10]. Thus, these games execute complex symbolic reasoning not only about *how* to execute certain actions, but also about *what* to execute at each moment. We believe that, by using an even higher level of abstraction in order to reason also about *why* actions have to be performed, methods such as GOAP can be complemented and improved.

Adaptiveness in games has been already explored in academic AI research. However, existing approaches are either focused on individual reasoning[8,9], or do not take into account high-level definitions that would allow for reasoning *why* to make a particular decision on a specific context[17]. These approaches can get advantage of ALIVE by extending individual agents' reasoning cycle with *organizational awareness*.

In fact, organizational frameworks such as OperA[6] are already being explored for their use in *serious games*. In [18], organizational specifications are used to create a distributed intelligent task selection system that adapts to the player skill level and to model the storyline. With our work we intend to advance on this line of work by generalizing the use of organizational models for *fun games*, more focused on the realism of gaming experience, rather than on user modeling and learning.

2. What is cONCIENS?

The design and development of cONCIENS is part of the research for the European Project ALIVE. ALIVE [2] aims to combine existing work in coordination and organizational structures with the state-of-the-art in service-oriented computing, allowing system architects to build service-oriented systems based on the definition of organizational structures and how they interact. This framework defines three structural levels, which form the ALIVE environment depicted in Figure 1:

- The *Service Level* augments and extends existing service models in order to make components aware of their social context and of the rules of engagement with other services via semantic demarcation technologies.
- The *Coordination Level* specifies the high-level patterns of interaction (known as *workflows*) among services by using powerful coordination techniques from recent agent research. These *workflows* can be adapted at runtime, which is useful when the system has to react to unexpected events (such as failures and exceptions).
- The *Organizational Level* provides the *Service* and *Coordination* levels with a social context, specifying the organizational rules that govern interaction. This level makes services *organizational aware*, that is, services are aware of system's high-level objectives, structure and normative restrictions. This reflects in task allocation, *workflow* generation and agreement at the coordination level. For instance, the system will prevent *workflows* that violate normative restrictions from being generated and tasks are to be allocated to appropriate actors as defined on the organizational structure. This level also benefits from recent research in organizational dynamics to allow the structural adaptation of the system when changes on rules or restrictions happen.

The ALIVE Framework allows Game AI developers to think in terms of *why-what-how* when defining the decision-making actions for NPCs. That is, at the Organizational level, the developer defines "why to do something" by describing the elements of the organizational structure in terms of organization objectives, roles, norms, and restrictions. At the Coordination level, the developer defines "what to do" based on possible solutions and tasks to realize in specific situations; finally, at Service level, the developer defines "how to do it" in terms of which actual, low-level actions to perform in order to realize those tasks.

Moreover, the ALIVE framework applies *substantive norms* that define commitments agreed upon actors and are expected to be enforced by authoritative agents, imposing repair actions and sanctions if invalid states are reached. Substantive norms allow the system to be flexible, by giving actors –human or computer-controlled– the choice to cause a violation if this decision is beneficial from an individual or collective perspective.

Finally, the ALIVE Framework provides useful tools to define these elements, such as *OperettA*[13], a visual tool implemented as an Eclipse IDE plugin, which allows to specify the organizational concepts of roles, interactions and norms. These structures are implemented as coordination agents, used to build coordination plans for groups of agents enacting roles within the organization. Agents interact for enacting their roles either via direct communication or via service invocation. Monitors observe agent interactions. When these interactions are put together with the normative and organizational

