Five Grand Challenges in the Engineering of Networked Digital Games

T.C. Nicholas Graham
School of Computing
Queen's University
graham@cs.queensu.ca

ABSTRACT
In this position paper, I introduce five grand challenges in the development of networked digital games. While this list is not intended to be exhaustive, I argue that these challenges are significant in that they are holding back the development of the next generation of innovative games.

INTRODUCTION
Digital games have become an important entertainment medium, with the Entertainment Software Association reporting that 68% of American households play computer or video games [3]. Within these, multiplayer games, those that allow groups of people to play together, have become particularly popular.

Recent years have seen stagnation in the design of digital games. While innovative input technologies such as the Wii Remote have led to new forms of interaction in games, many successful games are derivative remakes of existing games [5]. The year 2010 has already seen the release of Bioshock 2, Mass Effect 2, God of War 3, Command and Conquer 4, among others.

In this position paper, I identify five challenges that are holding back innovation in the development of multiplayer digital games. While these challenges are by no means exhaustive, they are all important, and the gaming industry would benefit from their solution. The challenges are development cost, real-time consistency maintenance, supporting truly massive multiplayer play, allowing player-generated content, and security. None of these have obvious solutions, and therefore can serve as grand challenges for researchers in the engineering of networked digital games.

CHALLENGE 1: DEVELOPMENT COST
The cost of developing games has increased greatly, with modern "AAA" games costing upwards of $20 million to produce. But the game industry is hit-driven, where only a minority of titles recoup their development costs: according to the New York Times, as of March 2009, only 16 of 486 released titles for the Nintendo Wii console had been profitable [10].

Figure 1. Game sketching with Raptor can help evaluate fun before the game is even prototyped, liberating designers to try new ideas.

Very large studios can allow a few hits to pay for many titles that fail. Smaller studios do not have this luxury, as a single failure can lead to bankruptcy. Many studios therefore act conservatively, releasing only revisions of games that have already proven successful.

Our first challenge problem is therefore to find ways of evaluating whether a game idea will be fun before the game has been implemented, allowing game studios to innovate with lower risk. One emerging solution is game sketching [1, 13], which employs *Wizard of Oz* techniques to allow games to be played before even prototypes are available (figure 1).

CHALLENGE 2: CONSISTENCY MAINTENANCE
Entertaining gameplay often depends on players' having a consistent view of the game world [12, 4]. For example, in a first-person shooter game, players need accurate representations of their opponents' positions in order to be able to aim at them [8]. In practice, because of latency in the networks connecting the players' computers, inconsistency in players' views is inevitable (figure 2).

One current solution to this problem is to design the game to be less real-time. For example, in World of Warcraft, players perform commands that trigger an action at some point in the

Presented at Design and Engineering of Game-like Virtual and Multimodal Environments, a workshop at EICS 2010, Berlin, Germany, June 20, 2010.

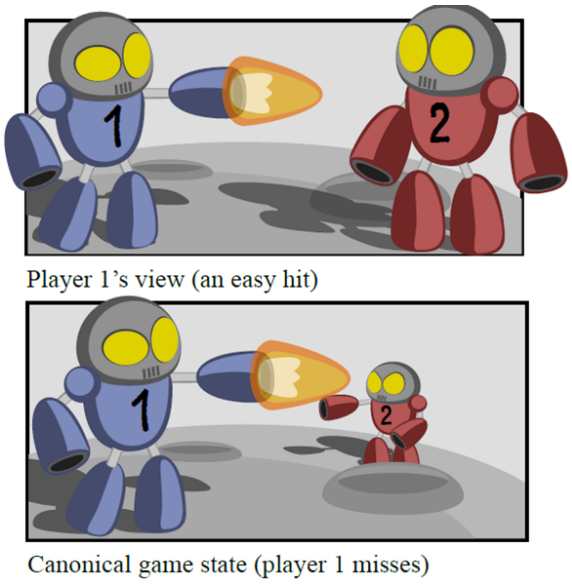

Figure 2. Consistency problem in first-person shooter. Maintaining consistency of players' views in real-time is becoming increasingly important with the introduction of motion-tracking input devices.

near future (e.g., casting a spell or using a special sword attack.) Since the actions are not instantaneous, latency can be masked. Another increasingly popular solution, *local lag*, makes latency predictable by inserting delays in local processing of input [11].

