Cell-based Music Organization in *Tom Clancy's EndWar*

Ben Houge
Assistant Professor, Berklee College of Music, 1140 Boylston St, Boston, MA 02215
bhouge@berklee.edu

Abstract

Tom Clancy's EndWar, a real-time strategy game developed by Ubisoft's Shanghai studio for Xbox 360 and PlayStation 3 and released in November 2008, featured an innovative, cell-based music deployment system that allowed for extreme variability and tight responsiveness without sacrificing production values. This paper describes the core functionality of this system in the context of the game.

Introduction

The emergence of optical media as a distribution format for video and computer games in the 1990s allowed for a substantial increase in the amount of digital audio that could be incorporated into a game (Collins, p. 63). Since then, there has been a tension between fidelity and variability in how game music is organized and deployed. Systems like LucasArts' iMUSE (Collins, p. 51) and Microsoft's DirectMusic (Marks, p. 368) allowed for intricate, note-based manipulations in real-time, but this flexibility came at a price: the game engine had to include a sample-playback synthesizer running in real-time, and any digital signal processing (e.g., reverb) similarly had to be performed in engine, on the fly. On the other end of the spectrum, games like *Half-Life* (1998, Valve/Sierra Entertainment) and *Diablo* (1996, Blizzard Entertainment) featured high quality studio recordings, but with minimal real-time manipulation, such that music tracks would play unvaryingly for long stretches of time, regardless of what was happening in the game.

When I set out to design a music deployment system for *Tom Clancy's EndWar* (2008, Ubisoft), my goal was to explore the middle ground between these two extremes, having had extensive experience with both paradigms. Most of the games to which I contributed music at Sierra from 1996 to 2003 incorporated a simple, loop-based music deployment mechanism; a track would loop as long as it was required to play, and when a transition was required, it would crossfade to another track. These games included *Arcanum: Of Steamworks & Magick Obscura* (2001), which featured a string quartet soundtrack performed by members of the Seattle Symphony; *King's Quest: Mask of Eternity* (1998); and *Leisure Suit Larry 7: Love for Sail!* (1996), which featured a jazz ensemble, among others. I also developed a highly interactive soundtrack using DirectMusic for Sierra's cancelled Xbox title *Johnny Drama* around 2001, employing many of DirectMusic's advanced features, such as chord maps and scripting.

On *EndWar*, which I served as audio director from early 2005 until late 2008, I required a music system with high production values, tight responsiveness, and wide-ranging variability, to follow the emergent contours of a real-time strategy game with high replayability and wide open outdoor environments. I also sought to eliminate loops and fades, two conspicuous game soundtrack signifiers.

System Overview

One of the game's key audio innovations, the *EndWar* music system features what may be described as a cell-based musical organization. A "cell" in this context is analogous to a drum loop in many ways, except that it does not loop. Like a drum loop, a cell is a digital recording of a short musical phrase, several beats in duration. For flexibility, a cell typically contains a stereo recording of a single instrument. However is not edited such that its duration is an exact number of beats, as a drum loop would be, and its last sample is not designed to lead seamlessly back into its first sample. Rather, a cell's practical musical duration (in beats at a given tempo) is defined manually and stored in metadata. When a cell is called to play, the corresponding wave file is cued, and when the number of beats specified in the cell's metadata has elapsed, the cell is considered complete (whether or not the wave file is still playing).

This allows each cell to continue playing until the natural terminus of the recorded sound; it need not be edited into a seamless loop. A cell may be shorter than its defined beat length (as in the case of a wood block) or longer (as in the case of a gong).

