The Intelligence in the MMOG: From Scripts to Stories to Directorial AI

Nathan Combs
BBN Technologies
Cambridge MA 02138

ABSTRACT
This paper describes how the design of behaviors in games and Massively Multiplayer Online Games (MMOGs) is based on a style of scripting that is consistent with a cinematic perspective of game design. This technique is presented as paradigmatic of how AI is conceptualized in games. This paper will outline why this approach is not likely to scale in the future at which time, it is suggested, a more declarative and then directorial style of developing and conceptualizing AI will emerge. This paper suggests that MMOGs are particularly vulnerable because of their need for substantial (and increasing) quantities of entertainment content to populate their worlds. Looking forward, this paper suggests that we may expect MMOGs to transition from a less-scripted universe into a more simulationist AI future.

Author Keywords
Massively Multiplayer Online Games, Artificial Intelligence, Directorial AI.

PROGRAMMING THE ENVIRONMENT
In the games community a script refers loosely to a relatively easily modifiable program for implementing game behavior. It may serve as "glue" that rides above lower-level functions. To the games industry, scripting is a means of partitioning concerns between the developers and the level (content) designers.

Most AI development in games is represented by a three-step scripted design pattern - e.g. characterizes most if not all games level/mission editors. First, the developer specifies an environment -- e.g. defines terrain, textures, a map, etc. Second, the developer specifies objects to embed in that environment. Objects may be static or fixed within the environment; or objects can be dynamic -- e.g. they can move around. Objects can also be placed to signify episodic behavioral elements -- e.g. triggers. Third, in the case of dynamic objects, the developer can link them to abstract pathway representations to indicate where they should move. Coordination of objects and events is often graphical. Using "springs" and waypoints as well as embedding logic into the objects themselves (via scripts and property values) -- coordinating behavior can be intricately defined.

While the details differ across individual games and genres, the pattern is familiar: there is a map, there are objects, and objects can be scripted in terms of the map. Operation Flashpoint (Codemasters) provides a graphical mission editor, whereas Unreal Tournament uses a more sophisticated Computer-Aided...
Design (CAD) level-editor. Morrowind (The Elder Scrolls) offers a “construction kit” that integrates of a CAD level editor and a plugin software architecture. Et cetera. This design pattern is an environment-based programming style that has its roots from the earliest game systems (e.g., RuthMOO, see Herdman). This style seeks to merge content and programming into a combined form that facilitates bootstrapping a game world instance.

Thus, the environment programming style is composed of three types of elements:

- Events (including “triggers”)
- Character (NPC) behaviors
- Spatial and coordination instructions

Events or plotlines may be unlocked or “triggered” as the player moves around and takes actions in the game world. Events may be explicit (e.g. an explosion when a player moves near a mine in an Operation Flashpoint scenario). Events can also be implicit, e.g. the consequence of one or more prior actions taken by a player (e.g. before you can obtain the quest from Dramaticus, you need to have done the following things…). In all cases, however, the events, the characters, and the spatial coordination instructions are expressed in terms of each other – the environment – rather than abstract goals.

**IMPERATIVE VERSUS DECLARATIVE PARADIGMS AND THE CINEMATIC LANGUAGE**

Declarative representations encode relationships and facts about the world. From these facts one can reason about the goals and rules of a game world. Most game scripts are expressed in imperative programming languages¹ (see GameDev.Net).

However, developers should not have to write in a style of (A.) to say (B.). Both have their place, and both styles should be an option. Why then do game developers overwhelming favor imperative styled scripting? One factor is likely the technology culture: most programmers were trained in procedural languages. Another factor could result from the cinematic “language” of computer games. Lev Manovich establishes this premise with the claim that “element by element”, cinema²

---

¹ Imperative programming languages are characterized by use of variables, state modification through assignment, and instruction sequencing (begin-end, blocks, loops).

² Declarative programming languages are characterized as having no implicit state (no assignment), expression evaluation instead of instruction sequencing, chaining or recursion (instead of loops).

³ On page 71. Lev Manovich defines his use of “cinema” (embraced by this paper) as a “shortcut” – they do not stand for film necessarily, “but rather for larger cultural traditions…includes the mobile camera, representations of space, editing techniques,
is being poured into the computer interface, including games (pp 86):

“…first one-point linear perspective; next the mobile camera and a rectangular window; next cinematography and editing conventions, and, of course, digital personas also based on acting conventions borrowed from cinema, to be followed by make-up, set design, and the narrative structures themselves. From one cultural language among others, cinema is becoming the cultural interface, a toolbox for all cultural communication… Cinema’s aesthetic strategies have become basic organizational principles of computer software. The window in a fictional world of a cinematic narrative has become a window in a datascape. In short, what was cinema has become humancomputer interface.”

From this we can suspect how a cinematic format might influence the design of behavior in computer games. Game level designers may be at greater ease thinking about game behavior in cinematic terms. Furthermore, when crafting a game they may find an imperatively structured cinematic language of “stage instructions” a more natural expression than would be statements of high-level rules and goals. This would be especially true if there were no creative intelligence available to translate those high-level rules and goals into exact and well-placed (e.g. imperative) instructions for specific action.

Just as the computer game is evolving, so too is the cinematic language that describes it. Larger worlds, more camera angles, plenty of perspectives – these stretch the cinematic language of interfaces and of behavior.

