Towards an Agent-Based Automated Testing Environment for Massively Multi-Player Role Playing Games

Markus Schatten, Igor Tomičić, Bogdan Okreša Đurić, Nikola Ivković
Artificial Intelligence Laboratory
Faculty of Organization and Informatics
University of Zagreb
Pavlinska 2, 42000 Varaždin, Croatia
Email: (markus.schatten, igor.tomicic, dokresa, nikola.ivkovic)@foi.hr

***Abstract*— Automated testing in massively multi-player on-line role playing games (MMORPG) is a challenging task due to the complexity of such games and their large numbers of mutually distributed but interacting components. Large-scale multi-agent systems (LSMAS) provide us with a suitable formalism to address such complex problems. Herein a first step towards an automated game testing environment, built for the open source Mana World MMORPG, will be presented that allows the implementation of software agents and agent organizations and provide the developers with valuable game-play and testing data.**

Keywords*— *MMORPG, LSMAS, organization, automated testing, games, software

I. Introduction

Automated game testing in general, presents a challenging problem most game developers have to face eventually due to the emerging complexity of contemporary video games. Especially massively multi-player on-line games (MMO) which allow for thousands and sometimes even hundreds of thousands players playing simultaneously, are an additional challenge due to the possible mutual interactions between players, especially in the form of organizing their behaviour to perform certain game-related tasks. Additionally MMO role-playing games (MMORPG) which, beside massiveness of simultaneous players, feature also complex logical tasks or quests for the players, are even more complex for automated testing, since implemented player bots, have to perform complex reasoning and plan their actions to solve certain tasks.

While most of the literature deals with techniques of preventing automated players from playing MMO games (like [1], [2], [3]) there were only few attempts to create automated testing environments for large-scale settings. For example in [4] Jung et al. developed the VENUS simulator and present an efficient method for simulating large numbers of virtual clients in on-line games in order to ensure game stability. In [5], [6] the VENUS II system is proposed, which allows for blackbox and scenario-based testing by generating user packages based on game grammar. In this way various game scenarios can be generated and tested in the simulation environment. In [7] Mellon provides us with lessons learned from automated testing of The Sims On-line MMO game. Therein the development team used a constrained test client, based on the actual game client application, that was controlled using various scripting approaches.

Herein we will use a game logic oriented approach that, besides load and stability testing, should also allow for the testing of actual quests in a given MMORPG. All the previously outlined approaches are not developed for role-playing games (RPGs) which include complex logical quests that often foster various forms of organizing between players. Large-scale multi-agent systems (LSMAS) provide us with the necessary foundations to address such complex scenarios [8], [9], [10], [11]. For our purpose we have used the open source MMORPG called the Mana World[1] in order to develop an agent based testing framework. The testing framework is part of a larger context of building a graphical modelling tool for the semi-automated development of LSMASs [12], [13], as part of the ModelMMORPG[2] project. This work in progress paper shows the initial efforts in building the various components of the to be established framework.

The rest of this paper is organized as follows: firstly in section II we provide a short overview of the Mana World MMORPG game. In section III a motivation for using an agent based approach to automated testing is given. In section IV an overview of the implementation of the framework is provided. In the end in section V we draw our conclusions and give guidelines for future work in this area.

II. The Mana World

The Mana World is a free open source MMORPG which is implemented with a 2D graphics interface (see figure 1), similar to classic RPG games like Zelda or Final Fantasy.

[1] https://www.themanaworld.org/
[2] http://ai.foi.hr/modelmmorpg

Fig. 1: The Mana World – an open source MMORPG

There are several official game servers available for players to connect to, but being an open source game developed under the GPL license, it allows players to implement their own local servers by using the server source code freely available at the GitHub service.

The game itself facilitates character personalization and development, mutual interaction by using several in-game mechanisms such as chat, trade, fights, social network creation, tagging (friends/enemies), party creation, story development, quest solving, etc.

The important aspect of solving quests is to encourage players to form parties as in-game organizations within which there are internal rules, task delegations, responsibilities, interrelations and other aspects of such goal-oriented communities which all are scientifically engaging for analytical observations and represent a critical input for modeling and simulations of artificial players.

The game server used in this work is called TMWAthena, and is composed of three distinct servers:

- **Character server** – dedicated to managing characters and connecting them to the map server.
- **Login server** – dedicated to managing accounts and connecting to the character server.
- **Map server** – dedicated to managing the game content (such as monsters, items, maps, etc.) and the interaction of such game contents with the players.

Understanding the structure and the communication protocols of the client/server interaction formed the basic prerequisite for further game development and manipulation.

