Generating Non-linear Narrative for Serious Games with Scenario Templates

Koen Samyn[1], Gaétan Deglorie[2], Peter Lambert[2], Rik Van de Walle[2] and Sofie Van Hoecke[2]
[1]*DAE, University college of West Flanders, Ghent University Association, Botenkoperstraat 2, B-8500 Kortrijk, Belgium*
[2]*Multimedia Lab, ELIS, Ghent University-iMinds, Gaston Crommenlaan 8 Bus 201, B-9050 Gent, Belgium*

Keywords: Serious Game, Domain-specific Modeling Languages.

Abstract: Complex social interactions between NPC's and players in a serious game remains a challenge. Standard game engines are focused on a game world with a fixed set of rules and linear gameplay. The problem is compounded by the unfamiliarity of many researchers with the basic structure and technicalities of a serious game. In this paper we propose a method to use linear scenario elements as templates for the generation of a non-linear narrative. We implement this method by creating an extra layer on top of the existing ATTAC-L modeling language which is a tool for developing virtual interactive scenarios. Users are thus presented with a language that offers limited flow control and simplifies the authoring process for the creation of scenario elements. Our method uses these existing scenario elements together with metadata, and fills in the templated elements with the contextual information of the current game state. By separating scenario concerns from non-linear narrative concerns, we hope to make it easier to develop interesting non-linear serious games that still conform to the requirements of evidence-based serious game research.

1 INTRODUCTION

Non-linear gameplay is a concept that is high on the wishlist for a lot of game developers. However, an example of real non-linear narrative is hard to find in the wide range of commercial or serious games. This should not come as a surprise because there are many hurdles to take before linear narrative offers the necessary consistency in terms of game logic and immersion. Every bifurcation in the story results in a doubling of the amount of scenarios that are needed to define the game. Needless to say that a naive approach towards the implementation of non-linear narrative leads to a huge amount of work, and ultimately to compromises that reduce the combinatorial explosion.

In this paper, we propose a solution for a non-linear narrative that is built on top of ATTAC-L (Broeckhoven and Troyer, 2013). ATTAC-L is a domain language for defining virtual scenarios that is being developed as part of the Friendly Attac project. The Friendly Attac project studies and develops an innovative serious game to help youngsters deal with cyberbullying issues. This is done by allowing youngsters, through the use of the virtual scenarios, to experience different roles (bully, victim, or bystander) in cyber bullying incidents during the game, to react to those experiences, and to get adjusted feedback based on their individual reactions. This will increase their empathy, enhance their social skills or teach/train them relevant coping strategies. ATTAC-L offers the user (domain experts) a set of graphical building blocks that can be assembled to create a simple scenario. An example of a simple ATTAC-L scenario is presented in figure 1. In this example, a player sends a tweet to Kate with the message 'Hi'. In the ATTAC-L language, an action or sentence can be defined by using a simplified grammatical structure, in which a **subject** can declare the execution of a **verb**. ATTAC-L also allows the definition of a choice structure (where the user can pick one of the provided options), and the definition of synchronous actions, where all the actions are executed at the same time.

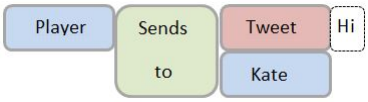

Figure 1: Attac-L example.

A recent implementation of ATTAC-L for serious games (Janssens et al., 2014) translates these graphical building blocks into an XML declarative language

Samyn K., Deglorie G., Lambert P., Van de Walle R. and Van Hoecke S..
Generating Non-linear Narrative for Serious Games with Scenario Templates.
DOI: 10.5220/0005363705230530
In *Proceedings of the 10th International Conference on Computer Graphics Theory and Applications* (GRAPP-2015), pages 523-530
ISBN: 978-989-758-087-1
Copyright © 2015 SCITEPRESS (Science and Technology Publications, Lda.)

which is interpreted within a serious game engine. This implementation provides support to test ATTAC-L scenarios in isolation and also to integrate the scenarios in a serious game.

The structure and syntax of the ATTAC-L language is simple as to facilitate the creation process of scenarios for users who are not familiar with game design. However this simplicity also restricts the range of possible scenarios and makes it very cumbersome to introduce non-linear game play. Some notable limitations of ATTAC-L are the lack of state management and repetition. To overcome these limitations, we introduce the concept of a **story engine**. The story engine selects a scenario template from a scenario database based on the current state of the player model and provides the template with contextual information from the game. The story engine is also responsible for maintaining the integrity of the storyline which is defined by additional metadata in the templates.

