Simulating Adaptive Quests for Increased Player Impact in MMORPGs

Emmett Tomai and Rosendo Salazar

University of Texas – Pan American, 1201 W. University Dr. Edinburg, TX, 78539
tomaie@utpa.edu

Abstract

In this paper, we present adaptive quests, an extension to the dominant quest model that guides and motivates gameplay in MMORPG shared worlds. The standard model has proven effective, but is significantly incompatible with the desire for player driven change in the world. We present an incremental step to increasing player impact, discuss the problems it creates with the quest model, and show how adaptive quests can help reconcile the two. We present simulation experiments supporting not only that adaptive quests help mitigate those problems, but that they can actually improve them over the standard model.

Introduction

The MMORPG genre has seen dramatic growth in the past decade. But while many diverse designs exist, the explosive popularity of first Sony's *EverQuest* and then Blizzard's *World of Warcraft* has entrenched a dominant model. One of the core elements of this model is *quest*-driven gameplay. In brief, players control avatar characters in a shared, persistent world. They are free to roam in the world, interacting with the environment and other players as they see fit. Throughout the world there are system-controlled entities (*Non-Player Characters or NPCs*) that the player can interact (speak) with to receive quests. These quests specify task requirements (e.g. kill 10 rats) and rewards (e.g. progress points and a shiny hat), and provide narrative text intended to situate and motivate the task (e.g. help us, the rats are eating all our food!). Multiple quests can be accepted, worked on, put aside, abandoned or completed at the players' discretion. This non-linear, on-demand system of guidance and motivation has proven remarkably effective, in spite of text that is often trite and formulaic, and tasks that are highly repetitive.

The quest model brings a significant limitation: it forces the shared world to be mostly unchanging. After a player finishes a quest, they are told via discourse how their actions helped the in-game characters. But the world cannot change, or the next player to come along wouldn't be able to perform the same task. Interestingly, the most successful games to date have lived with this limitation, making mostly static worlds with very little actual player impact. Some recent high-profile titles have explored creative ways to improve this trade-off, but remain strongly tied to the quest model. Two of the biggest MMORPGs, BioWare's *Star Wars: The Old Republic* and the later updates to *World of Warcraft*, are built around quests, but also use *instancing* or *phasing* techniques to separate players into their own copies of the world at key times, enabling non-shared change. Trion Worlds' *Rift*, one of 2011's significant successes, started with the idea of a fully dynamic world, where player actions freely change everything. But they found in testing that players felt disconnected without story to anchor them. They eventually created two layers of gameplay: a static quest-based system and a dynamic system of invasion events (Zacny, 2012). Going a step further, ArenaNet's upcoming *Guild Wars 2* has instanced quests for individual story, but out in the shared world the areas transition between states in response to player actions. Each state has its own quests, which are assigned by location rather than speaking to NPCs. Concerns have already been raised that the cyclical nature of these transitions may reinforce the perceived lack of impact, but this is based only on initial public beta testing.

There is clear initiative in recent MMORPG design to increase player impact, ideally in the shared world. But these designs are also holding on to what is effective about quest-based gameplay. In this paper, we investigate *adaptive quests*, an incremental extension to the dominant quest model that dynamically alters quest requirements to match changing enemy populations in the world, in order to enable increased player impact without abandoning quest narrative. We assess the feasibility of this extension through simulation experiments.

Adaptive Quests

Consider the quest text in Figure 1, taken from an early quest in *World of Warcraft*.

> Hey, you there! Ye're a stout-looking <class>. Lend us a hand? It looks like today's big earthquake shook a bunch o' those barbarous troggs out of the ground, and they're sure steamed about something. My men are doing what they can to hold them off, but we could use a hand. Do yer part - head just south of here and help dig me mountaineers outta trouble. We've got to hold the line!

Figure 1. Quest text (copyright Blizzard Entertainment).