III. Agent-based Approach to Automated MMORPG Testing

Herein we have chosen to approach the problem of automated testing of MMORPGs using LSMAS. There are several reasons for that: (1) MMORPG players are by definition distributed entities that interact, compete, collaborate and organize – a definition that closely matches the one of (artificial) agents, (2) MMORPG players have to solve complex tasks in forms of puzzles and quests indicating reasoning capabilities as of intelligent agents, (3) MMORPG avatars are situated in a predefined but complex environment in which they have to act both reactively and proactively in order to achieve game objectives, as do agents in (more or less) realistic environments.

In the mentioned ModelMMORPG project we have partially developed an organizational ontology and metamodel for graphical modelling of LSMAS (see for example [14], [15], [8]) that allows for modelling a wide range of various LSMAS scenarios. Automated testing of MMORPGs is one such scenario, as we will show further.

IV. Implementation

The testing framework consists of three logical layers: (1) a lower-level interface (dealing with the technical implementation of the Mana World client network protocol), (2) a higher-level interface (implementing actual agent classes, agent behaviours, roles, agent communication, agent reasoning, knowledge bases etc.), and (3) a modelling tool (allowing for graphical modelling of tests and generating agent implementation stubs to be developed further for individual test).

These three layers are described in more detail in the following subsections.

Our approach in developing tests is loosely a top down approach. Firstly, agent organizations, which represent the various players' forms of organizing (e.g. guilds, parties, etc.) are modelled using a graphical modelling language that we have developed. Then the model is translated into a concrete implementation of agent classes facilitated through and agent-based platform called SPADE (Smart Python Agent Development Environement) [16]. These agent classes, are then extended with low-level game client methods that allow various game related tasks like walking around, picking up items, fighting etc., to be performed by agents.

A. Lower-level Interface

The lower-level interface was loosely based on an old Python script used to implement bots for the Mana World. The script has been rewritten in full and extended with multiple details of client implementation in order to allow agents to connect to the game server, create avatars, and perform all relevant tasks to be able to solve quests.

figure 2 shows a sample session of the lower-level interface. The provided screenshots illustrate changing Python player coordinates by using the "setDestination" function of the "Connection" class.

The following listing shows the actual implementation of the Python function which enables the basic navigation of the character on the loaded map by using two-dimensional coordinate values.

```
def setDestination(self, x, y, direction):
    '''Set destination (walk to given x, y coordinates with orientation direction like in setDirection)'''
    debug("SET DESTINATION")
    debug("X: %d" %x)
    debug("Y: %d" %y)

    data = bin(x)[-10:].replace('b', '0').rjust(10).replace(' ', '0') + bin(y)[-10:].replace('b', '0').rjust(10).replace(' ', '0') + bin(direction)[-4:].replace('b', '0').rjust(4).replace(' ', '0')

    data = data[:8], data[8:16], data[16:]
    data = [int('0b' + i, 2) for i in data]

    b1, b2, b3 = data

    self.srv.sendall("\x85\0%s" % struct.pack("<BBB", b1, b2, b3))
```

B. Higher-level Interface

The higher-level interface is being developed using the mentioned SPADE platform in Python. It currently features basic agent classes, agent behaviours and communication facilities to allow agents to organize mutually. The agent classes closely resemble a BDI (belief-desire-intention) architecture in which agents act based on their beliefs (knowledge about the world they live in), desires (objectives they want to pursue) and intentions (plans and commitments on achieving objectives through acting upon the environment).

The knowledge about the Mana World is stored in a specially developed [10] SWI Prolog knowledge base that can be directly accessed by the agent through its knowledge base interface.[3].

The knowledge base also featured an automated planner based on the STRIPS algorithm [17]. In the planner all quests are coded in form of STRIPS rules comprising three parts: (1) preconditions – statements that have to be true about the world in order for the rule to be applicable, (2) deletions – statements that won't be true about the world when the rule is executed, and (3) insertions – statements that will become true about the world only after the rule has been executed. In this way game related quests can be modelled quite easily, for example to solve a quest in which the player has to buy a certain item from some NPC (non-playing character) for a certain amount of money a rule would include the following statements:

- **Preconditions**: player has enough money, player is within reach of the NPC
- **Deletions**: player hasn't the payed money any more
- **Insertions**: player has acquired the wanted item.

The planner also features a list of quests to be solved, which is queried by the player agent every time the environment is updated in order to test if the preconditions of a certain quest to be solved are met. If this is the case, the agent tries to solve the quest, else the quest is put back into the list for a later time. In case the agent doesn't have any quests to be solved it does a random walk in the environment to find an NPC and get an actual quest.