Modern game controllers, however, such as the Wii MotionPlus, Microsoft Natal and PlayStation Move encourage increasingly real-time styles of play, where players' actions are reflected in the game world as they occur.

Possible avenues for addressing this problem include finding accurate approaches for client-side prediction, better approaches for determining the consistency requirements of different circumstances, and imaginative consistency maintenance algorithms tuned for specific game situations.

In summary, our second grand challenge is how to accommodate truly real-time play in the presence of network latency.

CHALLENGE 3: MASSIVELY MULTIPLAYER

Games increasingly allow large groups of people to interact in real time. Massively multiplayer online games (MMOGs), for example, allow thousands of people to concurrently log in to the same world. Current games have, however, suprisingly small limits on how many people can interact in the same virtual space. This is due to bandwidth limitations (conveying the positions and actions of large numbers of players from server to client in real time), CPU limits on the server, and limitations of the graphics processing unit of the client in rendering large numbers of detailed models.

To address this problem, MMOGs carefully funnel players so that only a few dozen are involved in the same combat

Figure 3. The City of Heroes' *Mission Architect* is a rare example of a tool allowing players to create content within an online game.

(or other activity.) When numbers of players become large, games frequently "instance" playing areas, creating multiple copies of an area, each capped in population. Other current solutions include level of detail updates [2] and predictive client-side updates [9].

Our third grand challenge is to allow truly large-scale concurrent play, as necessary to support battles with hundreds of units per side.

CHALLENGE 4: PLAYER-GENERATED CONTENT

Players of online games consume content dramatically faster than developers are capable of creating it. One solution to this problem is to permit players to create content and add it to the game for themselves and others to enjoy. Being able to participate in meta-gaming (the game of creating games) has the potential to extend players' interest in the game world. Examples of this approach already exist. NCsoft's City of Heroes MMOG includes a "mission architect" feature (figure 3) that allows players to create quests for other players' to carry out. The feature includes a voting function, allowing players to rate quests and therefore allow the best player-created content to be easily found. EA-MAXIS' Spore game includes a sophisticated model editor that allows players to create creatures, buildings and vehicles for inclusion in the game (figure 4). While Spore is a single-player game, player-created models are automatically transmitted to a server, and subsequently may appear in other players' game sessions.

There are many barriers to the broad adoption of player-generated content in a networked context. Since player-created models cannot be dynamically loaded onto the client,

Figure 4. EA-MAXIS' *Spore* provides a sophisticated editor allowing players to create 3D models for inclusion in the game. Unfortunately, the editor can be used to create content that many players may find objectionable.

Figure 5. The challenge of allowing player-created content is illustrated by one of dozens of "penis monsters" created by players of Spore shortly after its release.

Figure 6. The security challenge: virtual gold sales through third-party web sites can imbalance game economies and can provide incentive for theft of in-game items that now have real monetary value.

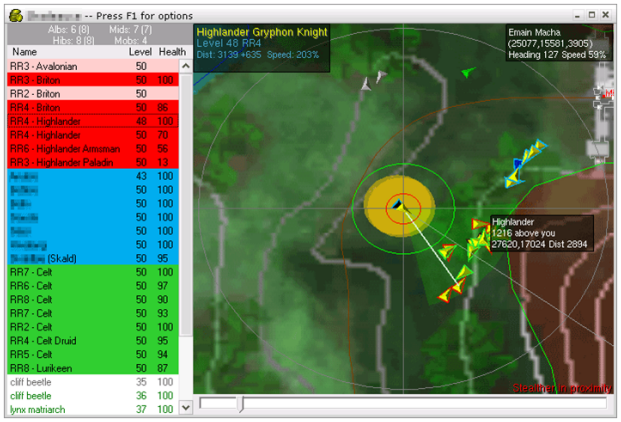

Figure 7. DaocSkilla tool used to cheat in EA-Mythic's *Dark Age of Camelot* Massively Multiplayer Online Game.

prohibitive levels of bandwidth may be required from the server to the client. The scalability implications of player-generated content can be seen in Linden Labs' Second Life, which according to Kumar et al. can host only 40 players per server as opposed to thousands in traditional MMOGs [7].

Game companies may inadvertantly find themselves redistributing player recreations of copyright IP, leading to legal problems. Players may use the games' mechanisms to engage in illegal activity, e.g., setting up a casino. Finally, players may use these tools to create offensive or lewd materials inconsistent with the game's intended market (figure 5).

