Gameplay semantics for authoring adaptivity in mobile games

Ricardo Lopes
Delft University of Technology
The Netherlands
r.lopes@tudelft.nl

Ken Hilf
Carnegie Mellon University
Pittsburgh, PA, USA
kenhilf@gmail.com

Luke Jayapalan
Carnegie Mellon University
Pittsburgh, PA, USA
ljayapal@alumni.cmu.edu

Rafael Bidarra
Delft University of Technology
The Netherlands
r.bidarra@tudelft.nl

(a) (b)

Figure 1: In *(a)*, player looses its balance on the rolling snowball, due to steep slope. Later on, slope difficulty has decreased, and player jumps over an obstacle the snowball picks up *(b)*

1. INTRODUCTION

Using procedural content generation (PCG) in game development can contribute to further engage players in more fun, balanced and richer gameplay. An adaptive game can achieve this by: (i) distinguishing its players' interaction [3], and (ii) accordingly adapting game elements, using PCG [4].

We investigated adaptive mobile PCG methods with a case study on *7's Wild Ride*[1], a game developed at the Entertainment Technology Center of Carnegie Mellon University. We changed the original game to include a player-dependent PCG-based dynamic difficulty adjustment (DDA), and concluded that it indeed keeps the game evenly *balanced* for anyone, contributing to attract and retain the typical wider audience of casual mobile players. In this paper, we describe the role that *gameplay semantics* have in supporting this DDA mechanism, specifically by allowing designers to control PCG methods directly, thus effectively authoring adaptivity.

2. LEVEL GENERATION

7's Wild Ride is a side-scrolling platform game for mobile devices, where players have to prevent the main character from falling off a rolling snowball. They keep the character alive by: (i) tilting the mobile device left and right to counteract the gravity effects of navigating slopes (balance), and (ii) jumping on the snowball over obstacles that it picks up.

In our case study, a level generator and a player model were implemented to support the on-line creation of game content. The generator retrieves slope segments and obstacles from a content library and combines them at runtime, as the player advances through the level. Input difficulty values steer the generator search process, since each piece of content is accordingly classified. These input values are provided by the player model. Player performance is evaluated by the player model and converted into individual desired difficulty scale values for specific skills. For example, if you repeatedly cannot balance the character (tilting), a lower balance difficulty is issued, which constrains the generator to sequentially select segments with easier slope combinations. This way, the game's difficulty is adapted to the individual player skill. More details on the level generator and the player model were described elsewhere [2]. Fig. 1 illustrates an example of the DDA mechanism in action.

3. GAMEPLAY SEMANTICS AND DDA

Semantics is all information about the world and its objects, beyond their geometry (*e.g.* attributes, interrelationships). Our semantic library *Entika* [1] is a hierarchical class (relational) database responsible for storing all semantic game entities. Game designers use this library to specify semantics, atop geometry, in a game world and its content, in a generic and reusable way.

Gameplay semantics is all information which captures the gameplay value of all game world entities. It can express, for example, affective experiences, player performance, player actions or game actors. Importing gameplay semantics and embedding them in game worlds (at runtime) can be used to steer online PCG methods.

For our *7's Wild Ride* case study, we used gameplay semantics to support the DDA mechanism. Using *Entika*, we designed a semantic layer mapping game content to difficulty. Semantic entities for slope segments and obstacle entities can associate geometry models (Unity prefabs) with their difficulty scale values. These entities support a new difficulty-based semantic scheme: (i) each entity has an associated difficulty scale value, and (ii) each level segment has various locations suitable for obstacle placement, each classified with its own difficulty value. The level generator gets the desired difficulty scale input values from the player model and retrieves, from the semantic library, authored game content with semantics

[1] a video of our research is available at http://graphics.tudelft.nl/~rval

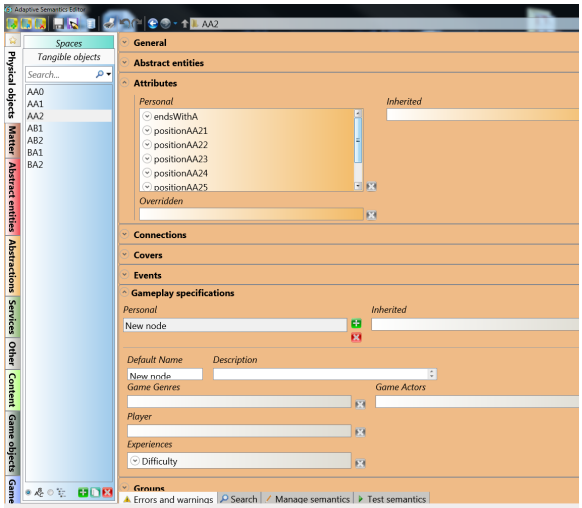

Figure 2: *Entika*: specifying valid positions and difficulty value for a level segment

that matches those inputs. The combination of this semantics-based content results in a dynamic level where retrieved slopes are placed sequentially and obstacles are independently instantiated each slope. Such semantics-based PCG supplies players with generated content which matches their skills, as pre-specified by designers.