This quest text reflects a certain state of the world. There is an ongoing conflict, a specific event leading to a specific battle, and an immediate need. It comes from a *kill* quest, where the player is tasked to kill 6 "Rockjaw Invaders". These enemies are in-game entities known as *mobs* (from mobile objects). Mobs are inserted into the world and wander within an authored region or path. Each region has one or more specific types of mobs that *spawn* there, controlled by *spawn points* that ensure the population is always available for the next player who comes along to kill them. The sample quest text works with a spawn point that is located where the battle is taking place, which maintains a population of Rockjaw Invaders. This system presents a static world where the Rockjaw Invaders are always there, always fighting, and the player actions have no real effect. The narrative is incoherent, as the player is ultimately thanked and honored for making a change that did not happen. This is by far the most common quest experience found in today's MMORPGs.

Player impact could be increased by allowing player actions to reduce mob populations and eliminate spawn points, actually defeating the enemy. However, simply introducing such *defeat-able spawn points* would lead to two additional problems. The problem of *availability* has been mentioned already: the quest model expects those mobs to be available for any player to work on that quest at any time. With defeat-able spawn points, defeated mobs would no longer be available for others. The problem of *competition* already exists in the quest model because variable numbers of players are pursuing the same mobs at any given time. Particular mobs can become a scarce resource, and players are forced to race each other for kills in order to finish their quests. Current trends view this as a frustrating experience to be avoided. Aggressive re-spawning to meet demand is the standard solution, but that reduces perceived player impact even more. Defeat-able spawn points would exacerbate this problem by decreasing mob numbers.

To address these problems and allow the use of defeat-able spawn points, we propose using adaptive quests that dynamically alter target mobs and locations. In the example text, a player might as easily be sent to fight goblins to the north instead of troggs to the south. Because the other events (e.g. the earthquake) are unverifiable to the player, there is a good deal of leeway. The target mobs could be constrained to a particular type (e.g. has to be troggs), or only to a class of mob types (e.g. civilized types vs. wild animal types). These are simple extensions to the dominant quest model, yet beyond what has been attempted to date.

Related Work

There has been a rise in research on MMORPGs as the genre has grown, primarily focused on broad player motivations and behaviors (cf. Yee, 2006), and correlating them with practical, external concerns like network load (Feng, et al. 2007), addictive behavior (Hsu, et al. 2009) and real-money trading (Fujita, et al. 2011). Very few projects, to our knowledge, have explored modeling and altering game play systems like quests to assess their impact on in-game player experience. Even work directly investigating what players do and why (Suznjevic & Matijasevic, 2010) is concerned with the types of activities players pursue; a much higher level of abstraction than immediate decisions in response to game play systems.

The goal of computer-mediated narratives that adapt to dynamic situations is at the heart of interactive narrative research. By far the most mature pieces of work in the field, resulting in deployed playable game experiences, have worked with relatively small sets of deeper character interactions in a single-player setting (cf. Mateas & Stern, 2005, Rowe, et al. 2009). But the idea of extending the quest system is not novel. The GrailGM system (Sullivan, et al. 2010) aims to extend RPG quests to provide more diverse player choices than combat actions. However, although it draws examples from both single-player RPGs and MMORPGs, there is no discussion of the constraints on world change in a shared space. Similarly, (Fairclough & Cunningham, 2003) and (Riedl, et al. 2011) both use the multiplayer MMORPG quest model, but focus on the constraints of automatic plot generation between a few directly interacting players. Neither project appears to be targeting the issues of large-scale interaction. The True Story system (Pita, et al. 2007) proposes and implements a quest generation system intended for MMORPGs, but it is not evaluated, and it is not stated how it would deal with the problem of change in the world.

The idea of defeat-able spawn points is not dissimilar to recurring events that occur in many current MMORPGs. These are essentially spawn points that cycle on and off. Having defeated mobs return at a later time does give some sense of change, but faces the same problems with competition as any spawn point. Aggressive re-spawning can be used but makes it quite clear that the end of the "attack" is on a timer. That it re-occurs in the same location after a pause only reinforces the lack of player impact.

