STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVI**, Number 3, 2011

OWL-BASED MODELING OF RPG GAMES

MONICA ROMAN, IOANA SANDU, SABIN C. BURAGA

Abstract. In this paper we propose a general method of using RDF and OWL models to represent the knowledge within an electronic game. Following this approach, we are able to separate programming from resource allocation (creating and allocating the AI procedures and dialog, structuring and manipulating game elements, etc.). This work demonstrates the integration of the DL-based ontology in a game by using Jena, a Java semantic framework. As a case study, we are implementing the base for a fantasy RPG type game that includes intelligent methods for several key activities: character creation, NPC generation, a simple battle system, etc.

1. Introduction

This paper proposes a new approach for creating a game architecture and storing and retrieving data from a knowledge base by using the existing semantic Web standards: RDF (Resource Description Framework) and OWL (Web Ontology Language).

Ontologies have been previously used in games for creating Artificial Intelligence, dialog or NPC generation. For this project, we set out to create a knowledge base that will help us better define a fantasy RPG (Role Playing Game) software and help us analyze possible situations that can come up at any point in the game.

In a game ontology, rather than organizing a game by its characteristics or elements, the elements themselves are organized. This approach makes exploring issues regarding games and game play much faster and easier [12].

As games have taken an important place in mass media, they have become more complex at the architectural level. They have moved from being standalone programs built for single platforms into multi-platform, networked

Received by the editors: April 10, 2011.

2010 *Mathematics Subject Classification.* 68T30, 68U35.

1998 *CR Categories and Descriptors.* code [**I.2.4**]: Knowledge Representation Formalisms and Methods – *Representation languages*; code [**I.2.3**]: Deduction and Theorem Proving – *Inference engines*.

Key words and phrases. knowledge representation, game computing, RDF, OWL.

and/or Web-based systems [8] Therefore, linking data sources in a coherent manner has also become more important.

To accomplish our goals, first we created the RPC game ontology by using the Protégé system [3]. We modeled the basic elements needed by an RPG type game that are loosely influenced by the Game Ontology Projects classification [12] – for related work, see section 2. Also, we took into account the famous classical electronic games like Dungeon and Dragons, Heroes of Might and Magic III, Diablo, World of Warcraft, Dragon Age: Origins, The Elder Scrolls: Morrowind and Oblivion, etc. Details about the concrete modeling are given by section 3.

The second step was to use the Pellet reasoner [10] in order to obtain various inferences to assist users in creating game characters and selecting a certain type of a character. Additionally, several AI activities (NPC generation and behavior, battle scenarios, etc.) could be performed on the basis of these inferences.

The third phase was to implement a software prototype – briefly described in section 4 – in order to access the models and query the inferred ones. For this, we chose the well-known platform-independent Jena framework [13].

In our opinion, this proposal opens interesting possibilities in developing semantic Web-based games.

2. Related Work

On the subject of using semantic Web technologies in the context of computer games, the Game Ontology Project (GOP) [12] is proposed that started analyzing, studying and classifying games from an ontological point of view. Because terms and concepts are not very well defined in games, GOP identifies their important structural elements and the relationships between them, organizing them hierarchically. Its main subclasses are *Interface*, *Rules*, *Entity*, *Entity Manipulation* and *Goals*.

There are several disparate approaches of using ontologies concerning different aspects of game development. For example, [5] gives a proposal of using ontologies to position NPCs in an environment based on their behavior is given, [6] presents a method to generate correct stories on the basis of an ontological model, and [9] demonstrates that ontology-based retrieval for strategies are easier to adapt than those returned by classical retrieval methods (case-based planning) [7].

So far, ontologies have been used for various purposes in games, but none offered a full, detailed structure of an RPG game – including elements like: characters, story and resources –, that was integrated using Jena in a Java based game.

3. Ontological Model of the RPG Game

3.1. **Generic Resource Modeling via RDF and OWL.** In the semantic Web, everything is asserted via RDF (Resource Description Framework) [1] statements: $\langle s, p, o\rangle$ triples, where s – subject, p – predicate, and o – object are simply Web addresses (Uniform Resource Identifiers). Using this model, any kind of resources could be denoted, These are the basic elements of an ontology.

Resources are identified by an unique URI – for example:

`http://www.semanticweb.org/ontologies/2011/0/RPGontology#Warrior`

An object can be a resource or a literal (having a certain datatype), but subjects and predicates are always resources.

The OWL (Web Ontology Language) [1] is a broadly accepted ontology language used to model vocabularies for a certain domain. Its main components are classes, properties and relations.