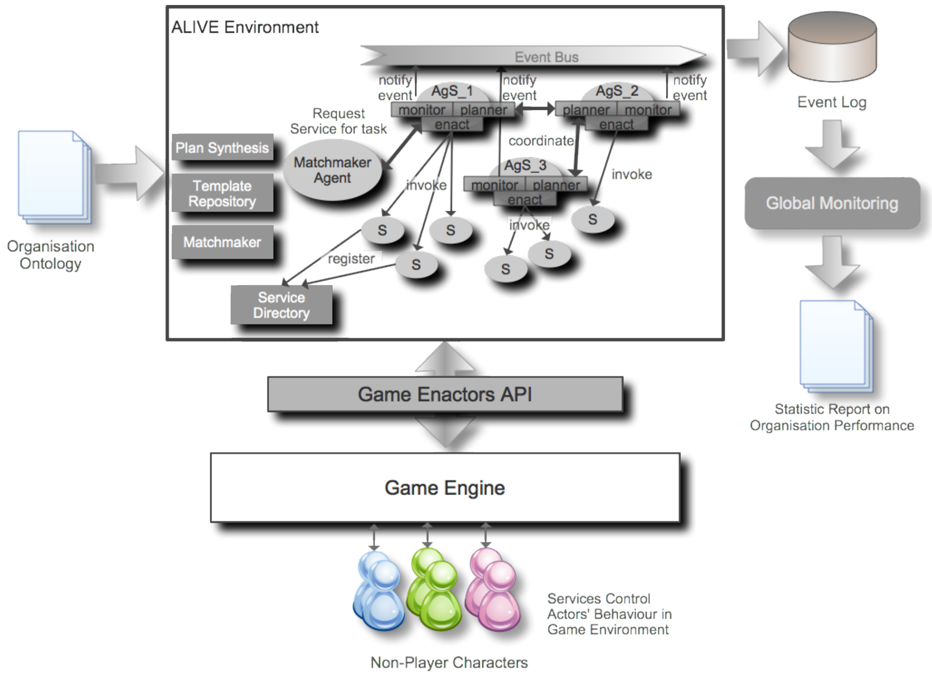


Figure 1. cONCIENS architecture

states – e.g. obligations, permissions, roles – they allow the agents to reason about the normative effects of their actions. The detection of normative states is a passive procedure that consists in monitoring past events and checking them against a set of active norms[1].

This set of tools and methods provides inherent support to the development of complex, re-usable Game AI solutions. Figure 1 depicts the global architecture of cONCIENS, which extends the ALIVE environment by providing:

1. A practical solution to couple agents to the Game Engine, by defining the Game Enactor programming interface.
2. A tool to describe the Organization Ontology, which contains a representation of agent structures.
3. The elements to describe game actors' behavior via social structures based on norms, roles and their enactment, promoting the balance between autonomy and story direction.

The research aim of cONCIENS is to provide solutions to the issues presented in Section by integrating the ALIVE framework to academic and commercial games. This approach provides extended flexibility to the elements that imply intelligent behavior, e.g. actors and characters, teams of individuals, and narrative storylines. In addition, it provides a methodology and metrics [16] that can be applied to evaluate the organizational behavior using the games' environments as simulation scenarios. Hence, it would be possible to compare, learn, and improve NPC's behavior with an approach based on organization theoretical solutions for Game AI, contributing to overall flexibility and adaptiveness.

3. A case study based on Real-Time Strategy: *Warcraft III*

We are currently testing CONCIENS with different games to validate our proposal, including *Grand Theft Auto IV*, *Lincity* or *World of Warcraft*. We intend to analyze what is the advantage in terms of realism, flexibility and adaptability. Moreover, our application to simulation environments will provide results useful for organizational research. In this paper, we focus on *Warcraft III*, a popular real-time strategy commercial game.

3.1. Real-time strategy games

For many years, computer war games have been designed as turn-based games. Real-time Strategy (RTS) games are an evolution of turn-based war games, in which the player has to command a team of virtual individuals with diverse capabilities to achieve a common objective, usually to defeat the teams of human- or computer-controlled rivals. Other sub-objectives include the collection and micro-management of resources, technological evolution, and so on.

From the AI development point of view, RTS games present two common issues:

1. Computer-controlled opponents become rapidly predictable and easily defeatable by using simple yet optimal strategies. NPC adaptation is rarely seen.
2. Although – at a high level of abstraction – the concepts and strategies of a RTS games are common to all of them, it is difficult to find AI solutions that can be reused, even between games from the same companies (see Figure 2).

function UpgradeEx takes nothing	wait_build 2 forge	rule getNextGathererUpgrade {	(defrule (goal 16 0)
local unit u = GetTriggerUnit()	upgrade 1 p_ground_weapon 70	int upID=kbGetCheapestUpgrade(tID);	(can-research ri)
local integer id = GetUnitTypeId(u)	upgrade 1 p_plasma_shield 70	int pID=aiPlanCreate(id, planProg);	=>
call DisableTrigger(trg_upgrade)	wait 2700	aiPlanSetVariableInt(pID, upID);	(release-escrow wood)
call IssueImmediateOrderById(u, up)	wait_build 1 cybernetics_core	aiPlanSetDesiredPriority(pID, 25);	(release-escrow food)
endfunction	upgrade 1 p_armor 70	aiPlanSetEscrowID(pID, cEscrowID);	(release-escrow gold)
	upgrade 2 p_plasma_shield 70	aiPlanSetActive(pID);	(release-escrow stone)
	wait 3600	}	(research ri))