Performance and change objectives (Desmet et al., submitted) are an additional consideration for the design of a serious game. In an evidence-based serious game founded on the intervention mapping protocol (Bartholomew et al., 2011), performance objectives are the behaviours we would like to obtain, whereas change objectives are defined as the determinants that influence these target behaviours. In our proposal it is possible to link performance objectives and change objectives to a scenario template. This linking process makes it possible to track the progress of the player throughout the duration of the game and to present the player with the scenarios that are needed to improve his/her proficiency in the subject matter.

The remainder of this paper is as follows. In the next section we discuss the related work. In section 3 we give an overview of the concepts of non-linear storytelling and game play adaptation. In section 4 we delve deeper into the technical implementation of our framework. In section 5 we discuss the adaptability of the game play that allows us to present the player a game environment that is tailored to his/her current skill set. In section 6 we present a testing framework that validates the selection of scenario templates for a number of player types. Finally, we present the conclusions and future work in section 8.

2 RELATED WORK

In (Prensky, 2005) the author makes the case that digital game-based learning should maintain the balance between learning and (fun) gameplay. Although we present a method for creating interactive and non-linear stories and not the actual creation of a game, this balance must be kept in mind when determining the feature set of the story engine. Another important consideration is the payoff versus reward system which indicates that players are prepared to put in the work if the reward is sufficient.

In (Greitzer et al., 2007) a cognitive model is presented that suggests that learners/players benefit from a layered and non-linear approach to learning. This approach was applied for the serious game CyberCIEGE by clustering scenarios in **layers** with greater scenario complexity in each successive layer. We present a similar approach but cluster the scenarios in terms of performance or learning objective. The CyberCIEGE game does not adopt an adaptive player model, however we follow some of the recommendations of the authors for our approach (see section 5 with regards to player driven adaptation of the narrative).

In project Muse (Llobera et al., 2013) a method is presented that allows content creators to create an interactive drama (ID) by adopting the concepts of providence and the Zelig interaction metaphor, which means that non-player characters can automatically take on roles that are necessary for the advancement of the storyline. Our presented aliasing approach (see section 4.1) is similar but introduces additional conditions on the role taking process.

3 OVERVIEW

An overview of the story engine is presented in figure 2.

When the game requires the start of a new scenario, a scenario is selected from a database of scenario templates. The selection process takes the current state of the player model into account, including the previous actions and the current location of the player. The scenario might also require the proximity or availability of non-player characters (NPC's). Furthermore the scenario might define the characteristics (traits and facets, see section 4.1) of these NPC's.

Once a scenario is selected, the aliases in the scenario template are replaced with the appropriate and currently available NPC's and/or items. The scenario template can require an NPC, with a predefined set of personality characteristics, and/or a specific state from the player model. An example scenario template can define an alias for a character A with a personality that contains the bully trait, and an alias for a character B that is a victim of bullying. The aliases A and B are then dynamically replaced with the selected