Classes contain all the "things" that have the same properties and restrictions. Relations can be defined between classes or individuals. Properties are attributes that characterize classes or individuals (they can be either datatype properties that define a value or object properties that have as return value an object).

There are three types of OWL that vary in complexity, expressivity and flexibility (OWL Lite, OWL DL and OWL Full).

We based our work on OWL DL that supports more complex ontologies, without losing the computational completeness – the formal model is based on several species of Description Logics (DL) [2]. In OWL the main ontological constructs are classes, properties and objects. In DL, we are using concepts, roles and individuals. DL are having a special importance in providing a logical formalism for ontologies, in general, and the semantic Web, in particular [4].

Using RDF and OWL DL, we can form a knowledge base, where TBox (terminological box) component is composed by OWL constructs, and ABox (assertional box) component is including RDF statements.

3.2. **Proposed Ontological Model in the Context of RPG Games.** Our ontology focuses on characters, their quantitative features (e.g., level, health points) and the items they can use and equip, depending on their class and their race.

Using the ontological model, we can easily create a character or a fight in our game by retrieving and saving data in the ontology. For reasoning aspects, we focused our attention on Pellet [10]. Our proposed ontological model has the $\mathcal{ALCHIF}$(D) logic expressivity (attributive language with complements, including functionality, role inverse, and role hierarchy constructs) [11].

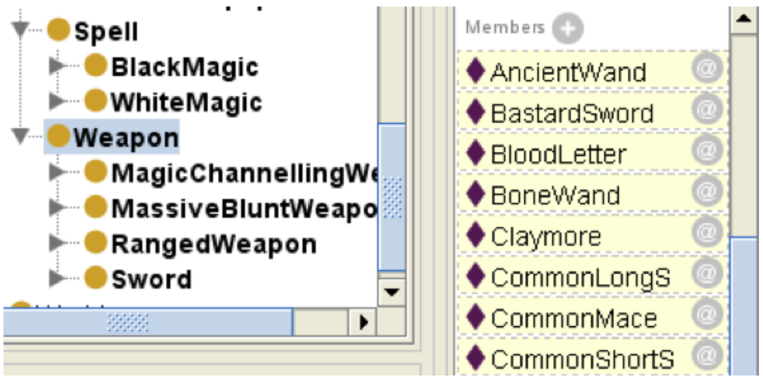

FIGURE 1. Using DL inferences to obtain individuals that are members of the class *Weapon*

3.2.1. *Population of the Game.* We started by giving a character the possibility of being one of 4 disjoint races: *Human*, *Elf*, *Orc* and *Goblin*. Each race can only use certain objects. We defined the ontological class *UsableEntiy* that contains all the entities in the game that a character can use (the equipment): weapons, armor, items and spells.

We then specified the classes *HumanEquipment*, *ElfEquipment*, *OrcEquipment* and *GoblinEquipment*, respectively, and gave them the properties that they contain objects that only humans, elves, orcs and goblins can use.

Using the Pellet reasoner, the model was enriched with the inferred appropriate subclasses and individuals:

- *HumanEquipment* $\equiv$ canBeUsedBy **some** *Human*
- *Human* $\equiv$ *Race* $\sqcap \neg($*Elf* $\sqcup$ *Goblin* $\sqcup$ *Orc*$) \sqcap$ canUse **only** (*Armor* $\sqcup$ *HealthPotion* $\sqcup$ *MagicPotion* $\sqcup$ *PoisonPotion* $\sqcup$ *RangedWeapon* $\sqcup$ *Sword*)

A character can also have a class (group) that can only use certain objects. We defined 6 disjoint character classes: *Warrior*, *Paladin*, *Wizard*, *Warlock*, *Rogue*, and *Barbarian*. Afterwards, we created the classes *WarriorEquipment*, *WizardEquipment* and so on and then the reasoner populated them with objects that only the respective group can use. For example, *WarriorEquipment* $\equiv$ canUsedBy **some** *Warrior*.

Because class equipments have more restrictions than race equipments, they will be inferred as subclasses of the appropriate *RaceEquipment* classes. This means that we linked what objects a certain race can use and what objects a certain class can use.