Stretching leads to transitions – for example, Greg Kasavin describes his unease with the migration from 2D to 3D game displays in terms of a weakening of a traditional (and narrowly scoped) view of the cinematic underpinnings of the game:

“My problem with 3D graphics in games--and it's always been my problem with 3D graphics in games--is that they're unedited. You can often view the action from any angle, and frames of animation are typically never skipped. In a way, then, I think the cinematic power of gaming almost took a step back with the transition from 2D to 3D. 2D game characters are displayed precisely how the artist chooses to display them to you. There is no extraneous frame of animation to be found. 3D game characters, meanwhile, are yours to control, so you may rotate them and view them from whichever unflattering angle you like.”

Another perspective, however, is that all this inevitable stretching leads to an extended view of a cinematic language (e.g. Lev Manovich). Yet, even if we broaden the definition of a cinematic language to fit the MMOG experience - Kasavin’s unease does emphasize a particular need co-opt non-linear and goal-driven behavior within a cinematic language of MMOGS. Granting new freedoms of perspective, exploration, and activities to players, ironically, may end-up shifting greater burdens to the AI to complete a coherent game world experience. True, the imperative style of description has historically appealed to (procedurally-minded) developers and (cinematically-minded) content designers, however, this may change, in part driven by pressures from within the industry.

For game worlds we may see an increasing use of simulation to construct player spaces in game worlds. At recent Game Developer
Conferences. Speakers, e.g. Peter Molyneux, Will Wright (2004), claimed that games AI will play an increasingly important role in creating new dynamic opportunities for players within game worlds. Emergent game worlds will allow players to remain engaged longer with less developer workload. A consequence of this paradigm shift away from scripted worlds is that this will likely drive MMOG development and design towards more declarative AI models. However, cost-effectively modeling large worlds will likely require new ways of expressing and managing (and maintaining) MMOG AI.

Further driving change will be a need to play test ever larger and more sophisticated virtual worlds. Current approaches grounded on play testers is labor intensive and will likely not be able to scale well (e.g. cost). Consider, for example, the uniquely MMOG habit of deferring testing to the player community (ala frequent and post-launch patches and tweaks). One solution would be to make greater use of external simulation. This likely is best achieved using external (and easily manipulated) models of game worlds that are constructed separate from the worlds themselves. These models can then be examined without the cumbersome interventions into a live system.

While today simulation is used within the MMOG play tester community, its application is narrowly scoped (e.g. Carpenter 2003) to restricted in-game applications such as combat (even then, however, it’s use is very constrained) etc. To go to the next step beyond testing game subsystems and play facets in piecemeal will require larger and integrated model of the game world and of player behaviors. Without it, one will not be able to evaluate system properties such as stability and convergence (or the lack thereof) with broad implications to players and the game world (e.g. player economy).

Ultimately, and counter to current industry practice, it would be in the best interest of MMOG developers to drive their live implementations as well as their simulations from a single description. In the industry today, the rules of the game, the functions and forms of the game world itself, are established in the code. The code defines as well as is the game. Game developers do generate external specifications (documentation) – but these are not testable (cannot be executed, like code). Furthermore, they all suffer from the the problem common to software development: documentation invariably drifts from implementation. Modern software engineering practices have long strived to integrate more tightly specification and behavior (see Unified Modeling Language). However, these solutions are imperfect and invariably, drift between what is built and what is said is built happens. While there has been some speculative work about coupling “game generation” from high-level descriptions (e.g. Orwant)4 – this paper speaks to the more

4 Ramesh Srinivasan et al. proposed a Community Compiler. Their idea is based on Orwant’s work in as much as they hypothesize a similar high-level language that could drive generation of "communityware". With a Community Compiler, they imagine empowering participants with the ability to experiment and iterate on community design. Or in their words:

“we hope that community members engage in the ‘iterative process of trial and error’ and achieve fluency about the domain: their community.”

reasonable possibility of separating only the game behavioral components from the implementation (analogous to how enterprise systems are often designed to separate the “business rules” from the enterprise application).

The problem is all the more complicated for MMOGs as there are many levels of performance to consider, e.g. Lars Konzack described seven layers of computer game (2002): hardware, program code, functionality, game play, meaning, referentiality, and socioculture. Any significantly sized MMOG simulation would span many (most) of these layers – an immense challenge. This perhaps suggests a need for more knowledge-rich encoding of an MMOG – one perhaps based on declarative paradigms and programming techniques. Related to this point, Raph Koster (2004) spoke to the value of simulation-based testing of an MMOG database in this way:

“Several years ago I proposed that you could create agents mirroring playstyles and patterns and let them loose in your

actual MMO database to attempt to model things like min-maxing of equipment, areas or zones that are underutilized (a very common problem), population centers, etc…

But it’s a non-trivial amount of modeling. You’d need to have a baseline for comparison, which would presumably be an already running game, along with collections of data to match the experimental results to. Then you’d iterate your agent behaviors and models until they matches the real game to a certain margin of error. Then you’d attempt to test changes within the world on that agent-driven testbed…

(Applying) it to another game, or even to the same game after accumulated changes, would be risky, though. After all, games select for certain user types and behaviors. It’s quite likely that a generalized model would be difficult to develop…