Gameplay semantics allows designers to participate in authoring the DDA mechanism without having to program. More importantly, it holds the expressive power to directly specify the building blocks and the selection rules which support level generation. Designers can create slope segments and even combine them to form groups with specific semantics. They can express constraints to the generation process, *e.g.* which segment can(not) follow which segment. Entities for obstacles can be created independently from the specification of valid level locations for their placement. And each of these entities and locations can be associated with numerical values for difficulty scales for obstacle placement or slope selection. The level generator uses all this information to synthesize (on-line) an adaptive level. With gameplay semantics, designers can even author DDA in a iterative way. Any corrections and changes to the semantic library directly tweak the level generator behavior. Fig. 2 illustrates gameplay semantics specification on *Entika*.

To the best of our knowledge, this is the first time a semantics-based adaptive approach is applied in the mobile device segment. Android OS and the Unity game engine allowed quick integration with the technology supporting the persistent semantic library, SQLite and C#. We developed a light-version of a semantic engine, which is able to import and query semantic content from the database, as the game is running, on a mobile device.

4. RESULTS

In order to assess if gameplay semantics, as authored by designers, is expressive enough to support an effective DDA mechanism, we evaluated the success of our DDA mechanism in *7's Wild Ride*, with 22 participants (children between 6 and 12). In a play test session, performance data was logged and short interviews were conducted. Before each game session (3 minutes), players had to complete a tutorial to learn the basic game mechanics: balancing and jumping.

Table 1 summarizes the answers of our interviews. We asked all participants to rate the *challenge* they felt throughout the game, from 1 (less) to 5 (more). We also asked them a similar question about the *unfairness* of the game's progression wishing to assess whether the DDA mechanism was effective in providing the most appropriate balance, in a non-obtrusive way. Being children, this concept was hard to explain. Therefore, this question was posed with a more negative connotation, in order to assess any frustration (or satisfaction) and injustice.

Table 1: Questionnaire rates for *challenge* and *unfairness*

	1(less)	2	3	4	5(more)
Challenge	0%	0%	55%	41%	4%
Unfairness	9%	45%	23%	15%	9%

Results for challenge showed that the game is *balanced* (rate 3), an indicator of a successful DDA mechanism. Results for fairness seem to show that the majority of the participants found the game either fair and satisfying (rate 1,2) or balanced (rate 3). The negative assessment nature of this question further validates our previous findings: again, the game seems *balanced*.

Furthermore, by correlating these answers with logged player data, we could see that the players which rated a higher degree of challenge did so because they progressed more and, as such, achieved higher values for the difficulty scales. Correlating the fairness answers with these player categories (with progression and without progression) allowed us to observe that although gradual difficulty progression seems to imply a higher degree of challenge, that actually leads to a higher degree of satisfaction. More details on these correlations can be found in [2].

5. CONCLUSIONS

From these results, we conclude that our semantics-based DDA mechanism can adjust the game to different player categories, keeping gameplay balanced and players satisfied. The effectiveness of the DDA mechanism allows us to conclude that gameplay semantics, as *e.g.* that supported by *Entika*, has expressive power enough to allow game designers to control DDA and PCG, in an adaptive mobile game.

Using gameplay semantics, designers can effectively author adaptive PCG in mobile games, being actively involved in the game design and content creation loop. Semantics provides, therefore, a powerfull method to deploy important gameplay-based knowledge in the mobile game platform, an environment where additional player knowledge and adaptation can have an increased importance, to attract and retain the wide audience of casual mobile players.

6. REFERENCES

[1] J. Kessing, T. Tutenel, and R. Bidarra. Designing semantic game worlds. In *Proceedings of the third workshop on Procedural Content Generation in Games (PCG 2012)*, Raleigh, NC, USA, May 2012.

[2] R. Lopes, K. Hilf, L. Jayapalan, and R. Bidarra. Mobile adaptive procedural content generation. In *Proceedings of the fourth workshop on Procedural Content Generation in Games (PCG 2013)*, Chania, Crete, Greece, May 2013.

[3] A. M. Smith, C. Lewis, K. Hullett, G. Smith, and A. Sullivan. An inclusive view of player modeling. In *6th International Conference on Foundations of Digital Games*, Bordeaux, France, July 2011.

[4] G. N. Yannakakis and J. Togelius. Experience-driven procedural content generation. *IEEE Transactions on Affective Computing*, 99:147–161, 2011.