Figure 2. Example AI scripts for *unit upgrading*: (from left to right) *Warcraft III* and *Starcraft* (*Blizzard*), and *Age of Mythology* and *Age of Kings* (*Microsoft*)

For these two issues, this is an scenario that could very well benefit from the adaptability offered by the ALIVE infrastructure and serve as a useful proof-of-concept. RTS games are also interesting for our purpose in the sense that the concepts they deal with can be directly mapped to the ALIVE domain, i.e., organizational structure, roles, role hierarchy, objectives, and coordination. We aim to produce computer opponents capable of adapting to unpredictable scenarios by dynamically improving at the organization and coordination layers. Moreover, this type of game would provide us a clear visual interface to execute simulations of organizations in real-time.

3.2. Modeling

We have modeled the organizational specification of an abstract RTS game. This specification identifies the set of stakeholders, the goals of each of them, landmarks and scenes related to those goals, and the normative structure of the system. The set of stakeholders

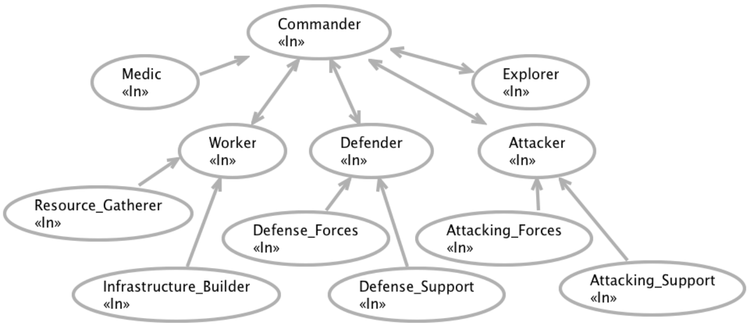


Figure 3. Social structure for generic RTS games (*OperettA* Tool screenshot)

includes *commander*, *medic*, *explorer*, *worker*, *attacker*, or *defender*. These are directly mapped to *roles*.

For each *role*, its *goals* have been identified. as well as the hierarchical relationships among the set of *roles*. Figure 3 shows the roles (nodes) and the hierarchical relations between them (edges) in a graph-like representation. For instance, in order to fulfill the objective *Produce_new_worker*, the objective *Gather_Gold* has to be taken care of. As *Produce_new_worker* is pursued by the role *Unit_producer*, and *Gather_Gold* by the role *Gold_Gatherer*, a relationship between the roles *Unit_producer* and *Gold_Gatherer* is created, because the first role is dependent on the second one for the fulfillment of its objective. For each objective, a *state description* is modeled, representing the state of the world where the objective has been fulfilled.

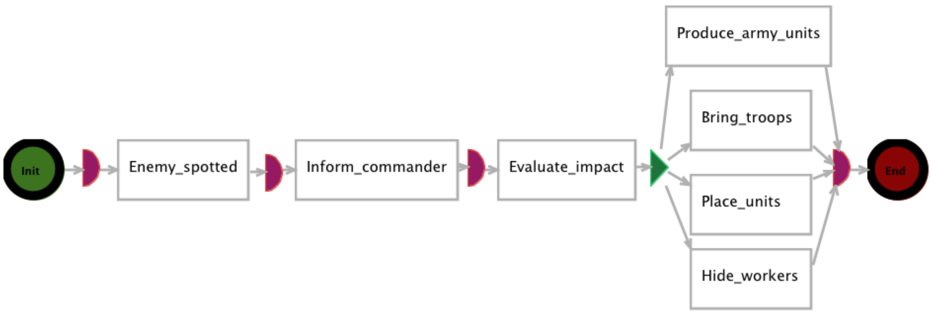
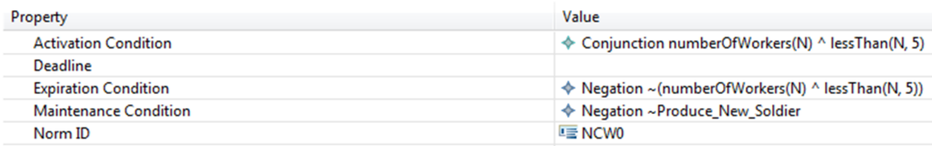