If you could get it to the point where the agents evinced boredom, it would be a hellaciously powerful tool for churn management and game longevity…”

From these comments one can suspect how MMOG implementations based on scripted coding and rules embedded in live implementation preclude easy analysis based on simulation. Transitioning from a declarative paradigm for MMOGs will likely involve transitioning from imperative styled scripting towards declarative coding styles will be hard – and counter to the experience of current developers. Such a transition, however, may be facilitated by use of hybrid forms. Yet, shifting programming styles can

5 We are likely to witness the development of declarative infrastructure first with server based games (including MMOGs), for three reasons. First, these architectures are already largely componentized (e.g. database, login, etc.). Second, server-resident games share a number of technical and performance requirements with server-based enterprise applications, e.g., performance, scalability, and support for dynamic “hot swapping” of logic and rule-bases. Furthermore, from the enterprise sector there is an established tradition for using declarative design (e.g. encoding the “business rules” separately from the implementation). This sector has evolved a range of products that uses rules-based scripting to customize middleware.

6 Hybrid imperative and declarative programming languages may provide good technical bridges between the imperative and declarative styles, e.g. a functional
only be the tip-of-the-iceberg; this transition will likely need to be underwritten by a more goals-oriented conceptualization of behavior and AI design in MMOG worlds. Which will in turn impact toolsets, infrastructure, and production pipelines.

**THE CASE FOR ARTIFICIAL INTELLIGENCE AS STORY TELLER IN MMOGS**

The AI in a video game symbolizes the challenge of a technology whose success ultimately relies upon a deception that is shaped in complex ways by the relationships of the players with the technology, the game, and themselves: from the adversarial, to the cooperative, to the team oriented.

In order to have fun, a player must at some level work with the computer so that the intelligence that runs the game is able to connect to the player. At this point it is necessary to distinguish from a spectrum of possible "AI's" in a game. At one extreme people might refer to the in-game characters with which they interact as controlled by the "AI" At the other extreme, people might also speak of the invisible hand that runs the game world as controlled by the "AI"

System simulation games such as Maxis' SimCity fall at one end of the games AI spectrum. The player manipulates the simulated forces within a fictional city. While not all the forces are directly antagonistic to the player (e.g. traffic), their side-effects can undermine the best laid plans. The AI is then the "god in the machine" which the player tries to tame through planning and manipulation of the instruments available (road building, setting tax rates, etc.).

At the other end of the spectrum are games whose AI is invested largely or exclusively within the virtual entities that are the antagonists. Arena game genres such as Unreal Tournament and Doom clearly do this. With these games, the player runs around and shoots these entities before they shoot him. Many of these worlds are devoid of any other AI except for the baddies trying to "gotcha" (why clutter it up with non-combatants?). Here, all the behavior in the game is focused upon an explicit foe.

Ultimately game playing comes down to engaging the participants in an unfolding story. To this end, the computer marshals its AI, whether as virtual evil minions, or as puppet-master of a virtual society. The AI of a game rides above the individual entities within that game - it is associated with the directorial thrust of the game narrative. Whether one is playing a deliberate third-person simulation, or experimenting with "what-if" scenarios in a war game, or sitting in the hot seat in a first person shooter, or trying to mollify cranky giraffes a new exhibit in Zoo Tycoon, the AI is the facilitator of the player's experience.

One advantage with identifying the AI as part of the larger experiential or storytelling apparatus is that it simplifies the basic question confronting most games AI discussions: what is the AI in a game? Most of what is labeled AI in games, e.g. path-finding and collision detection, seems timid: these are just components of intelligent behavior. Furthermore, other game AI algorithms, such as computing the market clearing price of virtual commodities and algorithms to compute weak points in a line of attacking goblins seem unrelated. All these, however, are united under a larger purpose of game-
related intervention focused on pushing the player experience along.

Traditional distinctions of AI programs were based on their applicability to cognition. In his article "Minds, Brains, and Programs (Searle)," Searle distinguished between strong and weak AI. Strong AI referred to the unrealized case of "the computer is not merely a tool in the study of the mind; rather, the appropriately programmed computer really is a mind, in the sense that computers given the right programs can be literally said to understand and have other cognitive states." Weak AI referred to AI programs that illustrate or simulate or illustrate some aspect of the human cognitive process without really "thinking" themselves.

However, the objective of games AI is different from AI research. It is about entertaining the player. Thus, we need to measure games AI in terms of the efficacy of the player interaction.

Alan Turing proposed in 1950 the Turing Test as a substitute for the question "Can machines think?" The Turing Test was predicated upon deception: if you can write a program that could fool people into believing the program was representing a real person, then you have an artificially intelligent computer program.

While controversial, this perspective was productive in that it was inclusive of computer programs such as Eliza (Weizenbaum), and later chatter-bots (see A.L.I.C.E.). Using clever conversational techniques, they were able to fool people into believing they were representing people directly (e.g. hiding behind the curtain). What this showed was how simple mechanisms can lead to interesting apparent behaviors. So, while solutions like these are grossly simple in what they say about the internals of human cognition, they speak volumes about how people relate to each other within a structured channel.

