MMOG based on MAS: The MMOG Layer

(Extended Abstract)

G. Aranda V. Botti C. Carrascosa
Univ. Politécnica de Valencia
Departamento de Sistemas
Informáticos y Computación
Camino de Vera s/n – 46022
Valencia – Spain
{garanda,vbotti,carrasco}@dsic.upv.es

ABSTRACT

Massively Multiplayer Online Games present a new and exciting domain for service-oriented agent computing, Multi-Agent Systems, Agent Organizations and Electronic Institutions in order to face the challenges of designing a modern Massively Multiplayer On-line Game. This article follows previous pieces of work in the line of Multi-Agent Systems and Massively Multiplayer Online Games research being its main focus the so called MMOG Layer: a software layer which is independent of the environment simulation and the human-interface devices.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems; K.8 [**Personal Computing**]: General—*Games*

Keywords

Multi-Agent Systems, MAS, Massively Multiplayer Online Games, MMOG

1. MMOG BASED ON MAS ARCHITECTURE

This piece of work follows in the footsteps of previous research efforts like [1] and [2] in which games in general and MMOG in particular are researched as natural scenarios for agents and MAS. A MMOG (like most complex systems) can be seen as a system split into several layered subsystems, with each layer being relatively independent and taking care of one aspect of the whole MMOG experience. From the perspective of this work, a MMOG is split into three layers: The Human-Computer-Interface (HCI), which is the client-side of the system; the Intelligent Virtual Environment, which is the virtual representation and simulation of the game environment; and the MMOG layer.

The **MMOG Layer** is a complex subsystem where all the game logics and mechanics are implemented and must be solved at run-time. It is independent of the IVE layer. It implements the game rules / norms controlling the game development. It is the *place* where all the game clients connect to play, and along with the IVE layer must facilitate game server scalability. In this line of research, this subsystem is seen as the core of a MAS and requires, at least, one agent platform as its foundation.

2. THE MMOG LAYER

Like any other agentification process, one of the key ideas is to identify the agents and types of agents that will conform the system. In this case, the agents are based on the concepts and entities that form the whole game experience of a MMOG, and are explained in more detail in [2].

As stated before, the **MMOG Layer** is essentially a dedicated, open MAS which runs the game. This MAS uses agent technologies like agent services, Electronic Institutions[5, 3] and Agent Organizations[4] to model some game mechanics, and translates the common issues and situations found in MMOGs into problems that can be solved using classic software agent features, such as agent interactions, agent communication protocols (like auctions or call-for-proposals), service-oriented computing, event-driven behaviors, role models, etc.

2.1 A common Ontology for MMOG agents

One issue left partially unsolved in previous works, and that this paper addresses, is the need for a common ontology for all these agents to be able to share information in a consistent way (i.e. they need to speak the same content language). It is clear that every game or game genre has its own mechanics, rules and players, but, in essence, all MMOGs share some common concepts, like player avatars. These common concepts can be expressed using a common set of conventions, a common knowledge base. Then, in order to express specific concepts for a given game or genre, custom ontologies that derive from the original one can be described. This common ontology is called **MMOG Ontology**. Besides providing a good starting point for expressing knowledge in MMOGs, the use of a common ontology (or set of ontologies) allows for some interesting developments: first, some parts of the MAS (or even the whole system) can be reused or cloned from game to game with little to no change needed; second, it would be possible to interchange or carry over knowledge between different MMOGs, such as player information, statistics, avatars, etc; third, it settles a common way of working on a MMOG that transcends

Cite as: MMOG based on MAS: The MMOG Layer, (Extended Abstract), G. Aranda, V. Botti, C. Carrascosa, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009),* Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 1149–1150

the current project and helps developer to establish work methodologies; and fourth, it allows for rapid game prototype creation using already available tools for developing ontologies.

The **MMOG Ontology** has been defined using the Web Ontology Language (OWL). More precisely, it is expressed using the OWL-DL (Description Logics) sublanguage for two reasons: one, some of the features of this sublanguage, like defining *one-to-many*-like relational properties, where needed; and two, the full set of the OWL language (called OWL-Full) is not deemed as being computable. Even so, it is a quite light ontology, as it is composed of some base classes that have *datatype*[1] and *object* properties. The ontology uses the *'mmog:'* prefix and it presents the following classes: **mmog: Avatar**, **mmog: Clan**, **mmog: GameBeacon**, **mmog: GameItem**, **mmog: GameZone**, **mmog: Goal**, **mmog: Profile**, **mmog: Requirement** and **mmog: RewardPolicy**.