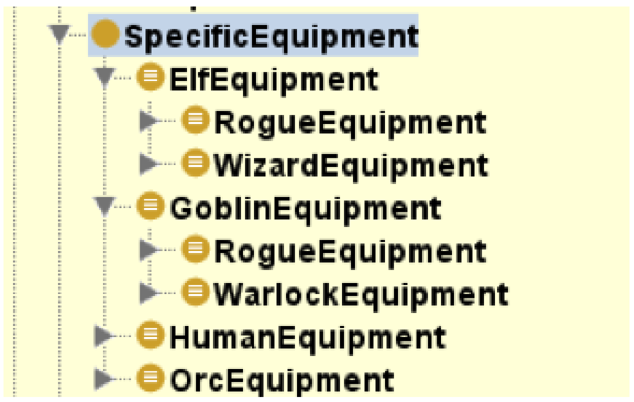

FIGURE 2. Classifying (inferring) specific equipment based on given restrictions

3.2.2. *Modeling Characters.* When creating a new character, we can choose a race for it and – based on what we specified earlier – we can get all the classes it could have (e.g., if we have an elf, then it will show us that he/she can only be a member from *Rogue* or *Wizard* classes; she/he can not be a *Warrior* because warriors can equip heavy armor, but elves can not).

Each character can have certain common quantitative features (statistics) that we modeled via datatype properties. For example, a character hasHP (Health Points), hasMP (Magic Points), hasDP (Dexterity), hasPD (Physical Defense), hasPO (Physical Offense), hasMD (Magical Defense), hasMO (Magical Offense), hasXP (Experience Points) and hasLevelNumber.

Also, every character has a gender (male or female) and a type (playable or non-playable). A NPC (Non-Playing Character) can additionally have a trait – hostile, coward and neutral – and can inhabit certain places in the world: abandoned ruin, forest, swamp, etc.

3.2.3. *Places of Interest.* For our ontology, we also created classes for shops and quests.

In shops, characters can go buy and sell usable entities – for example, a Potion Shop can have as items: some Medium Health Potions and some Minor Magic Potion, while an Armor Shop can have various types of armor.

We defined quests as a class that can have instances with data type properties like has Dialog (has Quest Accepted Dialog, has Quest Rejected Dialog, has Quest Completed Dialog, etc).

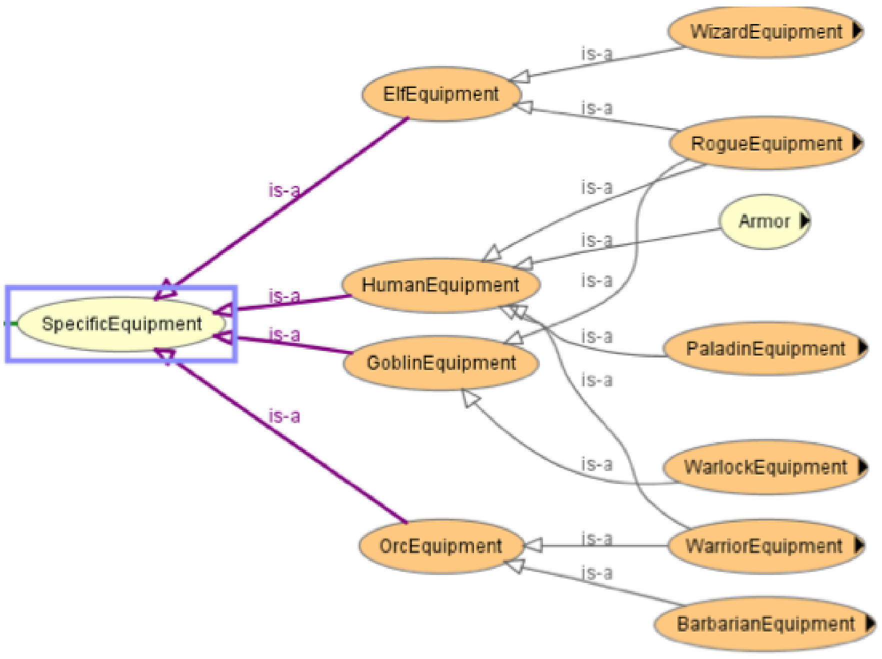

Figure 3. Visualizing inferred classes about equipments by using OWLViz plug-in

4. Implementation Details

We started by analyzing RPG games in general and studying what elements we would need for our game.

We first built a knowledge base that contains almost all of the games information and searched for ways it could help us faster develop a game. For the construction of the ontology, the best choice was to use Protégé system [3], the most widely used knowledge-base editor. It allows creating, editing and visualizing RDF or OWL specifications and automatically checks for inconsistencies. It also automatically populates classes with instances. It offers a substantial number of plug-ins and can be extended by third-party ones. The OWLViz plug-in enables a clear visualization of the asserted and inferred classes hierarchy.