Instinctively, it seems like games AI should be considered similarly. After all, games are interactive devices that manipulate drama and story-telling, and channel a player’s experience to a crafted end. In this view, the AI is about the intelligence behind the props, the characters, the world of some virtual stage. The aggregation of the intelligence behind these elements shapes the narrative. Should we look to established narrative patterns from more traditional (e.g. linear) mediums for ideas (Frasca, “Simulation versus Narrative”)? Or not? While most ideas are nuanced (Frasca, “Ludologists love stories too”), a pragmatic perspective from Jonas Carlquist is a good place to start:

“I, myself, do not find it hard to see narrative structures in computer games. Many of the recent games follow a pattern that we are familiar with from movies and popular literature. But one main difference between games and other narrative genres concerns the audience’s role; in computer games the players have to interact with the story, something that challenges the linearity of the narrative structure. The storyline of a computer game is often a branching one, which complicates the game’s ability to tell a

---

7 On his games AI resource website Alex Champandard (2001-04) describes the major categories of games AI as Combat, Non-Combat, Analytical and Efficiency. “Analytical Oriented AI” he narrowly defines as a currently non-existent (in commercial games) AI that “is intended to enhance the game experience, but not through a direct interaction between the player and the characters...” The distinction I would make here is that I would reclassify all what is now called games AI under a broadened “analytical” umbrella.
compelling story in the way we are used to.”

Thus, if the AI of a game is tied up in its narrative structure and its perception by the player, then the form of that perception will likely say a great deal about the AI of that game. Could this then be the Test for the AI? In other words, does good drama equal good games AI?

The connection between the player and game involves many technical elements working well together: visual appearance, entity movement, etc. Michael Mateus (2004) describes the AI problem of games as one of creating an engaging player experience. He elaborates this as a first class AI research problem beyond traditional research interest in identifying a correct or somehow optimal AI. This is an important distinction because it claims game intelligence as it is perceived by a player and as it amplifies that player’s experience are the important consideration. This is in contrast to developing objective metrics of AI that are unrelated to the context of play and its socio-culture.

---

8 Domike, Mateus, and Vanouse (2003) suggests thinking in terms of an “AI as an expressive medium.” This conceptualization fits well here. Expressive AI “conceives of AI systems of cultural artifacts… (that participate in a) specific cultural context in a manner that is perceived as intelligent. Expressive AI views a system as a performance…(emphasis added).”

9 Insight into the relationship between the player and the AI can be found on a Sony Everquest FAQ, which presented comment (below) from a beta-tester in response to an AI question. It illustrates how players collaborate, in some sense, with the game world to project intelligence into the AI they encounter –transforming simple behaviors into deeply nuanced ones:

“… This game has the most advanced AI I’ve seen. Bar none (emphasis added). Even the little intricacies that make the game more believable were taken into account. As I am battling one creature (say a large bat for this purpose), I’ve seen a giant rat come up to me, stop, and watch the fight as if sizing me up. You really get the feeling that these things are looking you in the eye, and trying to decide if you are dinner, or something to be feared.

Some attack without warning, others circle in from behind. Still others will act like they are going to go right past you, and suddenly turn around and lay into you with swords, fists, or teeth. The AI definitely appears to be modeled on creature intelligence, some just fight smarter than others. The smart ones will even retreat from a fight and go get help. I have no doubt that the AI programmer is somewhat of a sadist. A sadist with a sense of humor.

You’ll know that sense of humor when you see a hill giant way off in front of you in the distance, and 2 minutes later find that it has made a wide circle to sneak up on you from behind.

Add to this some truly impressive tracking and following code, and SPEED. These creatures are FAST. There just isn’t any outrunning some, you better hope you have the strength or the speed to get away quickly, and they will follow you for a great distance. I have seen one gnoll retreat from a fight, go get helpers, and return on two different tangents to surround the intruders. I’ve seen groups split up, half appearing to have lost interest in the fight, only to turn and rush in.
One way of describing the player interaction is in terms of internal narrative structures, or micro-worlds. (Gingold, 2003). This sense shares with Interactive Theatre (Mateas, 2002) the sense that a larger world experience is decomposed into smaller scenes. This sense is different from those used in AI research (e.g. Papert and Logo; also more recently, NetLogo); in these cases micro-worlds are spaces characterized by sufficient detail and consistency to enable meaningful exploration (in some sense of comparison to the real world): a stage for a different purpose.

Academic and entertainment micro-world designs would converge in the case of where a game world is both entertaining and complete – complete in the sense that to some level of fidelity the world represents an entire facsimile of a real world system. To judge completeness, however, requires that the world needs to be fully declared on some level – the facts and relationships consisting that world are observable and knowable. For reasons I'll suggest later such a convergence is likely unnatural between simulation and entertainment worlds. Unlike with AI research, a contemporary game micro-world is a creation for the player, likely only a façade. It does not imply a self-contained and complete system onto itself.

In an entertainment world, a player is likely to travel through many micro-worlds through the course of a game. Transitions between micro-worlds depend upon the player's actions. Adventure games such as Diablo, Baldur's Gate, and Fallout relied upon randomly generated scenarios to flesh out the spaces between their crafted scenarios. This encouraged replay. The instances of these micro-worlds may be uncountable – in the case of where one spent all their time chasing down random encounters, for example. Yet rarely did these micro-worlds extend beyond a handful of classes/types: limiting the number of patterned variations.