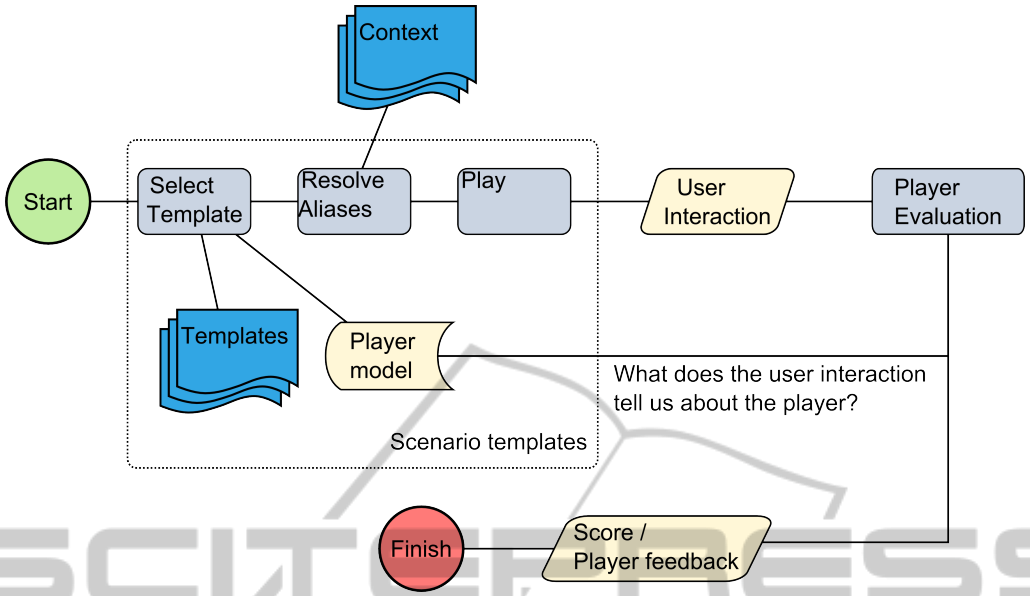

Figure 2: Story engine overview.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS

NPC's by querying the context of the game.

Next the actions in the scenario are played or executed, and possible options are presented to the player. The decision of the player will trigger an evaluation and leads to an update of the player model. The player model is used to adapt the **probability** that a given scenario template will be selected, given the performance objective to which it belongs.

4 SCENARIO TEMPLATES

In our implementation, a story is composed of a hierarchical structure of scenario templates. A scenario template defines via metadata the circumstances, under which a scenario can be selected and executed, and the requirements for the NPC's that will take part in the scenario. Finally, the scenario template includes the actions of the NPC's and the possible options for the player. The following subsections explain this in further detail.

4.1 Aliasing Approach

A central part of the scenario template is the definition of an alias. An alias acts as a **placeholder** for NPC's or items. When a scenario template is selected for execution, the alias is replaced with a concrete NPC in the game. An example of an alias can be found in listing 1.

Listing 1: Alias example.

```
<alias>
  <select type="NPC" id="CharacterA">
    <trait type="victim"
      value="80%" op="higher"/>
    <facet type="student" value="true"/>
  </select>
</alias>
```

In listing 1 a game object with type NPC is selected. Within the selector it is possible to add multiple criteria for this NPC. It is possible to select for a certain **personality trait** of the NPC by specifying the type of the trait (in this case victim), a percentage value of the trait (in this case 80%) and an operator. In the example an NPC with a victim trait higher then 80% is selected. Additionaly a **facet** of the NPC can be selected with the facet element, in this case the NPC must be a student.

In the remainder of the scenario template, this NPC can be referenced by its alias **CharacterA**. This is shown in listing 2.

Listing 2: Alias useage.

```
<gamemove subject="player" id="gm1">
  <verb type="sendto">
    <mediaObject object-class="sms">
      <who id="CharacterA"/>
        <property key="content"
          value="Hi ${CharacterA.name}!"
        />
        <ref id="sms1"/>
    </mediaObject>
  </verb>
</gamemove>
```

In listing 2 a special **subject** (the player) sends a message to the `CharacterA` alias. Within the message, an escape sequence can be used to query to properties of the alias. When the scenario is played the escape sequence `${CharacterA.name}` will be replaced by the name of the selected NPC.

Finally it is possible to **propagate** an alias to other scenarios by defining another query within the select tag, as shown in listing 3

Listing 3: Propagation example.

```
<alias>
  <select type="NPC" id="CharacterA">
    <adopt id="scenario1"/>
  </select>
</alias>
```

The `adopt` tag instructs the `select` tag to adopt the same NPC as the one that was selected for `scenario1`. A prerequisite for this tag to work is that `scenario1` must be executed before the current scenario. This can be enforced by the use of requirement meta-data as explained in the next section.

4.2 Storyline Control

The scenario template system enables the reuse of story elements in the larger context of the game. The story engine also provides several possibilities to control the order of the scenario elements, as defined in the metadata of the scenario template.

A first important metadata property is the replayability of a scenario template. An example of a replayable type of scenario template is a **help** scenario template. In this case the scenario template requires an NPC with a **victim** personality type that has lost an item. The player might opt to help the NPC or to ignore the NPC. Listing 4 shows an example of a scenario with a replay meta tag.

Listing 4: Help NPC.

```
<scenario id="help_npc">
  <metadata>
    <property key="replayable"
      value="true"/>
  </metadata>
  <select type="NPC" id="CharacterA">
    <trait type="victim"
      value="80%" op="higher"/>
    <facet type="student" value="true"/>
  </select>
  <eventpart type="options">
    <option id="1" score="2">
      Help ${CharacterA.name}
    </option>
    <option id="2" score="2">
      Ignore ${CharacterA.name}
    </option>
  </eventpart>
</scenario>
```

Another possibility is to require the (positive) execution of a scenario template A before scenario template B can be executed. In the help quest example (named `help_npc`) helping the victim is required to open up the execution of another scenario template named `explore_quest`. An additional feature is that the `explore_quest` scenario can require the same NPC as the `help_npc` scenario.