Simulation of Aggregate Player Dynamics

Because an MMORPG is a complex multiplayer system, aggregate player dynamics impact each player's individual experience. Extending the quest model changes those dynamics and must be evaluated in that context. This would ideally be done in a live game, but there are numerous impracticalities there. To address this problem and evaluate our defeat-able spawn points and adaptive quests, we have created a lightweight MMORPG simulator. This simulator is intended to act as a first pass for assessing theoretical impact of design decisions and system extensions. It cannot capture arbitrary human behavior, but we believe that it can flesh out the space of possibilities, highlighting problems and verifying hypotheses before undertaking the high cost of live testing.

The simulator implements the dominant quest model, with quest-giving NPCs, mobs and spawn points. These are organized within a single *zone*, a geographical area with author-specified regions that serve as *quest hubs* and *questing areas*. Quest-giving NPCs are placed in the former, while mob spawn points are placed in the latter. The simulator runs any number of concurrent simulated players through a zone where they seek out, accept and complete quests. For this experiment, we use only kill quests as described above, and *delivery* quests that direct players between hubs. This setup is most similar to the early part of real games, where completing quests is by far the predominant activity.

Aggressive Spawn Points

The standard model in our simulator uses *aggressive spawn points* to manage player demand for mobs. The spawn rate for these points varies linearly with the difference between a specified maximum number of mobs and the current number of mobs spawned and still alive. As that difference increases, new mobs are spawned faster. The maximum number of mobs for an aggressive spawn point is initially set by authorial specification (based on expected quest demand), but can increase as more players arrive in the region. It is hard capped based on a global maximum mob density applied to the region it is in.

Defeat-able Spawn Points

Defeat-able spawn points allow players to reduce their mob populations to nothing, defeating them. The simplest way to do this is to just turn off re-spawning, but that is problematic for variable-sized groups. Instead, our defeat-able spawn points continue to re-spawn mobs to a maximum, but also reduce that maximum. If 1 player kills 1 mob, the maximum goes down by 1, but if 20 players kill 20 mobs concurrently, it also goes down by only 1. Thus each player's contribution has impact, but a crowd does not instantly trivialize the encounter. Eventually, the defeat-able spawn point reaches a minimum population threshold which it maintains until no players in the region require additional kills. At that time, the spawn point has been defeated and dies out.

Simulated Players

The simulation is given a number of concurrent players to simulate in every run. It starts by logging in (spawning) new simulated players into the zone around a starting hub location. A ramp-up time is specified that controls the initial log in rate. Players log out when they run out of quests to pursue. Once the target number of concurrent players is reached, every player that logs out is replaced by a new player.

Each simulated player is controlled by a player model that determines what actions they take. (Smith, et al. 2011) presents a taxonomy of player models, under which this is a straightforward Individual Generative Action model that stands in the place of an actual player's in-game actions. However, the goal is not to accurately predict what a particular individual would do, but rather to create a diverse set of possibilities. We have separated higher-level choices about which quests to pursue and when to turn-in from lower-level behaviors like fighting, searching and traveling. We implemented a naïve preference system to create a distribution of random players who would prefer the closest quest, the most rewarding quest and so forth. But it is not clear that such features are any more valid across a random population than say the quest that appears first in the log. Collecting user preference data is beyond the scope of this work, and reserved for future work. The results obtained here used random choice instead. The lower-level behaviors are specified using a *behavior tree*, a technique for writing modular game AI that has gained popularity, notably after use in Bungie's *Halo* series. A behavior tree is an agent controller that is recursively updated with each game step to produce a desired behavior. Each non-leaf node in the tree specifies particular traversal semantics such as sequential, looping or priority selection. All transitions between sub-trees are therefore explicit in the tree structure (in contrast to finite state machines where states internally trigger transitions), and any tree can be parented into another tree to create more complex behaviors. For example, in our player behavior tree, there is a sub-tree for fighting that looks to see if any quest-relevant targets are available and attacks them. There is also a sub-tree that wanders around randomly in the region of a selected quest. By parenting these to a priority selector node, in that order, we create a behavior that attacks relevant mobs where they are available, and otherwise wanders looking for them.