C. Graphical Modeling Tool

The basis of the system of agents using the mentioned interfaces, allowing them to communicate with the game engine, is, in the context of this paper and the accompanying research, constructed using a customized graphical modelling tool[4].

The modeling tool, in its present work-in-progress version, provides the user with functionalities that allow them to model

[3]SPADE allows for using SWI, XSB, eCLiPSe, Flora-2, SPARQL and integrated first-order logic knowledge-bases

[4]The model is available at https://github.com/Balannen/LSMASOMM

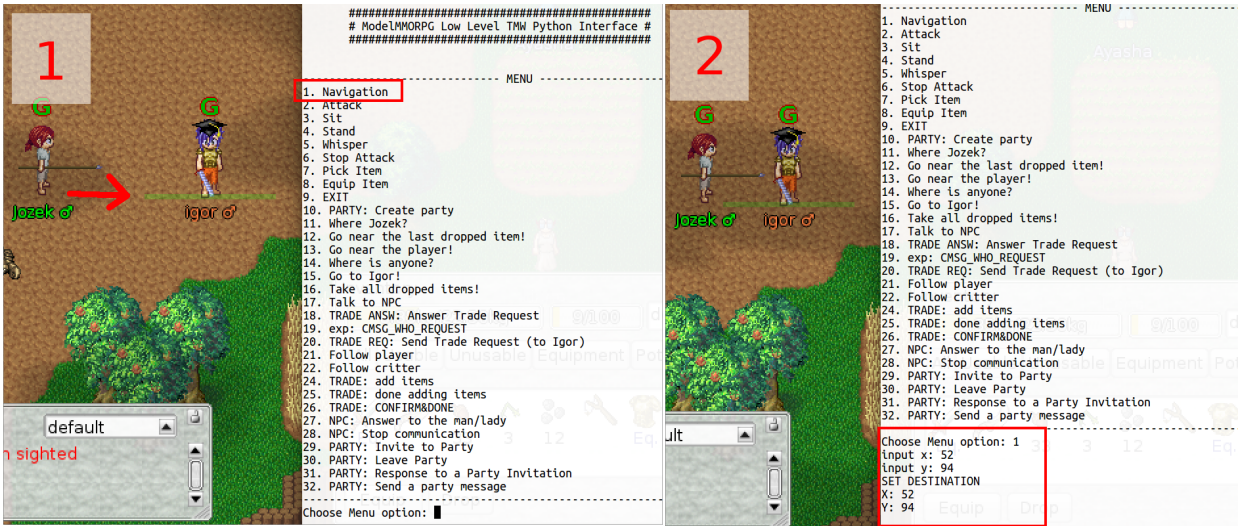

Fig. 2: Manipulating player coordinates with the Python client

a system comprising agents, groups of agents, roles as special types of grouped constraints, actions granted to individual agents upon playing a specific role, sets of actions designated as processes that are poised to fulfill specific goals, and quests (in-game goals) that consist of a series of subgoals or tasks that are reachable by a specific combination of processes, i.e. action sets.

The elementary concept of such a model, applicable to the example shown in this paper, is an organisational unit representing an individual agent. Since the idea of roles is integral to the organizational approach which, in turn, is in the basis of the models developed for the mentioned modelling tool, roles are a *de facto* central elements of a model. Figure 3 shows a situation that is possible within The Mana World game, detailed as follows. Individual agents, shown yellow in Fig. 3, are modeled following the *tabula rasa* idea, i.e. the only behavior they possess at this stage of the metamodel is intended for changing roles, thus gaining new actions. It is useful to note here that the element named *Player* in Fig. 3 represents a class of individual players, i.e. individual agent-players will be instantiated at runtime. Those individual players can form groups (most commonly called parties in an MMORPG) that utilize cooperative power of players. Each player can play a number of roles (e.g. Herbalist, Scout, Warrior), shown in blue in Fig. 3, at the moment constrained to one at any given point in time. The roles shown in Fig. 3 provide players with actions needed to successfully complete tasks (shown in red in Fig. 3) that form the quest called *The Quest for the Dragon Egg* (the topmost red element in Fig. 3.

One of the novel features in the context of LSMAS models and modeling tools, that this particular tool provides, even though only in initial stages at the moment, is SPADE code generating functionality. This functionality, still in a work-in-progress state, generates basic code, i.e. a code skeleton, for the modeled system, thus giving the model users basic overview of their system, and advanced system functionalities not explicitly visible in the model, e.g. system and services for agents' role changing actions and utilization of available knowledge on ways of solving quests based on their respective wanted tasks.