Figure 4. Interaction structure for *Defend city* (*OperettA* Tool screenshot)

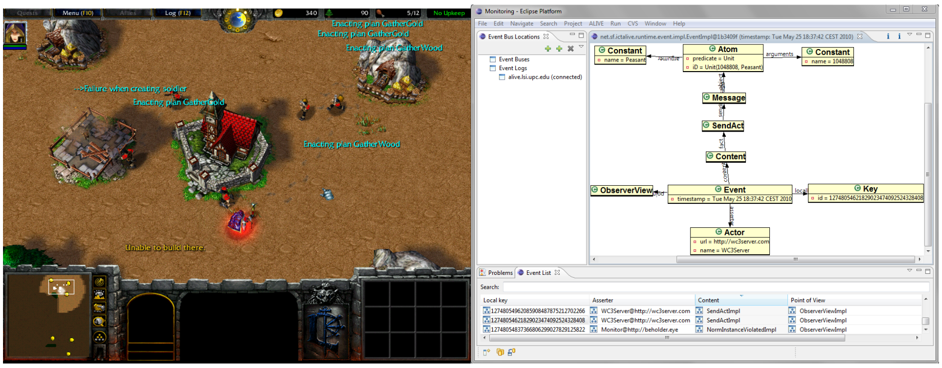
To define how each of these goals must be accomplished, *landmarks* are defined: for each objective, a set of ordered landmarks which must hold true in order to achieve a certain goal defines a *scene*. For each scene, an instance of its execution in the actual environment entails a certain state of one or more objectives. *Figure* 4 shows an example of scenes (nodes) with the transitions between them (edges) in a graph-like representation. For instance, when the objective *Gather_wood* is fulfilled, the landmark *Wood_Gathered* is reached. Role *Wood_Gatherer* is involved as the landmark *player* because it has the objective *Gather_wood* assigned.

The last element to be defined on the organization level is the set of norms. Norms are defined by the *activation*, *maintenance* and *expiration* conditions [15], modeled as



Property	Value
Activation Condition	Conjunction numberOfWorkers(N) ^ lessThan(N, 5)
Deadline	
Expiration Condition	Negation ~(numberOfWorkers(N) ^ lessThan(N, 5))
Maintenance Condition	Negation ~Produce_New_Soldier
Norm ID	NCW0

Figure 5. Norm example applied to our case (*OperettA* Tool screenshot)



(a) Game (b) Monitor Tool

Figure 6. Warcraft III units enacting actions sent from the ALIVE platform

partial state descriptions. Figure 5 shows an example of a norm modeled for the use case: *it is forbidden to produce a soldier unless 5 workers are already available*.

3.3. Implementation

We have designed an intelligent agent that is connected to the *Warcraft III* game through the CONCIENS Game Enactor, allowing for bidirectional communication via sockets. The agent is organizational aware by reading and integrating the ALIVE organizational and coordination models into its reasoning cycle.

The low-level events are obtained through the CONCIENS Game Enactor, and the ALIVE environment provides mechanisms for the interpretation of these event, providing organizational meaning. Thus, the agent is capable of perceiving the "state of the world", reacting to events happening in the game at runtime, e.g. a unit being created, or a soldier spotting an enemy, and of reasoning about which actions should be taken in the game, taking into account the current state of the world and the organizational constraints (e.g. objectives and normative constraints). Also, an agent may (or may not, depending on the individual utility) decide to discard a particular action if a norm is forbidding to enact it given the current state of the world.

The Game Enactor allows agents to enact actions in the game (see Figure 6). Once the reasoning process has decided which are the next actions to be performed, agents are able to communicate with the game, making the unit responsible of each action to enact it according to the role and plan structures defined in the organizational specification.

Currently, CONCIENS agents are implemented in Java on top of the *AgentScape* multi-agent platform. These agents are organizational aware, and are capable of planning and enacting plans based on ALIVE models. The planning process can be adjusted by an internal configuration module called Plan Rules (see Figure 7).

4. Future work: validation by organizational model competitive comparison

In [3] we discussed the need for metrics to analyze the impact of integrating ALIVE with commercial games. In this paper we propose the use of reorganization techniques to evaluate the effects of using and dynamically modifying organizational models. In this proposal, we confront a CONCIENS-controlled army against another CONCIENS-controlled army. Applying the reorganization mechanisms currently being developed for the ALIVE project, we test different variations of the same organizational model by analyzing their performance in battles between them. This performance is measured by checking the fulfillment of the organizational objectives.