Listing 5: Explore quest.

```
<scenario id="explore_quest">
  <metadata>
    <property key="replayable"
      value="false"/>
    <require>
      <scenarioref id="help_npc"/>
    </require>
  </metadata>
  <select type="NPC" id="CharacterA">
    <adopt id="help_npc"/>
  </select>
  <!-- npc actions ... -->
</scenario>
```

In listing 5 the `scenarioref` tag defines the scenario that is required before this scenario can be started. Of course it is possible to define multiple scenarios as **required**.

The `explore_quest` scenario in the same listing can be executed unconditionally. However, if the player did not choose the help of the NPC in the `help_npc` scenario, the execution of the `explore_quest` is inconsistent. To remedy this inconsistency a new metadata tag can be used, as shown in listing 6, where the explore quest has been updated with an additional **condition**.

Listing 6: Explore quest with condition.

```
<scenario id="explore_quest">
  <metadata>
    <property key="replayable"
      value="false"/>
    <require>
      <scenarioref id="help_npc"/>
    </require>
    <conditions>
      <faction
        idA="CharacterA"
        idB="player"
        value="friend"/>
    </conditions>
  </metadata>
  <select type="NPC" id="CharacterA">
    <adopt id="help_npc"/>
  </select>
  <!-- npc actions ... -->
</scenario>
```

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

With a **faction** tag it is possible to specify that the two characters, as defined by the attributes `idA` and `idB` must be friends with each other. The **player** value is a special value that references the player and the **CharacterA** value is defined by the select tag in the same scenario. In this case the type of the condition was set to **faction** but it is also possible to query specific attributes of an NPC, or to check the inventory of a character or player. Some of these possibilities are shown in listing 7.

Listing 7: Conditions.

```
<conditions>
  <inventory
    id="player"
    item="key#731"
    value="true"/>
  <attribute
    id="CharacterA"
    attribute="health"
    value="50%"
    op="higher"/>
</conditions>
```

The **trait** and **facet** selectors, defined in section 4.1, are valid within the condition tags. The **faction**, **inventory** and **attribute** tags are also valid within a **select** tag.

5 PLAYER-DRIVEN ADAPTATION

Serious games are used to teach certain behavior or knowledge to users. To provide an optimal learning experience, the teacher should adapt to the respective student, e.g. by taking more time to thoroughly explain the concepts that the student has trouble with. However, the average video-game does not adapt gameplay to each individual player. Typically, one constructs some kind of player-model based on real-time player input or post-game feedback. The realized model can then be used to tweak certain parameters of the game (e.g. generating different level geometry to increase challenge for skilled players). In an educational context, learning state can be checked through performance or learning objectives. We use these performance objectives to track the player's learning state in real-time and adapt the occurrence of learning challenges in order to better suit the problem areas of player.

In the next subsections we expand on performance objectives, extracting the player model and probability distribution of template selection respectively.

5.1 Performance Objectives

A serious game based on the Intervention Mapping Protocol needs to define change and performance objective, and link them to the correct scenario template. **Performance objectives** define the desired behavior for the player in the game world but mostly in the real world. A game with cyberbullying as topic can for example define positive bystander behaviour for a player which means that the player has to disapprove of bullying behaviour in a visible and public manner. **Change objectives** define the underlying determinant for the behavior. It may be that, in order to disapprove of bullying in a public way, the player needs to expect that this will end the bullying without putting himself at risk. Another change needed in the determinants could be that the player has to believe the victim is not to blame for being bullied.

In our implementation a scenario template can address one performance objective to induce the desired behavior together with one (typically) or more change objectives. This information is stored into the metadata of the scenario template. The scenario templates can then be clustered by performance objective which enables the evaluation of the player and the player-driven adaptation of the storyline.