V. Conclusion & Future Work

In this work-in-progress paper we have presented the current state of implementation of an agent-based automated MMORPG testing system using a game-oriented approach. As opposed to most other approaches, the proposed system will allow for testing game-logic by implementing agents that use a specially created knowledge-base to be able to solve quests in the selected MMORPG called the Mana World. We believe that this approach can be extended to any MMORPG, given that the various quests or rules of the game can be coded into a new knowledge base.

Currently the proposed system comprises three components: (1) a lower-level interface, (2) a higher-level interface, and (3) a graphical modelling tool. Whilst the lower-level interface is fully developed at the time of writing this paper, the other two components are still being tested and developed further. Especially, in the higher-level interface, the connection between an agent's perception of the environment and the knowledge base has to be improved, so that agents can rely on up-to-date information from the environment. The modelling tool features necessary concepts for successful description of a system, but even slightly complex systems cause a cluttered and unfriendly interface. Therefore a multi-perspective modeling approach has to be enabled, allowing users to develop a model on various levels presented by different models. The code generating feature has to be upgraded appropriately to conform to the idea of multi-model modeling. Lastly, one of the foreseen improvements of the modelling tool is introduction of concepts for modelling additional selected organizational features of the system. These and similar tasks are part of our future work.

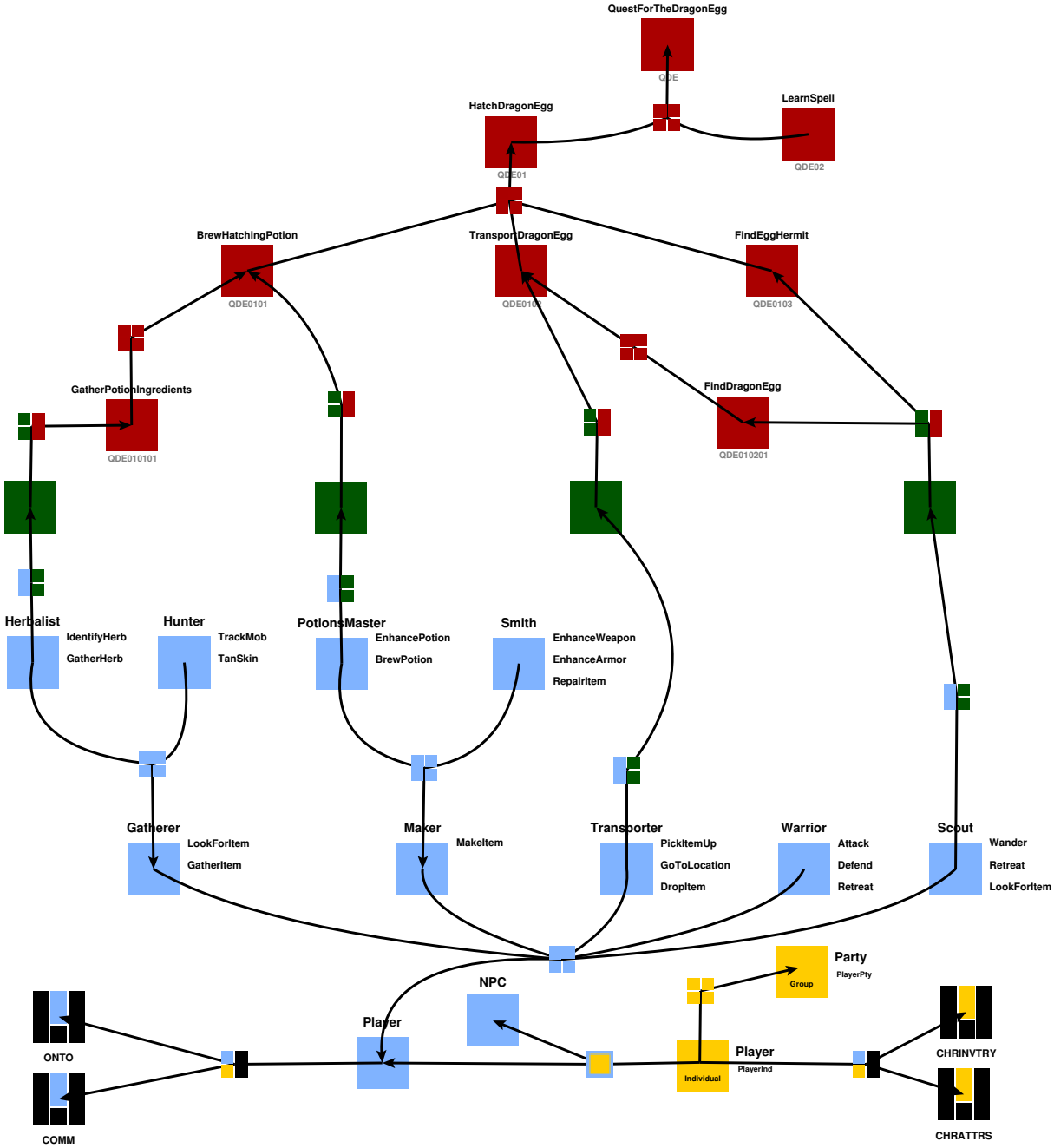

Fig. 3: Modeled selected concepts of The Mana World

ACKNOWLEDGMENT

This work has been supported in full by the Croatian Science Foundation under the project number 8537. We would also like to acknowledge the Mana World development team which often helped us with various implementation specific details.

REFERENCES

[1] J. Yan, "Bot, cyborg and automated turing test," in *International Workshop on Security Protocols*. Springer, 2006, pp. 190–197.

[2] P. Golle and N. Ducheneaut, "Preventing bots from playing online games," *Computers in Entertainment (CIE)*, vol. 3, no. 3, pp. 3–3, 2005.

[3] M. van Kesteren, J. Langevoort, and F. Grootjen, "A step in the right direction: Botdetection in mmorpgs using movement analysis," in *Proc. of the 21st Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2009)*, 2009, pp. 129–136.

[4] Y. Jung, B.-H. Lim, K.-H. Sim, H. Lee, I. Park, J. Chung, and J. Lee, "Venus: The online game simulator using massively virtual clients," in *Asian Simulation Conference*. Springer, 2004, pp. 589–596.