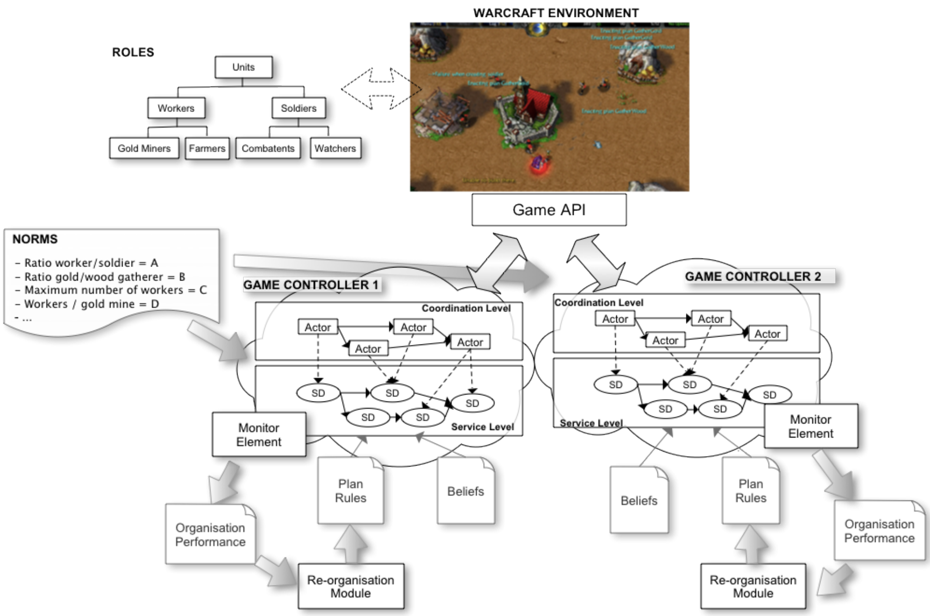


Figure 7. Reorganization in *Warcraft III*

In Figure 7 we show the architecture of the CONCIENS vs. CONCIENS battle. We have created the organizational model reflecting the *Warcraft III* environment, as seen in Section 3, including the definition of norms that guide and affect the objectives. Each ALIVE environment or Game Controller, via the execution of its agents, combines information from norms, organizational structure and the state of the world from current states of the *Warcraft III* environment. This state of the world is continuously updated and new plans are formed based on the Plan Rules.

In these battles, two Game Controllers are connected to the *Warcraft III* Environment through the Game API. Each controller is attached to one organization (one army), but both controllers can load the same or different norms and plan rules configuration:

each instance may play a variation of the same organizational model. The objective is to compare the impact of different Plan Rules and/or Norms' configurations on the organizational performance of each organization (army).

To test this impact, the Monitor continuously calculates the organizational performance for each instance as a set of operation metrics based on the norms fulfillment and the achievement of game objectives (e.g. wealth, units, etc.), and generates an individual report. The Reorganization Module may alter the Plan Rules and/or the Norms' parameters, aiming to improve the organizational performance.

These monitoring functionalities are already implemented and the outputs of the organizational behavior are being exported into data for statistical processing. The results of this validation proposal will be released in future work.

5. Conclusions

Our research addresses a common problem of commercial Game AI solutions by providing an approach based on the integration of an organization theoretical control system for NPC. We suggest that this combination contributes to Game AI solutions by providing an adaptive, extensible and flexible solution to game development industry.

The main advantage of this approach is that, through cONCIENS, developers can specify NPCs' behavior in terms of "why" they should do something, not only "what" and "how" to do it: actors in a game are described as an organization whose behavior is based on specific roles, norms, dependencies, and capabilities. Our aim is to provide a methodology and tools for developers to model gaming scenarios using social structures.

We have implemented cONCIENS, an architecture for the integration of the ALIVE's organization specification and coordination framework to existing commercial games. We have implemented the cONCIENS Game Enactor middleware that proxies information in two-ways: from the actual game environment to the cONCIENS architecture and *vice versa*, allowing developers to plug the ALIVE framework to existing games, as long as the basic interface methods to control NPCs actions are available.