Games such as Total War, XCOM (e.g. UFO Defender), highlighted the generative power of the micro-worlds by stringing together a long sequence of instances. Players could switch between two categories of micro-worlds – one centered upon a high-level strategic map and another around randomized tactical battlefields. From the randomized composition – the game communicrated to the player the illusion of a large game world. A vast world is not necessary to communicate a large experience. A vast world is useful, however, as a simulation for AI and against which players can emerge new content.

It wasn't until the Massively Multiplayer Online Game (MMOG) came along that non-scripted large worlds became possible: players and socialization populate these wide-open spaces with interesting detail. In contrast, Elder Scroll's Arena (adventure game) used a random world generator to fabricate the countryside for those players who ventured outside the cities. The generated world between cities was huge. Unfortunately only the scripted areas of this world appeared interesting; the random villages and random folk found in the "between lands" seemed bland, paler cousins of the city folk. After a while these distances became monotonous to travel.

It is not the size of the game world that determines how open-ended a game feels. The size of the solution space appears to be the important determinant of the viability of a
long-term play experience. A large game world with infinite things to do but where all actions lead to a few outcomes feels more confining than a smaller game with many outcomes. Will Wright (Game designer - The Sims) put it this way:

“So I guess what really draws me to interactive entertainment and the thing that I try to keep focused on is enabling the creativity of the player. Giving them a pretty large solution space to solve the problem within the game. So the game represents this problem landscape. Most games have small solution landscapes, so

Dave Rickey cited the experience of MMOGs to suggest that world size is not in itself the important determinant of player entertainment:

“… In scaled units, all of UO's original gameworld amounted to about 2 square miles, EQ's original world was about 7 miles², Asheron's Call exceeded 500 miles², but much of that was effectively empty (in apparent reaction to this, AC2 was 25 miles²). Dark Age of Camelot's original map was 54 miles², Star Wars Galaxies had 8 planets of 100 miles² each, City of Heroes was roughly 50 miles², and Horizons a mind-boggling 15,000 square miles.

…Most of AC's 500 and Horizon's 15000 square miles of space is bland, uninteresting, and populated randomly (and often sparsely) with monsters that are obviously there for no particular reason except to be there. This space serves little gameplay purpose except to separate more interesting locations.”

immersion breaking behaviors should players deviate from anticipated AI design assumptions. A trigger may be the common example of a scripted cheat: "that tank wasn't there a minute ago, but because I crossed a magical red line... poo!" The poo may work the first time, and it may continue to work so long as the player agrees to play the scenario in the ways the designers intended. Invariably, however, it is doomed – at some point the player will see "poo" as an ornament to be gamed but not played.

Perhaps, the less-scripted a cheat feels – the less it is tied to a specific outcome, or specific place, or specific time in a game world -- the more likely it is to be accepted as part of the world’s rule set. An example here is with how Everquest simplified path-finding for monsters. It was often ignored: walls in small buildings and trees may not block a blow for a monster, but it could block a player. The purpose of this simplification was to lighten the server impact (a common objective for MMOGs). Because the application was ubiquitous - few players saw this as a cheat per se.

As games have become more immersive, players demand AI opponents that are more "like" other players: almost as good as me, but not quite. Players want to "barely win". It is well understood by AI developers of robot adversaries, or "bots," for first-person shooter games, that the key to a good "bot" is not that it be a superman (that the player can’t beat), but rather a challenging and lively AI. In other words, "gamers are not interested in the ability of a bot to be truly intelligent, but rather put up a good fight (Moyer)" and is "entertaining to play with and against. (Atkin)."

A common pattern of player interaction is based upon "reveal as you go." Adventure games provide a prototypical example: the player explores a world and interacts with its contents, in pursuit of some objective. A player proceeds using trial and error and game designers need to fabricate self-contained chains of storyline. In MMOGs, these are often called Quests. Typically, NPC behaviors are subservient to the Quests – this introduces another element to the scripted NPC AI cheat: the behavior, if not outright existence, of an NPC becomes predicated upon a fragment of storyline requiring to be unlocked.

Most NPC behaviors are themselves "script-driven" instead of "rule-driven". Rules are declarative representations that, given variables in the game, encode relationships and facts about the game. From these facts one can reason completely about the game world (what is true about it or not). Scripts are imperative representations — they are instructions to process game variables and compute some conclusion. Scripting is more intuitive for game designers and developers to work with. It is easier to conceptualize and write a script that says: "go right, go left, turn around twice, go straight, then fire your gun..." then it is to formulate a set of rules that could shape the entities movement to the same effect.

For example, one MMOG (ref Eve Online) boasts on its website a game engine that can execute a very large numbers of scripts (using Stackless Python – a programming language, see Cameron). Their claim, implicitly, is that because their game developers can write lots of little scripts for specific game objects very quickly, their world is more interesting. This approach would contrast with an approach based on a general rule-based model of behavior.

While scripting can lead to sophisticated behaviors, typically these behaviors are brittle outside of the context for which they were designed. Computer components can roll out of the way of grenades and fire beneath tables, or hide cavalry in tree lines and flank your advance at very inopportune moments. These
behaviors are realistic only so long as the player performs within anticipated parameters. Furthermore, these behaviors are ad-hoc: simple to do but hard to generalize. A developer need only say "turn right, turn left..." versus develop a reasoning model for evading attacks, say. When it comes to the AI, even if a player is not explicitly held to a single pathway and allowed to roam around, a world of scripts means that the content of the world is less a live world than a world of linear mini-plots awaiting to be unlocked.