5.2 Player Evaluation

To predict the player model, we assume that a player has a certain proficiency with each performance objective. We map this in range of 0 to 1, where 0 states that the player does the opposite of what the performance objective requires, 0.5 that the player acts randomly (i.e. neither in favour nor in opposition of the performance objective) and 1 that the player fully acts in accordance with the performance objective.
Our concept is to predict the player model during gameplay based on the actions performed by the player. Currently the actions that can be performed are of a binary nature, acting either for or against the performance objective. We start by assuming that the player has a 50% proficiency in all performance objectives (i.e. starting out without a bias). For each event, the action is checked to be for or against its performance objective. If it's for, we simply increase the proficiency for that objective by a certain value. If it's against, we decrease the proficiency.
This increase/decrease value, named adaptation rate from here on, is changed according to the current proficiency level (see Figure 3). For the mapping of proficiency to adaptation rate, we chose a bell curve (modelled as 2 sigmoid-functions) because of following reasons:

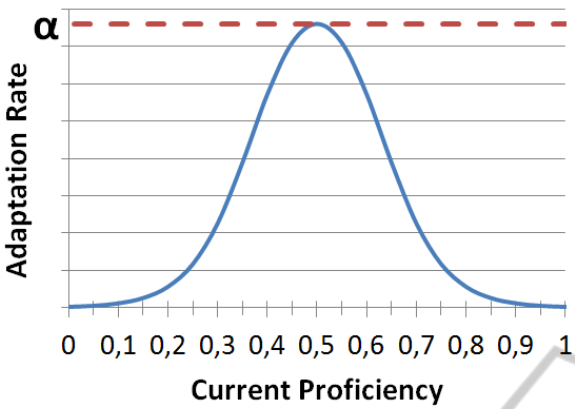

Figure 3: Adaptation rate as a function of proficiency.

- A high adaptation rate near the center makes the proficiency go in either direction fast and thus allows the system to quickly revert to 50% when evaluations alternate (i.e. avoids creating an unnecessary bias).
- A progressively lower adaptation rate near the ends of the proficiency spectrum creates a barrier so that truly mastering a performance objective does not come too easy.
- Making the curve flatter near the ends of the proficiency spectrum creates a barrier for return. When having achieved the far end of the spectrum for an objective, we don't want the proficiency to return in the opposite direction for the slightest opposite action. For example having a mastered an objective, the player's proficiency should not drop significantly for a single mistake once and a while. Similarly having failed an objective, the proficiency should not go up significantly if the player happens to make a correct 'guess' once in a while. Only if the player consistently makes correct choices, the proficiency should change accordingly.

The peak of the curve is controlled by the learning rate parameter α. An evaluation using our prediction model and the current error-rate will be displayed in the results section.

5.3 Probability Distribution of Template Selection

The scenario templates are grouped according to performance objective. During gameplay, whenever a new scenario has to be loaded, a new template is selected at random. This is achieved by first selecting a performance objective or pool (a set of templates with the same performance objective) at random and

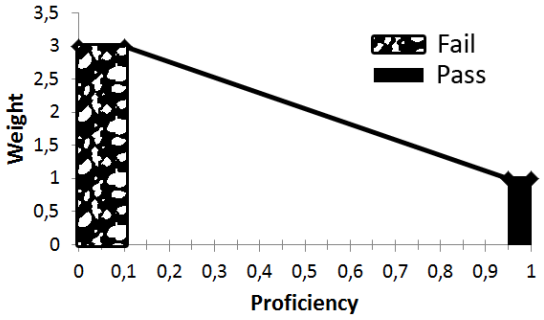

Figure 4: Proficiency to weight mapping.

from this pool in turn randomly selecting a template. To adapt the narrative to the player, we change the probability distribution of the performance objective pools to better match the problem areas of the player. As discussed above the predicted model attempts to estimate the current proficiency of the player. Each proficiency matches a certain performance objective and is expressed as a number between 0 and 1. From these proficiencies we will automatically deduce the appropriate probability distribution.

In order to inhibit the possibility of creating inappropriately large or inappropriately small weight values when using a simple formula for the weights (with proficiency as input), we introduce thresholds to create weight-plateaus (see Figure 4) and avoid performance objectives with a probability of nearly 0% or nearly 100%. If a proficiency ever enters one of these plateaus, the performance objective will be flagged as such (either 'fail' or 'pass' can be used to automatically detect noteworthy events for a supervisor/teacher, especially in the case of 'fail'. Between the thresholds, the weight value are linearly interpolated, creating a probability distribution that is never in favour or disfavour of a single performance objective. Selecting the scenarios using this methodology will hopefully lead to a more balanced game/learning experience.