[5] C.-S. Cho, K.-M. Sohn, C.-J. Park, and J.-H. Kang, "Online game testing using scenario-based control of massive virtual users," in *Advanced Communication Technology (ICACT), 2010 The 12th International Conference on*, vol. 2. IEEE, 2010, pp. 1676–1680.

[6] C.-S. Cho, D.-C. Lee, K.-M. Sohn, C.-J. Park, and J.-H. Kang,

"Scenario-based approach for blackbox load testing of online game servers," in *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2010 International Conference on*. IEEE, 2010, pp. 259–265.

[7] L. Mellon, "Automated testing of massively multi-player systems: Lessons learned from the sims online," *GDC 2003*, 2003.

[8] M. Schatten, J. Ševa, and I. Tomičić, "A roadmap for scalable agent organizations in the internet of everything," *Journal of Systems and Software*, vol. in press, 2016.

[9] M. Schatten, I. Tomičić, and B. O. Đurić, "Multi-agent modeling methods for massivley multi-player on-line role-playing games," in *Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2015 38th International Convention on*. IEEE, 2015, pp. 1256–1261.

[10] M. Maliković and M. Schatten, "Artificial intelligent player's planning in massively multi-player on-line role-playing games," in *26th Central European Conference on Information and Intelligent Systems*, 2015.

[11] B. O. Đuric, I. Tomicic, and M. Schatten, "Towards agent-based simulation of emerging and large-scale social networks. examples of the migrant crisis and mmorpgs," *European Quarterly of Political Attitudes and Mentalities*, vol. 5, no. 4, p. 1, 2016.

[12] M. Schatten and P. Terna, "From players to agents via the project modelmmorpg and the slapp simulation shell," in *SWARMFEST 2016: 20th Annual Meeting on Agent Based Modeling & Simulation*, 2016.

[13] B. O. Đurić and M. Schatten, "Defining ontology combining concepts of massive multi-player online role playing games and organization of large-scale multi-agent systems," in *Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2016 39th International Convention on*. IEEE, 2016, pp. 1330–1335.

[14] B. O. Đurić, "Organizational metamodel for large-scale multi-agent systems," in *Trends in Practical Applications of Scalable Multi-Agent Systems, the PAAMS Collection*. Springer, 2016, pp. 387–390.

[15] B. O. Đurić and M. Konecki, "Specific owl-based rpg ontology," in *Central European Conference on Information and Intelligent Systems 26th Inernational Conference*, 2015.

[16] M. E. Gregori, J. P. Cámara, and G. A. Bada, "A jabber-based multi-agent system platform," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. ACM, 2006, pp. 1282–1284.

[17] R. E. Fikes and N. J. Nilsson, "Strips: A new approach to the application of theorem proving to problem solving," *Artificial intelligence*, vol. 2, no. 3, pp. 189–208, 1972.