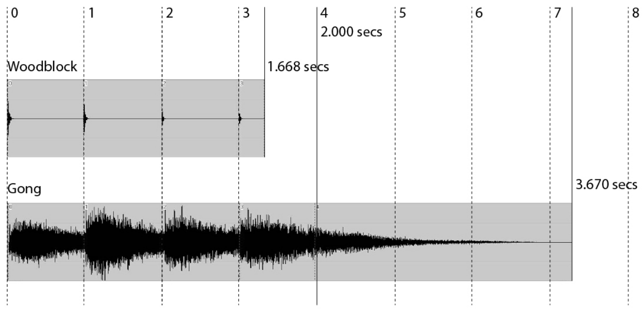

Fig. 1: Four steady beats on a woodblock take up less time than the same rhythmic pattern performed on a gong, but both cells are assigned the same duration of four beats. Tempo displayed is 120 bpm.

Cells are arranged into "pools." When a pool is told to play, its constituent cells are called in random order (with no repeats, if there are more than two cells in the pool). If a cell's length in actual time exceeds its assigned beat duration, the remaining audio simply continues to play, mixed in with the beginning of the next cell. Since no sample-accurate concatenation needs to occur, it is very straightforward and idiomatic to create the effect of a loop with variations, which would otherwise require extensive editing and testing, to ensure that the end of each variation could transition seamlessly into the beginning of every other variation. When the pool is told to stop, it simply plays the current cell to completion; there is no sudden truncation or artificial fade out.

Pools take parameters for minimum and maximum pause (in beats). If these values are set, then a random number of silent beats between these two limits is played between cell repetitions. In my observation, there is huge potential for musical development in dynamically modifying these minimum and maximum pause values, allowing for a passage's density to be altered without changing its fundamental material. Unlike many game soundtracks, in which transitions are a matter of stopping one piece of music and starting another, this system allows the music itself to evolve in a very natural way, changing from the inside, as it were, without drawing attention to the transition.

Several layers of pools may be played at the same time. With each pool containing a variety of different cells (which of course may be of different beat durations), and with varying pauses between cell repetitions, a rich, non-repeating texture can be rapidly constructed. And because the constituent cells are relatively short in duration, the system still has the ability to respond quickly to emergent game events without resorting to crude crossfades.

Pools may be nested. Several pools may be placed inside of a parent pool; a pool inside of another pool is referred to as a "subpool." A subpool, in addition to its minimum and maximum pause parameters, also takes minimum and maximum repetition parameters. When a subpool is called, it will choose a number of cells corresponding to a value between its minimum and maximum repetitions; when that number of cells has been played, the parent pool's minimum and maximum pause parameters are invoked, after which another subpool is chosen. Of course, there may be several layers of these nested subpools playing simultaneously. (This behavior was partially inspired by the slow, overlapping evolution of parts in Morton Feldman's composition *Why Patterns?*)

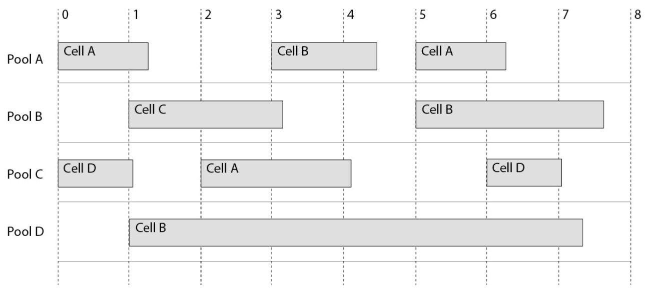

Fig. 2: Several layers of pools, each intermittently deploying a variety of cells, all quantized to the same pulse, result in a rich, evolving musical texture.

Generally the goal was for different layers of subpools to transition independently of each other, to maximize the combinatorial potential of the system. However, in a few situations, we wanted to allow two or more subpools to transition at the same time. In this case, one pool could be specified as a "master pool," and when its parameters caused it to transition to another section, other pools could be set to transition at the same time.

Production Pipeline

To develop this system, I first developed a prototype in Max/MSP that would play back percussive phrases I had recorded on traditional Chinese instruments. At the same time, we were fielding demos from composers for *EndWar*, and we eventually settled on Alistair Hirst and Matt Ragan of Omni Audio (then Omni Interactive Audio) in Seattle. (The Omni Audio team also provided most of the sound effects for *EndWar*.) Once I was happy with my music prototype and the composers and been chosen, I flew to Seattle to work with them for a week on the development of three different pieces of music, which I took back to Shanghai and set about implementing into my Max/MSP prototype.