mmog:Avatar: This is the generic class to represent an avatar within the game. It does not matter if its a player-controlled avatar (i.e. played by an AvatarAgent) or an AI-controlled avatar (played by a NPCAvatarAgent). Avatars are uniquely identified by their *hasName* property which ties each of them to a unique name string identifier within the game system.

mmog:Clan: This class represents a Player Clan that the system implements by means of an Agent Organization. It has properties referring its members, lords, goals, reward policies and constraints.

mmog:GameBeacon: A *beacon* is a sort of *'named point'* of the virtual world. A marked position inside a mmog: GameZone which is referred by name. For example, the starting point or one of the exits.

mmog:GameItem: An inanimate object within the game world uniquely identified by a code string. For example, in an adventure game, the items that the players virtually carry in their inventories are instances of a subclass of mmog:GameItem.

mmog:GameZone: A class to represent a game zone. As stated earlier, a game zone is a segment of the virtual game world with its own name, theme and uniqueness. It holds one or more requirements to be entered.

mmog:Goal: A collection of conditions expressed within valid semantic terms of the game. When all the conditions are met, then the goal is reached. For example, a condition could be the equivalent expression of *'Place object A in beacon XYZ'* or *'Avatar B is a member of Clan C'*.

mmog:Profile: The profile of a player within the game community: The player's screen name along with all the information the system wants to keep from the player, the avatars the player controls, which role the player plays within the game system and the contact list for this player (i.e. the *'friends list'*).

mmog:Requirement: A requirement to enter a given GameZone. It can specify terms regarding player clans attributes, goals or player role.

mmog:RewardPolicy: A reward policy to apply once a goal has been achieved. It holds a collection of actions defined within game terms. Actions will normally be specified using terms from derived ontologies of specific games.

And so, this ontology serves as a foundation to build other, more specific ontologies to express the knowledge of a particular game or type of game. In the next section, an example of these kind of ontologies can be found.

3. CONCLUSIONS AND FUTURE WORK

This work has continued the research line of MMOG Multi-Agent Systems by providing a starting point for building and exchanging semantic content between agents of these kind of systems. The existence of a basic ontology (a common starting point) to represent game-related knowledge is needed in order to achieve greater long-term goals like standardization, inter-systems communication or define a methodology to build these kind of system from the ground up.

In this work, such common ontology has been defined using the standard OWL-DL semantic language. It can be downloaded from the web: *http://gti-ia.dsic.upv.es/ontologies/* for deployment and use.

An interplay between two previously unrelated agents and services platforms, THOMAS and SPADE, has been created in order to allow agents of the MMOG Layer to provide and consume services specified in the OWL-S semantic language, which is derived from the OWL language used in the common ontology.

Moreover, the path has been open to properly define and specify the agent interactions and protocols that conform the whole communication scheme of the system, as now there is an ontology for these agents to use and to express their semantic knowledge and an organizational platform to support and enforce services, norms and agent organizations.

4. ACKNOWLEDGEMENTS

This work has been partially funded by TIN2006- 14630-C03-01, PROMETEO /2008/051 and GVPRE /2008/070 projects and CONSOLIDER-INGENIO 2010 under grant CSD2007- 00022.

5. REFERENCES

[1] G. Aranda, C. Carrascosa, and V. Botti. Intelligent agents in serious games. In *Fifth European Workshop On Multi-Agent Systems (EUMAS 2007)*. Association Tunnisienne dIntelligence Artificielle, 2007.

[2] G. Aranda, C. Carrascosa, and V. Botti. Characterizing Massively Multiplayer Online Games as Multi-Agent Systems. *E. Corchado, A. Abraham, and W. Pedrycz (Eds.): Hybrid Artificial Intelligence Systems (HAIS 2008), LNAI 5271*, pages 507–514, 2008.

[3] A. Bogdanovych, H. Berger, S. Simoff, and C. Sierra. Narrowing the Gap between Humans and Agents in E-commerce: 3D Electronic Institutions. *K. Bauknecht, BP oll, and H. Werthner, editors, E-Commerce and Web Technologies, Proceedings of the 6th International Conference, EC-Web*, pages 128–137, 2005.

[4] N. Criado, E. Argente, V. Julian, and V. Botti. Organizational services for spade agent platform. In *IWPAAMS07*, volume 1, pages 31–40. Universidad de Salamanca, 2007.

[5] M. Esteva, B. Rosell, J. Rodriguez-Aguilar, and J. Arcos. AMELI: An Agent-Based Middleware for Electronic Institutions. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 236–243, 2004.

[1] A property whose value is a basic data type such integer, float, string, boolean, etc.