6 RESULTS

To validate our concept we ran several simulations of our system. The system uses 5 performance objectives, marked as PO1, PO2, PO3, PO4 and PO5 respectively. Each performance objective has a pool of 30 scenario templates, making a total of 150 templates. A single run of the system triggers 100 scenarios, predicting the player model after each event and adjusting the probabilities accordingly. All result data is the average of 1000 runs of the system.

A digital player model was created to simulate all

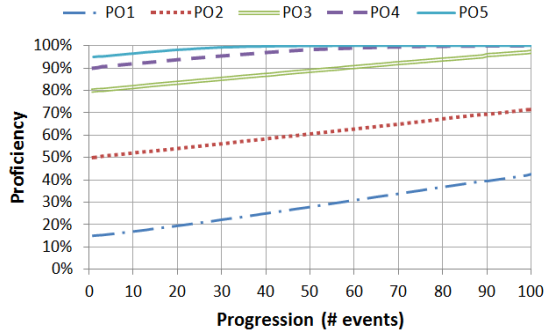

Figure 5: Simulated player proficiency/progression.

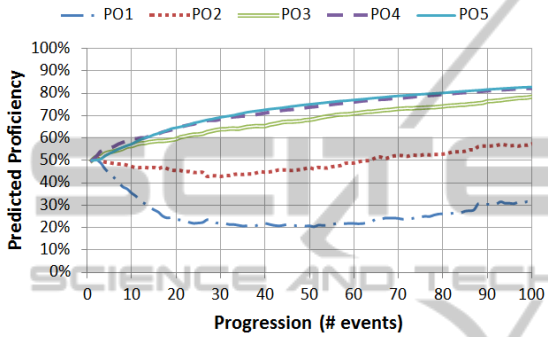

Figure 6: Predicted player profiency over time.

interactions. The player is modelled to have proficiency in all 5 performance objectives as has been described in subsection 5.2. Table 1 displays the proficiency of the player for each performance objective and the matching base learning rate or α-value. In our current model we assume that each time the player interacts with a scenario he/she learns from it, represented as a small increase in proficiency (i.e. learning rate).

Table 1: Player statistics.

Player	PO1	PO2	PO3	PO4	PO5
Starting Profiency	0.15	0.5	0.8	0.9	0.95
Learning Rate	0.01	0.01	0.01	0.01	0.01

The proficiency of the simulated player is tracked over time, as can be seen in Figure 5. Not surprising, proficiency in all objectives rises. From the interaction with the scenarios the story engine tries to predict the player's proficiencies. For the adaptation rate we chose an α-value of 0.15. If the α-value is too low, the prediction will reach the actual proficiency very slowly; if the α-value is too high, the prediction will become unstable.

The predicted player model over time is displayed in Figure 6. The prediction takes about 20 events to get a good approximate measure of the player's proficiencies. The predicted proficiencies are all lower than the actual proficiencies off the simulated player.

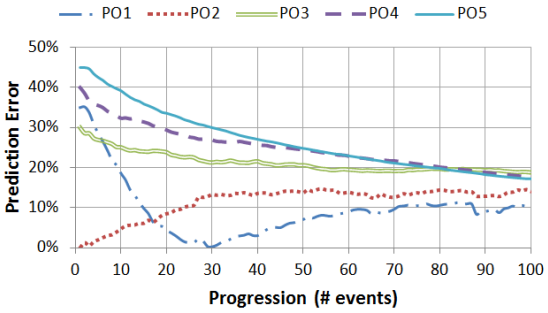

Figure 7: Prediction error over time.

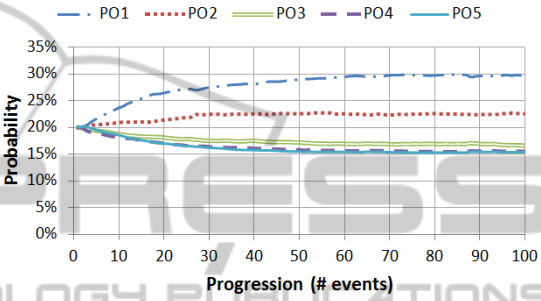

Figure 8: Adaptation of performance objective picking probability.

Figure 7 displays the absolute difference between the predicted and actual proficiencies, i.e. error rates. The average error rate drops to about 15%, this is a reasonable value as a difference of under 20% would not yield a significant change in the resulting weight of a performance objective, i.e. no major differences in the resulting probability distribution.