One example is Morrowind (Elder Scrolls) - a gorgeous adventure game constructed on a vast game-world. Yet, its playability is ultimately limited by the number of plot lines that have been developed and placed in the world. When the townsfolk are not participating in some storyline, they march about witlessly and without any other role. It would be far superior if they could participate, somehow, in a game world of creation – where new tensions, new stories, new detail emerge from the backdraft.

The shallowness of games AI seems to have made MMOGs particularly vulnerable when compared to the evolution of the single player game. In part this is because of a deliberate choice by MMOGs to push back to the players more of the responsibility for creating their in-game experience. Developers have opted to streamline MMOG AI in order to support greater player server head-counts as well as to manage team/collaborative play. Typically this means fewer AI entities (NPCs) as well as simplified behaviors. The result is that MMOG AI often feels blandly incompetent (e.g. "Vending Machines", see Reynolds, 2004). Single player games can also control the player involvement and focus more tightly. With single player games...

“(Players) cannot do anything else (than follow the limited set of actions asked of her to perform)... in a martial arts fighting game, I can’t ask questions of my opponent, nor do I expect him or her to start a conversation with me. All I can do is “attack” my opponent by pressing a few buttons, and within this highly codified situation the computer can “fight” me back very effectively. In short, computer characters can display intelligence and players. Robert Zubek described on his weblog:

“Aaron Khoo had an extreme example of how good AI could go wrong. One of his take-away messages from Mythica was that in-game combat was really a multiplayer puzzle involving finding good solutions to the different aggro problems (in the MMO parlance, the tactical problems of how to approach and attack different groups of enemies). But for this to work, the aggro AI had to behave in consistent, understandable, genre-cliched ways. Paradoxically, better AI would have changed the nature of the game, and been detrimental to the players’ enjoyment of the tactics puzzle. “


---

11 At the 2004 Workshop on Challenges in Game AI, Aaron Khoo described how MMOGs handicapped AI to meet server player density requirements as well as to support multiplayer play balancing. He described how they had to simplify monster attack strategies so as to not overwhelm...
skills only because programs place severe limits on our possible interactions with them.” (Manovich, pg 34).

Perhaps MMOGs have reappropriated the AI away from direct in-game interventions into tools and intelligence to help developers with analysis and play-balancing issues (e.g. suggested by Carpenter, Kennerly). Or at least MMOG developers may be thinking more intensely along these terms. Because of the issues of scale (e.g. degrees of freedom, number of players), a different kind of analysis of game play may evolve – likely a more intelligent one, one which models the players as well as their culture.

Another view is that MMOGs have not relinquished the AI but instead use players to directly play the AI as designed by the developers. In other words, MMOGs are not retreating from the AI challenge, they just rely upon people to implement the AI. Dehumanizing the human opponent in “Player versus Player” (PvP) games is one instance; by making the opponent player forces appear more like the NPCs it depersonalizes the conflict. Dark Ages of Camelot, for example, masks an opposing team member’s identity and restricts communication in the Realm-vs-Realm (PvP) areas.

The problem, however, is that players are at least unpredictable, and often pathological, in their behaviors (a point well made in T.L. Taylor’s discussion of Power Gamers, 2003). Furthermore, the numbers of players in MMOGs who profess to purely “role-play” tends to be quite small. On the one hand this means that players are cannot reliably assume responsibility for shaping the story. On the other hand it also suggests that the players are in need of game worlds with AI that can help organize the dynamism inherent to an MMOG and mould game experiences beyond predictable treadmill and socialization patterns. This suggests a directorial AI presence.

**A DIRECTORIAL INTELLIGENCE**

This paper has suggested that evolving to declarative AI paradigm in games and MMOGs is a necessary step for the future. However, it implies a difficult transition because of the developer culture as well as the current cinematic grounding of the design community. This paper has pointed out that the cinematic language of computer games is extensible, and suggests declarative extensions to the cinematic language of MMOGs are possible. But what does a declarative cinematic language of behavior entail?

It is suggested that this declarative form of AI will be *directorial*, based on a language that:

- Specifies rules and goals for the actors in the world
- Procedural only when necessary

If a directorial AI is the future of MMOGs, and MMOG AI is strengthened and fleshed out beyond its current pale forms – does this suggests a new contract between player and virtual world? Might this lead to a world where the players and the AI become co-stewards of the world they act in – each participating as well as amplifying the other?

MMOG plotlines, as well as other computer games, are for the most part the sum of their scripts, triggers, and waypoints. As this paper has suggested, this is due, in part, to the strictures of a strong cinematic viewpoint. As this viewpoint evolves so to will its imperative flavor weaken. What then? What form the AI will take in the future for our large virtual worlds? This paper imagines the following epoehal steps in the future development of AI for MMOGs.