We then searched for the best suited programming language that could give a proper support regarding the use of ontologies in a pragmatic manner. One of the most important aspect is to create and integrate an ontology model into a computer game.

We chose Java platform and Jena framework [13] built to be used within semantic Web applications. It allows working with RDF, RDFS, OWL, SPARQL and includes a rule-based inference engine. It connects our OWL ontology to the Java-based game prototype and enables reading and writing data in our ontological model (via OWL API). The Jena Ontology API in conjunction with a powerful reasoner (a built-in or an external) becomes a very useful instrument.

For our prototype, we stored information about individuals (the ABox compartment of the knowledge base) into RDF documents. For instance, the below mentioned RDF file holds the default values for all the game character classes: *Warrior*, *Paladin*, *Wizard*, etc. These values are specified for level 1 characters that well be used when a new playing character is created.

```
<rdf:Description rdf:ID="Warrior">
   <!-- A warrior is a game character -->
   <rdf:type rdf:resource="#Character"/>

   <hasHP rdf:datatype="xsd:byte">35</hasHP>
   <hasMP rdf:datatype="xsd:byte">0</hasMP>
   <hasDP rdf:datatype="xsd:byte">5</hasDP>
   ...
   <hasLevelNumber rdf:datatype="xsd:byte">1</hasLevelNumber>
</rdf:Description>
```

We used both the Jena implicit reasoner and the Pellet one. While trying to list classes and subclasses from our ontology in Jena, we encountered a problem. The ontology seemed to be too big or complex to process and with the default reasoner we got the results in approximately 30 seconds (unacceptable in a real game). When we imported Pellet, we got the inferred results in about 2 seconds. For our ontology, Pellet seems to be a more powerful and faster reasoner.

5. Conclusions and Further Work

This paper demonstrates the integration of an ontology in an RPG game using the Jena framework. By being able to define game rules ontologically and reason with the instances created against that ontology, our proposal could be view as an encouraging approach of using semantic Web technologies in the context of computer games, as an alternative method for the classical AI methods.

In the near future, we intend to extend our research by continuing to expand the ontology, to study certain complexity aspects, and creating a full working game with it.

References

[1] Allemang, D., Hendler j. *Semantic Web for the Working Ontologist*, Morgan Kaufmann, 2008

[2] Baader F., Horrocks, I., Sattler, U. "Description Logics", *Handbook of Knowledge Representation*, Elsevier, 2007

[3] Gennari, J. et al. "The Evolution of Protégé: An Environment for Knowledge-Based Systems Development", *International Journal of Human-Computer Studies*, Volume 58, 2002

[4] Horrocks, I. "Ontologies and the Semantic Web", *Communications of the ACM*, Volume 51, Issue 12, 2008.

[5] Machado, A., Amaral, F., Clua, E. "A Trivial Study Case of the Application of Ontologies in Electronic Games", *Proceedings of the VIII Simposio Brasileiro de Jogos e Entretenimento Digital*, Sociedade Brasileira da Computaçao, 2009

[6] Peinado, F., Gervas, P., Daz-agudo, B. "A Description Logic Ontology for Fairy Tale Generation", *Forth International Conference on Language Resources and Evaluation: Workshop on Language Resources for Linguistic Creativity*, 2004

[7] Menkovschi, V., Metafas, D. "AI Model for Computer games based on Case Based Reasoning and AI Planning", *Proceedings of the 3rd International Conference on Digital Interactive Media in Entertainment and Arts*, ACM Press, 2008

[8] Mepham, W. "Semantically enhanced games for the Web", *Proceedings of the WebSci'09: Society On-Line*, Athens, Greece, 2009

[9] Sanchez Ruiz-Granados, A. et al. " Game AI for a Turn-based Strategy Game with Plan Adaptation and Ontology-based Retrieval", *Proceedings of the ICAPS-07 Workshop on ICAPS 2007 Workshop on Planning in Games*, AAAI Press, 2007

[10] Sirin, E. et al. "Pellet: A Practical OWL-DL Reasoner", *Web Semantics: Science, Services and Agents on the World Wide Web*, Volume 5, Issue 2, Elsevier, 2007

[11] Zolin, E., *Complexity of Reasoning in Description Logics* – Available online at `http://www.cs.man.ac.uk/ ezolin/dl/`

[12] * * *, *Game Ontology Project* – Available online at `http://www.gameontology.org/`

[13] * * *, *Jena – A Semantic Web Framework for the Java Platform* – Available online at `http://jena.sourceforge.net/`

Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iasi, Berthelot Street, 16 – Iasi, Romania

E-mail address: `busaco@info.uaic.ro`