Implementing Adaptive Quests

Each hub has a set of quests to assign, and ordering constraints between them. Any time a simulated player interacts with a hub, it turns in all completed quests for that hub, and is given all available quests. The target mobs and location are part of the quest specification. In the adaptive quest extension, each time a quest is assigned to a player, it is re-bound to a valid mob type and location. A valid mob type can be the type specified in the quest, or any mob type that belongs to the same class as the specified mob type.

The adaptive quest re-binding algorithm categorizes all regions linked to a hub as *empty*, *low*, *medium* or *high*. Empty indicate the absence of a spawn point, while the latter three are levels of the player demand to mob ratio. For a given set of quests available to a player at a hub, if any can be bound to a low spawn point, they are. If any can be bound to a new spawn point for a mob type not currently being spawned by the hub, they are. Any quests not meeting either of those criteria are not assigned at that time, unless that would result in no quests being returned and the player logging out. In that case, the quests which can be bound to medium spawn points are considered next. If any of them can be bound to a new spawn point (which necessarily duplicates at least one medium spawn point), and the number of duplicate spawns of that mob type is less than a threshold, then a single new spawn point is created and a single quest returned. When a duplicate spawn point is created, the old spawn point is marked *closed* and can no longer be bound. This allows it to die out as players finish with it, freeing its region. If no such duplicate spawn point can be created, a single quest is bound to a medium spawn point and returned. If no quests with medium bindings are available, the same process is repeated for quests with high bindings. If no quests can be bound, none are returned and the player logs out.

In order to give the adaptive quest system greater flexibility, it is able to treat regions as two smaller regions comprising the same overall area and thus the same maximum mob population.

Measures

The simulator is instrumented to collect data about player experience in terms of *frustration*, *time to completion* and *completeness*. Frustration is a measure of undesirable competition, calculated as the percentage of time a player spends attempting to kill mobs for a quest, but is unable do so because there are none available. This measure requires internal knowledge of the player behavioral state, something difficult to assess with live players. Time to completion is simply the amount of time it takes a player to go through the zone quests, while completeness is the percentage of quests completed. As described above, in the adaptive quest cases it is possible for quests to be unavailable due to the changing world state, and players can move on none the wiser, thus missing content.

Experimental Setup

In these experiments, we assume that enabling defeat-able spawn points will increase frustration, due to mob scarcity. Our central hypothesis is that enabling adaptive quests along with defeat-able spawn points will mitigate that increase in frustration.

The first set of experiments tests the base assumptions that, for a given zone topology and set of quests, the frustration rate will increase when 1) higher numbers of concurrent players are present or 2) more ordering constraints are placed on the quests. The simulation is run over a simple zone with a single, central hub and 6 adjacent questing regions. 6 kill quests are defined, each with its own region and type of mob. Simulations are run with 20 to 200 concurrent players (in increments of 20), and with 1, 3 or 5 of the 6 quests unordered. Players are spawned into the zone with a 3 minute (game time) ramp up to reach the target concurrent player count. As players complete the zone and log out, they are replaced by new players coming in.

The second set of experiments tests the hypothesis that aggressive spawn points outperform defeat-able spawn points, but the combination of defeat-able spawn points and adaptive quests together outperforms aggressive spawn points. Superior performance is defined as decreasing frustration without decreasing completeness or increasing time to completion. That is to say, players are more able to efficiently pursue their quests, but without skipping content or wandering aimlessly. The same zone, quests, quest orderings and player counts are used as in the first set. The four conditions being compared are: aggressive, defeat-able, adaptive (type) and adaptive (class). The defeat-able only condition allows the spawn points to return after dying out, while the adaptive conditions do not. We further hypothesize that adaptive (class) outperforms adaptive (type) due to greater flexibility afforded.