---

12 See footnote 6 regarding functional languages – they represent a useful hybrid between declarative and functional forms.
The First Step: A Simulationist Future

A necessary step towards building worlds capable of generating interesting narrative subplots, is to build a large enough world with sufficient detail and apparent dynamism to engage the player. One can imagine a simulated society of NPCs, whose individual actions and behaviors combine and cascade in a vast number of ways… However, this vision is insufficient in that the events themselves may not present interesting choices to the player – they may not make good narrative fodder. This vision is necessary, however, because in order to achieve good narrative fodder for large, evolving MMOGs, the potential of the world (in terms of content) in the world will likely need to exceed the quantity that is actually consumed. Why is that? The surplus is useful to buttress the illusion. Consider:

“To create a convincing chicken, it helps if you have a flock.”

The attraction of a simulation foundation to an MMOG world is that it can give us the quantity of content required; it may give us a live flock offstage and more options exist for presenting a chicken as part of a storyline to a player. John Arras stated this perspective well:

“The problem with stories is that they are explicitly enumerating one possibility out of a vast world. If the goal is to create a long-term persistent game where thousands of people can influence the world, then it’s not going to work starting from the premise of a story: of a single enumeration of possibility. That's why "branching narratives" …will fail. It won’t be possible to enumerate a vast enough number of possibilities that players will feel like they’re really interacting with a living world. People are not stupid and they can tell when they’re severely limited and "riding on rails" rather than having freedom.

With a simulationist approach, creators build objects that fit into a system and the interactions won’t be predictable. This is very difficult to do, and I am not saying that I have the answer, but I think starting from the wrong approach will doom many efforts to failure because they won’t scale.”

Weblog discussion (Combs, June 2004), suggests there are two basic problems with respect to managing the objects of a simulationist world. The first part of the problem is mustering the right set of game objects (right time and place) to act out a scene. (As in Henry V, Shakespeare, to insure one has “A kingdom for a stage, princes to act. And monarchs to behold the swelling scene.”). The second part of the problem is getting those objects to convey the right dramatic meaning to the player (e.g., the meaning of "group of four goblins in the forest"). This first step, as I outlined above, only addresses the organization part of the problem. The next step needs to look at the dramatic context…

The Second Step: The Dynamic Difficulty Adjustor

Beyond getting the right game world objects together, the next step calls for an AI that can assist the player experience. One possibility might be along the lines of Robin Hunieke’s proposal of a Dynamic Difficulty Adjustment (DDA) system – an AI that can dynamically adjust the difficult and challenge of NPC engaged in tactical combat with a player.

Robin characterizes prior commercial DDA approaches as adopting two strategies: manual annotation (of game tasks and artifacts) and then evaluation; and off-line analysis based on data mining (e.g. Kennerly). Both of these approaches, however, are oriented towards
design evaluation/ play balancing, rather than dynamic alteration of play once commenced. A true DDA system would be exposed to two challenges. First, there is the issue of perspective posed by Richard Bartle (2004):

“This kind of tinkering is equivalent to making the virtual world adapt to the player, rather than the other way round... As for the effects when several players try to interact with the same object, dynamic difficulty is really going to make things look weird to some of them.”

The other issue is related: are we confident that optimizing local criteria of challenge and difficulty – e.g. inventory theory (Hunicke) – will lead to better dramatic arcs? It could make for more challenging encounter, or subgame, but does that necessarily translate into a more complete narrative? A generalization of DDA might correspond to an “Analytical AI” proposed by Alex Champandard. This would be an AI that:

“...is intended to enhance the game experience, but not through a direct interaction between the player and the characters...might be used to run the traffic lights in a racing game, or control the number and difficulty of Combat AI opponents. It could monitor the player’s health and reaction times and then adjust the damage taken or the speed of the opponents... (it) could provide subtle hints as to where the player should go next, such as making noises emanate from a room that the player is supposed to enter.”

The Third Step: The Game Master.

The First Step describes a simulationist world as a rich platform for building emergent game worlds. A large world filled with detail gives an AI room to maneuver and build engaging narrative structures for players. But potential is not enough. Identifying the narrative pathways is the next challenge. A DDA system might offer a means of embellishing the player’s experience, once the narrative pathway have been established an selected, but they do not in themselves provide a means to create new story arcs. Interestingly, Phoebe Sengers argues more broadly that the “juice” that binds an AI “coherence of action over time” (pg 259) is narrative. Without it, Phoebe suggests that a traditional “divide and conquer” strategy towards AI leads to a design under-integration and an AI behavioral Schizophrenia. The claim of this paper is that to move from the Second to a Third step is similarly necessary.

The First Step move us towards a virtual world which is intrinsically engaging to the player. The Second Step suggests how we might tune narrative paths through that world. However, there is still a shortfall between the potential of a simulationist world and what can be actually achieved from the First and Second steps: what AI will discover and guide the player along new paths through our brave new world?

To this end, let’s hypothesize a Game Master (GM) AI whose role is to create challenging encounters and engaging storylines. The GM’s objective would be to construct and manage larger narrative structures. Good drama is more than just hard monsters. Rambling storylines are not sufficient, nor are ones composed of orthogonal subgames or subplots13. How would a GM interact and shape the behavior of the NPCs? Ian Wilson

(2004) suggests dynamism and “AI role-playing” as foundational elements:

“The game masters primary means of interaction with the players would be through the use of NPCs. What this means is that the NPCs require sufficient behavioral abilities and flexibility to be able to act convincingly (according to their roles) in a very wide variety of highly dynamic situations, in real time without prior knowledge (i.e. a script).”