We conclude that this approach contributes to the Game AI issues of behavior control and strategy techniques, by providing:

- open specifications where NPCs are programmed in terms of *why* they must act in a certain way;
- enhanced flexibility and adaptiveness by describing NPC's behavior based on organizational terms;
- more "natural behavior" as NPC may act autonomously, respecting environmental conditions and organizational objectives that will be perceived as "natural"; and
- improved reusability, as the proposed solution is generic and can be attached to a variety of commercial games through a common interface and customized organizational models.

In this paper we have shown the results of the implementation previously proposed in [3]. The tools are already available at *http://{conciens,ict-alive}.sourceforge.net*, and the methodology will be provided in future work. We have also introduced a proposal to evaluate the impact of using and adapting organizational models on the performance of cONCIENS-controlled teams. Thus, also as future work, we intend to complete our implementation by integrating the proposal of Section 4. We are also extending the implementation to other games.

Finally, although we are not encountering scalability problems with the *Warcraft III* scenario at the moment, we find *AgentScape* to be not very well suited for real-time, event-intensive scenarios. Therefore, we are currently exploring the possibility of using a combination of *Pogamut*[7] and *2-APL*[5] as the agent platform for cONCIENS.

Acknowledgments

This research has been funded by the FP7-215890 ALIVE project, funded by the European Commission. Javier Vázquez-Salceda's work has been also partially funded by the "Ramón y Cajal" program of the Spanish Ministry of Education and Science.

References

[1] Aldewereld, H., Alvarez-Napagao, S., Dignum, F., Vázquez-Salceda, J.: Making Norms Concrete. (Proceedings of 9th International Conference on Autonomous Agents and Multiagent Systems 2010)

[2] Alvarez-Napagao, S., Cliffe, O., Vázquez-Salceda, J., Padget, J.: Norms, Organisations and Semantic Web Services: The ALIVE approach. (Workshop on Coordination, Organization, Institutions and Norms at MALLOW'09 2009)

[3] Alvarez-Napagao, S., Koch, F., Gómez-Sebastià, I., Vázquez-Salceda, J.: Making games ALIVE: an organisational approach. 2nd Int. Workshop on Agents for Games and Simulations (pp 112–124 2010)

[4] Charles, D.: Enhancing gameplay: Challenges for artificial intelligence in digital games. (Proceedings of the 1st World Conference on Digital Games 2003)

[5] Dastani, M., Hobo, D., Meyer, J.: Practical extensions in agent programming languages. (Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems 2007)

[6] Dignum, V.: A model for organizational interaction: based on agents, founded in logic. (PhD Thesis, Utrecht University 2004)

[7] Gemrot, J., Kadlec, R., Bída, M., Burkert, O., Píbil, R.: Pogamut 3 Can Assist Developers in Building AI (Not Only) for Their Videogame Agents. (Agents for Games and Simulations 2009)

[8] Laird, J., Newell, A., Rosenbloom, P.: SOAR: An architecture for general intelligence. (Carnegie-Mellon University Technical Report 1987)

[9] Leite, J., Soares, L.: Evolving characters in role playing games. (Cybernetics and Systems 2006)

[10] Long, E.: Enhanced NPC behaviour using goal oriented action planning. (PhD Thesis, University of Abertay-Dundee 2007)

[11] Millington, I., Funge, J.: Artificial Intelligence for Games. (Morgan Kaufmann 2009)

[12] Nareyek, A.: Game AI Is Dead. Long Live Game AI! (IEEE Intelligent Systems 2007)

[13] Okouya, D., Dignum, V.: OperettA: a prototype tool for the design, analysis and development of multi-agent organizations. Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: demo papers (pp 1677–1678 2008)

[14] Orkin, J.: Three states and a plan: the AI of FEAR. (Proc of the 2006 Game Developers Conference 2006)

[15] Panagiotidi, S., Vázquez-Salceda, J., Alvarez-Napagao, S., Ortega-Martorell, S., Willmott, S., Confalonieri, R., Storms, P.: Intelligent Contracting Agents Language. AISB 2008 (1, 49 2008)

[16] Penserini, L., Dignum, V., Staikopoulos, A., Aldewereld, H., Dignum, F.: Balancing Organizational Regulation and Agent Autonomy: An MDE-based Approach. Engineering Societies in the Agent World 2009, LNAI 5881 (pp 197–212 2009)

[17] Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I.: Adaptive game AI with dynamic scripting. (Machine Learning 2006)

[18] Westra, J., van Hasselt, H., Dignum, V., Dignum, F.: On-line Adapting Games using Agent Organizations. (IEEE Symposium on Computational Intelligence and Games (CIG'08) 2008)