The third set of experiments repeats the second set with a more realistic 3-hub zone configuration. The quest constraints in this zone require players to progress through the hubs in a fixed order, with 2 ordered and 1 unordered quest at each hub. There are no variations for quest ordering. We further hypothesize that the two adaptive conditions will perform better when the same number of regions are linked to multiple hubs (shared), opening up more potential adaptations. Zone specifications with 1 and 2 shared regions per hub are used. This results in six conditions being compared: aggressive, defeat-able, adaptive (type) 1-shared, adaptive (type) 2-shared, adaptive (class) 1-shared and adaptive (class) 2-shared.

Results

10 trials of 30 minutes game time each (accelerated) were run for each of the 180 conditions. For each condition, player frustration is reported as an average over all players in all 10 trials. Time to completion and completeness are reported as averages over all players in all 10 trials who finished the zone.

As expected, the assumption that increasing numbers of concurrent players increases frustration holds true for all conditions, as can be seen in Figures 2-5. The assumption that higher numbers of ordering constraints on the quests increases frustration also holds true, but the difference was minor compared to other effects. For space and clarity, we show only the condition with 3 of 6 unordered quests.

For the second set of experiments, Figures 2 and 3 show that the hypotheses that aggressive outperforms defeat-able and adaptive outperforms aggressive hold true for both frustration and time to completion. Adaptive (class) did so without omitting any quests for any player, while adaptive (type) maintained over 94% quest completion up to the 200 concurrent player condition. The hypothesis that adaptive (class) outperforms adaptive (type) holds true for time to completion (Figure 3), but not for frustration (Figure 2).

For the third set of experiments, Figures 4 and 5 show that the hypotheses that aggressive outperforms defeat-able and adaptive outperforms aggressive hold true again for frustration and time to completion in the 3-hub configuration. The number of shared regions had relatively

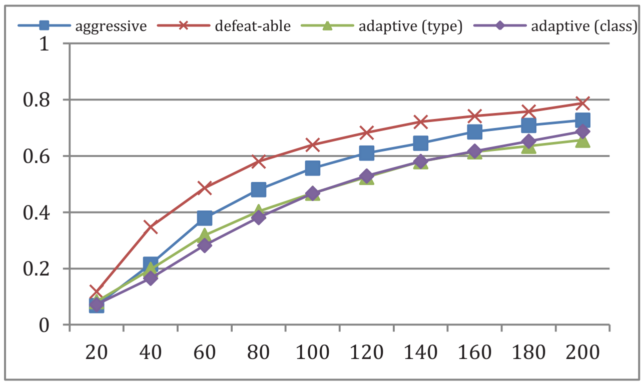

Figure 2. Frustration % (vertical) vs. concurrent players for the single hub configuration

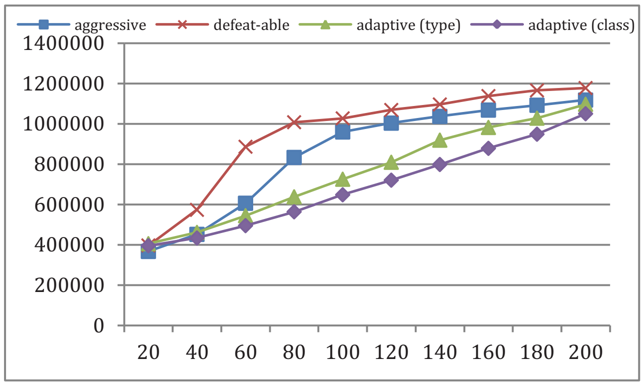

Figure 3. Time to completion in ms (vertical) vs. concurrent players for the single hub configuration

little impact on either, so only the 2-shared cases are included. Adaptive (class) again did so without omitting any quests for any player in both the 1- and 2-shared conditions, while adaptive (type) maintained over 93% in the 2-shared condition but only 86% in 1-shared.