Game companies do not typically have the resources necessary to evaluate all player content, and therefore must either rely on mechanisms allowing players to report undesirable content, or provide no official channel for distribution of player-created content. Because of the inherent limitations to these approaches, most virtual world games do not allow players to create and upload their own content.

Solutions to this problem must lie in the technical sphere of finding ways of predicting what content a player is likely to need in the near future and distributing it to the player's client asynchronously, through to attempting to find technical means of identifying objectionable or illegal content quickly.

Our fourth grand challenge is therefore to find mechanisms enabling players to unleash their creativity by creating content for use in networked games.

CHALLENGE 5: SECURITY

Cheating in online games can lead to reduced enjoyment for legitimate players, and even to financial loss from the theft of online goods [6]. Examples of cheating include the pur-

chase of virtual currency for real money (figure 6) and the use of cheating programs to gain advantage over other players (figure 7).

One set of current techniques for combatting cheating is architectural. These include ensuring that no game-critical information is resident in the game client (in case it is hacked), that all game actions are audited on the server (to detect suspicious behaviour), that no unnecessary information is in the client-server protocol (in case it is snooped), and that the protocol includes encryption and challenge-response features (in case of man-in-the-middle attacks.) These features constrain possible implementation architectures, notably ruling out peer-to-peer architectures, in turn constraining the solution space for our other four grand challenges.

Another approach is to design the game so that cheating is less desirable. For example, the *DaocSkilla* program (figure 7) provides a radar view, easing player-versus-player combat in EA-Mythic's Dark Age of Camelot game. Later MMOGs such as World of Warcraft have included radar views in the game client, removing the advantage conferred by this form of cheating.

Our fifth grand challenge is therefore to find techniques for reducing the opportunities of cheating in games, helping to free developers to pursue development of gameplay instead of anti-cheating methods.

CONCLUSION

In this position paper, I have identified five grand challenges in the design of networked digital games. While this list is not intended to be exhaustive, each of the listed problems is holding back innovation in multiplayer games. Each problem is complex and has no obvious solution, and therefore is appropriate for investigation within the academic community.

REFERENCES

1. M. Agustin, G. Chuang, A. Delgado, A. Ortega, J. Seaver, and J. Buchanan. Game sketching. In *Proceedings of the 2nd international conference on Digital interactive media in entertainment and arts*, pages 36–43. ACM, 2007.

2. M. Brockington. Level-of-detail AI for a large role-playing game. In *AI Game Programming Wisdom*. Charles River Media, 2002.

3. Entertainment Software Association. Industry Facts. http://www.theesa.com/facts/index.asp.

4. R. Fletcher, T. C. N. Graham, and C. Wolfe. Plug-replaceable consistency maintenance for multiplayer games. In *Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games*, pages 34–37. ACM, 2006.

5. U. Hagen. Where Do Game Design Ideas Come From? Invention and Recycling in Games Developed in Sweden. In *DiGRA*, pages 1–11, 2009.

6. G. Hoglund and G. McGraw. *Exploiting online games: cheating massively distributed systems*. Addison-Wesley, 2007.

7. S. Kumar, J. Chhugani, C. Kim, D. Kim, A. Nguyen, P. Dubey, C. Bienia, and Y. Kim. Second Life and the new generation of virtual worlds. *Computer*, 41(9):46–53, 2008.

8. M. Mauve. How to keep a dead man from shooting. In *Interactive Distributed Multimedia Systems and Telecommunication Services*, pages 741–779. Springer, 2000.

9. L. Pantel and L. Wolf. On the suitability of dead reckoning schemes for games. In *Proceedings of the 1st workshop on Network and system support for games*, pages 79–84. ACM, 2002.

10. M. Richtel. Video game makers challenged by the next wave of media. *New York Times*, March 30, 2009.

11. D. Stuckel and C. Gutwin. The effects of local lag on tightly-coupled interaction in distributed groupware. In *Proceedings of the ACM 2008 conference on Computer supported cooperative work*, pages 447–456. ACM, 2008.

12. J. Vogel and M. Mauve. Consistency control for distributed interactive media. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 221–230. ACM, 2001.

13. C. Wolfe, J. Smith, W. G. Phillips, and T. C. N. Graham. A model-based approach to engineering collaborative augmented reality. *Engineering of Mixed Reality*, pages 293–312, 2010.