What are the important ingredients here:

- How adaptable should the AI be?
- How expressive should the AI be?
- How declarative should the knowledge representation be?
- How autonomous should the pieces of the AI be?

It is axiomatic that a good human “dungeon master” (e.g. Advanced Dungeons and Dragons) needs to be good at adapting their narrative to the players. Dungeon masters respond to player cues about their transient as well as long-term (persistent) interests; they can improvise subplots (as well as avoid them) based on learned player interests: e.g., this party is (not) interested in tavern and random encounters – they (do not) like to stick rigidly to the main plotline, etc.

Beyond the GM AI needing to be adaptable (i.e., capable of learning about player interests), it also needs to be *expressive* in the sense given by Domike et al (2003) – an “expressive AI” is an AI that is perceived as intelligent in its cultural context (see also footnote 8). The question then becomes, how sensitized should the GM AI be to the social and cultural norms of the world? An AI that can change with the evolving cultural norms of the world would be more robust but also much more difficult to develop.

Implied by all these considerations are basic choices about form and determinism. For example, should the AI be fragmented into agents (e.g. NPCs), and should those NPCs be strongly autonomous (e.g. individuals capable of “reactive planning” as intended by Mateus and Stern, 2002). A contrasting view may see NPCs as mere proxies for an overarching directorial AI – a world whose intelligence and planning is integrated and strongly managed. Somewhere in between lies the possibility of NPCs that embody a duality of purpose. They try to satisfy local goals -- local to the agent and the nearby players -- while coordinating with higher level storyline objectives (e.g. per Magerko et. al’s notion of character “Directability”).

Another formative question is the degree to which the GM AI should be declarative – to what degree should its knowledge be represented separately from the world and contained within the expression of the AI system. This has implications on the potential of the AI in terms of its expressiveness and adaptability: The greater the degree that knowledge of the world, its culture and its individuals can be directly manipulated and reasoned by the AI system - the greater the opportunities for an expressive but yet extensible AI.

In terms of directorial considerations - Magerko and Laird (2004) proposed an AI director (Interactive Drama Architecture) that could mediate between the players and the plot. Their directorial AI would monitor plot and recognize errant player behavior as well as the reconcile such behavior with the overall plot design. The GM vision is a generalization of this sort of directorial vision: a GM would also juggle and reconcile many concurrent strands of direction and plot, each at a different level of maturity.

Whether it is Lars Konzack’s seven layers of computer game analysis (2002), or Nick
Montfort’s five levels (2004), it is necessary for the diverse elements of technology, players, and culture to interact to create an expressive game AI. Whether the elements are hardware, program code, functionality, gameplay, meaning, referentiality, or socio-culture (Konzack), what is important is that a GM AI should be considerate of the many levels a player may perceive the world and its narrative.

From the perspective of an expansive and growing cinematic experience, we suggest that a cinematic language that embraces MMOGs can (and should) expand to co-opt non-linear, goal-driven, and socially driven views of behavior. Just as traditional cinema has developed structured styles for relating stories and certain types of events to viewers, film idioms, so too might new kinds of idioms be developed for the cinematic language of MMOGs.

For example, perhaps someday, a GM AI could collaborate with a camera-controller AI\(^\text{14}\) -- leading to a symbiosis between behavior and perception of behavior in a game world. Consider a case where NPC actions are highlighted (e.g. dramatized) and reinforced using film idiom camera angles.

\(^{14}\) Camera-controller AI for interactive narratives in 3d game worlds (e.g Amerson and Kime) offers an interesting contrast to “GM” or world-behavior AI. In the former case the intelligence is one step removed from the world -- focused upon developing a presentation idiom. See Real-Time Cinematic Camera Control for Interactive Narratives by Dan Amerson and Shaun Kime, in The Working Notes of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment, Stanford, CA, March 2001.

**CONCLUSIONS**

This paper started out with the suggestion that a more declarative and directorial AI style of describing behavior in games and MMOGs will emerge within the games industry. For reasons of economics and scale, we may first expect MMOGs to transition into a less-scripted and statically delineated universe and then into a more simulationist AI future. Finally, this paper closes with the speculated form of the ultimate declarative goals-centric MMOG AI – a Games Master.

Ultimately, this paper maintains that future MMOG worlds may benefit from greater use of simulation – doing so will likely pose fundamental and difficult questions about story-telling, role-playing, and the autonomy and role of the Artificial Intelligence. We may anticipate future tensions between the simulationist world and a directorial vision of AI in MMOGs – these will in turn stretch the cinematic language of games further. These, however, will likely be creative tensions that push our explorations of the virtual world and the intelligence that resides within it.

**REFERENCES**


Carlquist, Jonas. “Playing the Story Computer Games as a Narrative Genre.”
http://www.hb.se/bhs/ith/3-02/jc.pdf
[Oct 14, 2004].
GameDevelopers Conference. March 22-26, 2004. San Jose, CA.
Herdman, S. J. “RuthMoo: Programming the Environment.”
Koster, Raph. March 6, 2004. Comment to “Can We Save MMOGs from Ourselves (using simulation).” Terra Nova weblog.
<http://www.onlamp.com/pub/a/python/20


NetLogo. <http://ccl.northwestern.edu/netlogo/>


Unified Modeling Language. IBM website. <http://www-