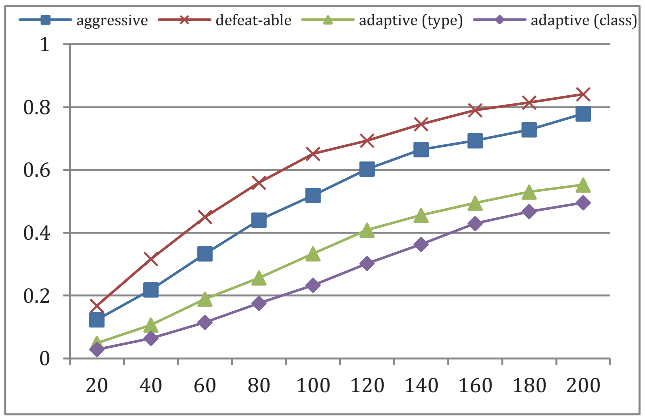

Figure 4. Frustration % (vertical) vs. concurrent players for the three hub configuration

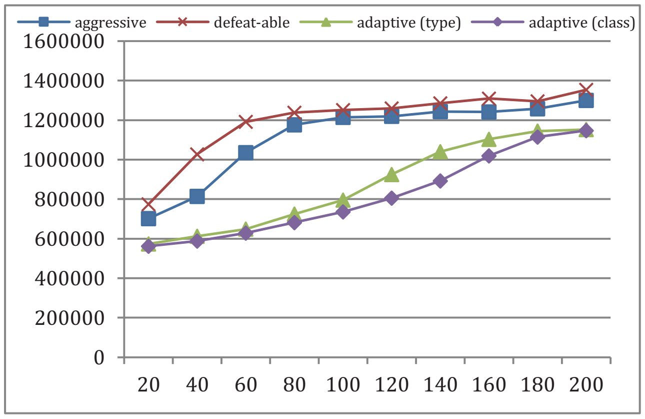

Figure 5. Time to completion in ms (vertical) vs. concurrent players for the three hub configuration

Figure 6 shows the number of players on average that finished the 3-hub zone in a 30-minute trial. This shows the real impact on player throughput in a highly structured zone.

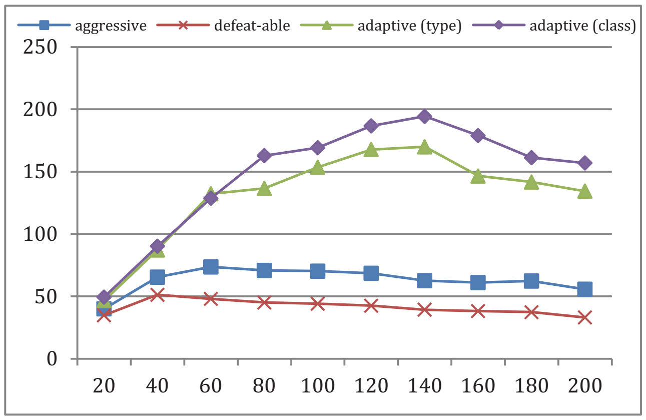

Figure 6. Number of players finishing the zone (vertical) vs. concurrent players for the three hub configuration

Discussion and Future Work

In this work, we assume that a defeat-able spawn point that can actually be beaten provides higher perceived player impact than a spawn point that cannot. Our results provide evidence that adaptive quests can allow these defeat-able spawn points to be used under the quest model, while reducing undesirable competition and vastly improving overall player throughput. The results were less sensitive to quest ordering and zone layout than expected, and managed to consistently scale better with more concurrent players in terms of reduced frustration. In some cases the cost was skipping content. At under 10%, and especially when a zone is overloaded, this is probably a good thing.