Finally, the resulting weights of each performance objective change the overall probability distribution of template selection (see Figure 8). This new distribution shows that the story engine creates a personal game/learning experience without going into extremes.

7 DISCUSSION

We reduced the technical complexity of writing scenarios for serious games by separating the modeling scenario template language from the serious game implementation. By decoupling the modeling and implementation, a writer can develop a scenario with only an abstract knowledge of the game world, and does not need to concern him- or herself with the technical details of the game engine. A disadvantage of course is that every character, location and item described in the scenario template needs to be provided in the game world.

The player model introduced in section 5 offers

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

the possibility to test scenario templates in terms of change and performance objectives. The current experimental learning rates will, however, need to be refined and validated during user testing.

A potential problem with the current implementation of the player-driven adaptation lies with antagonistic players. If a player selects the wrong option at every turn of the scenario the proficiency level for all performance objectives will drop and the scenario will not advance. It is out of scope of our implementation to impose a definitive solution to this problem. Within the current framework the player will have to work hard to recover from this situation, which might extend the duration of the game over practical limits.

In most games antagonistic play is avoided by providing the players with rewards if they make sufficient progress. This mechanism helps to retain players and maintain player interest, however for a serious game an even higher retention rate is required. Most serious games are played under supervision so there is an opportunity to intervene in the case of antagonistic game play. At any rate we provide a signal that a player is deliberately failing the game and the decision on how to act upon this information lies with the (pedagogical) domain experts.

8 CONCLUSIONS AND FUTURE WORK

We presented the concept of a story engine to introduce non-linear narrative using scenario templates and aliasing. The aliasing approach facilitates the design process by using simpler building blocks (scenario templates) as a way to create a dynamic experience with an individualized storyline inside the game. Specific to the topic of serious games and learning is the creation of a player-driven narrative that tests the player on his/her skills. This was achieved through the mapping of performance objectives to each scenario template using metadata and by creating a model of a player that is adapted towards.

As for future work, we foresee improvements in several areas. The learning rate for the simulation is set to a low and possibly unrealistic value. By comparing actual input from a player to our player model we should be able to derive a better player and game specific value. Further study is required to create an algorithm that can deduce an approximate learning value based on the user input.

Player evaluation is performed in a binary fashion, to allow for a more diverse way of action we would include a choice based system. This system would give the player a set of choices (actions) to perform based on an occurred event.

Our implementation is used to randomize the occurrence of unrelated events. However, to provide a controlled narrative while still using such a system, linked templates with smart selection are required. In such a system the pool of available event templates would be populated only by those events that can follow the previous one, i.e. an event that should logically follow from a choice made by the player or a completely unrelated event.

Our work focuses on the aliasing aspects of characters. However, a story may also require geographic bifurcations. E.g. the main character could choose a different path to reach the destination. In this case a scenario would not only contain aliases for characters but also for geographic locations (for example hidden caves, treasures, bridges, ...).

ACKNOWLEDGEMENTS

This work was funded by the IWT SBO Friendly ATTAC project (http://www.friendlyattac.be/).

REFERENCES

Bartholomew, L. K., Parcel, G. S., Kok, G., Gottlieb, N. H., and Fernandez, M. E. (2011). *Planning health promotion programs: an intervention mapping approach*. John Wiley & Sons.

Broeckhoven, F. V. and Troyer, O. D. (2013). Attac-l: A modeling language for educational virtual scenarios in the context of preventing cyber bullying. In *SeGAH*, pages 8. IEEE.

Desmet, A., Van Cleemput, K., Bastiaensens, S., Poels, K., Vandebosch, H., Verlogine, M., Vanwolleghem, G., Mertens, L., and De Bourdeaudhuij, I. The stepwise development of a serious game to mobilize bystanders against cyberbullying among adolescents the friendly attac project. submitted.

Greitzer, F. L., Kuchar, O. A., and Huston, K. (2007). Cognitive science implications for enhancing training effectiveness in a serious gaming context. *J. Educ. Resour. Comput.*, 7(3).

Janssens, O., Samyn, K., Van Hoecke, S., and Van de Walle, R. (2014). Educational virtual game scenario generation for serious games. In *SeGAH*, pages 8. IEEE.

Llobera, J., Blom, K. J., and Slater, M. (2013). Telling stories within immersive virtual environments. *Leonardo*, 46(5):471–476.

Prensky, M. (2005). Computer games and learning: Digital game-based learning. *Handbook of computer game studies*, 18:97–122.