This study focused entirely on an early-game questing scenario, with the assumption that players are primarily concerned with completing quests. We claim that this has some generality due to widely accepted design trends of separating the leveling experience from the more diverse *end game*, encouraging "playing together alone" and the fact that players not actively questing are less likely to impact quest mob populations.

A significant limitation of this study is the randomness of the simulated player choices. While these choices may be arbitrary, the particular zone layout and quest features most likely should create trends. These results serve as motivation and guidance for the challenging task of gathering data from actual players to develop accurate behavior models. We are rebuilding the simulator to connect real players to begin collecting this data.

There may be some legitimate concern that the quest model as described here is dying out with the upcoming generation of AAA MMORPGs. This may or may not happen, but we view the steps taken in games such as *Rift* and *Guild Wars 2* as exactly the type of evolutionary improvements to quest-based guidance that we are exploring here.

Acknowledgements

This project was supported in part by the UTPA Center of Excellence in STEM Education Undergraduate Research Program, Office of Naval Research contract W911NF-11-1-0150.

References

Fairclough, Chris; Cunningham, Pádraig. A Multiplayer Case Based Story Engine. Dublin, Trinity College Dublin, Department of Computer Science, TCD-CS-2003-43, 2003, pp6.

Feng, Wu-chang, Brandt, David, and Saha, Debanjan. 2007. A long-term study of a popular MMORPG. In *Proceedings of the 6th ACM SIGCOMM workshop on Network and system support for games (NetGames '07)*. ACM, New York, NY, USA, 19-24.

Fujita, A., Itsuki, H., and Matsubara, H. Detecting Real Money Traders in MMORPG by Using Trading Network. In *Proceedings of the 7th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, pp. 26-31, Oct., 2011.

Hsu, Shang Hwa, Wen, Ming-Hui and We, Muh-Cherng. Exploring user experiences as predictors of MMORPG addiction. *Computers & Education*. Volume 53, Issue 3, November 2009, Pages 990–999.

M. Mateas, A. Stern. 2005. Structuring Content in the Façade Interactive Drama Architecture. *Conference of Artificial Intelligence and Interactive Digital Entertainment (AIIDE),* Los Angeles, June 2005.

Pita, James, Magerko, Brian and Brodie, Scott. True story: dynamically generated, contextually linked quests in persistent systems. In *Proceedings of the 2007 conference on Future Play (Future Play '07)*. 2007. ACM, New York, NY, USA, 145-151.

Riedl, Mark, Li, Boyang, Ai, Hua, and Ram, Ashwin. Robust and Authorable Multiplayer Storytelling Experiences. *Proceedings of the Seventh International Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE).* 2011.

Rowe, J. and Mott, B. and McQuiggan, S. and Robison, J. and Lee, S. and Lester, J. 2009. Crystal island: A narrative-centered learning environment for eighth grade microbiology. *AIED Workshop on Intelligent Educational Games*. pp. 11--20.

Smith, A.M., Lewis, C., Hullett, K., Smith, G., Sullivan, A. An Inclusive Taxonomy of Player Modeling. Technical Report UCSC-SOE-11-13. University of California – Santa Cruz. 2011.

Sullivan, A., Mateas, A., and Wardrip-Fruin, N. Rules of engagement: moving beyond combat-based quests. In *Proceedings of the Intelligent Narrative Technologies III Workshop*. 2010.

Suznjevic, Mirko, Matijasevic, Maja. Why MMORPG players do what they do: relating motivations to action categories. *International Journal of Advanced Media and Communication*. Vol 4, Number 4/2010, pp 405-424.

Yee, Nick. Motivations for Play in Online Games. *CyberPsychology & Behavior*. December 2006, 9(6): 772-775.

Zacny, R. 2012. Rift Executive Producer Scott Hartsman explains how dynamic content drives the world of Telara. PC Gamer website. http://www.pcgamer.com/2012/03/06/rift-executive-producer-scott-hartsman-explains-how-dynamic-content-drives-the-world-of-telara