After the music deployment mechanism had been approved with final music in place, I worked with the lead audio programmer on the *EndWar* team, Wang Yichen, to implement my Max/MSP prototype in C++. Having worked through the system design in detail in Max/MSP, the transition to C++ was smooth, and I was able to provide guidance and serve as an ideal tester. We devised a simple tool that allowed me to write and read the musical metadata (stored in XML format). I also wrote another

Max/MSP patch to allow me to very efficiently audition and determine the practical duration of a given cell in beats.

I provided creative direction to Hirst and Ragan as they composed new pieces, and they would send MP3 mockups of how a new piece might play out in context. Once a piece was approved, they would bust the tracks out into individual cells and send them to me, and I would integrate them into our game engine using our custom tools. As with any system incorporating random variables, there was a significant amount of time involved in testing and tweaking the random parameters of each piece in context.

In total, Hirst and Ragan composed nine different sets of music. Each set was used in about three different levels (*EndWar* featured about forty levels in total), and there was a special piece of music for the main menu screen.

Technical Infrastructure and Constraints

EndWar used a highly modified version of the Unreal 3 engine. We gutted the Unreal engine's audio system and replaced it with Ubisoft's proprietary DARE audio engine, which was significantly more full-featured. DARE provided the requisite abstraction for defining and categorizing all of our sound effects and also provided all the background loading and file management.

The music system can be considered a high level system built on top of DARE. We bypassed individual event definitions for each musical cell, but used DARE for the low level loading and streaming.

Because the *EndWar* music system required many individual audio files to be mixed together in unpredictable ways in real time, it was not feasible to stream all of these files from the DVD. Instead, contrary to convention, most of the music data was loaded into memory. This might seem to be a major drawback of the system, but because we could set data compression values on each cell individually some cells were set to be extremely compressed, allowing us to fit a level's worth of music into less than 2 MB of memory. In fact, in some cases, the distortion artifacts of compression were exploited as a desired aesthetic phenomenon, supporting the game's ethos of gritty realism. Since cells with varying compression formats were being played together, the overall effect was of a high quality mix containing distorted elements, not of a low quality, overcompressed soundtrack.

We did reserve the option of playing two streaming stereo layers from the DVD. These were used as arrhythmic background pads, mostly comprised of sustained electric guitar sounds submitted to various (offline) granular synthesis treatments. These were deployed using the same cell/pool hierarchical organization, but they were typically longer than other cells, in some cases lasting up to one minute (as in the main menu music). Usually only one of these streaming cells would play at a time, but the ability to play two streams at once allowed for the possibility of overlaps at transition points. Due to their length, these cells could be faded out in certain circumstances.

From a compositional perspective, a certain understanding of the manner in which the material would be deployed affected the types of phrases that could be written. This type of modular organization was something the composers quickly internalized. Practically speaking, although not enforced by the system itself, each level's music tended to be written in one consistent mode, to accommodate a wide number of possible configurations. Most critically, all music for a given level had to be composed at one consistent tempo, although we did experiment with using the beat as a common denominator for metric modulation.

Stylistic Considerations

EndWar took its inspiration from contemporary battlefields, which is why we chose to pursue a rock aesthetic, but we sought to create an expanded rock palette abstracted from conventional song forms that encompassed a wide spectrum of intensity, from dense, energetic passages to quiet, ambient textures. Not only did this better match the range of emotions the soundtrack was designed to support, but it also evoked the a sense of exploration that is central to the real-time strategy genre. The whole system is predicated on a continuous, metronomic pulse to which all cells are quantized, but the system has no concept of measure or meter. A major inspiration for the soundtrack was the freely improvised rock of the trio Massacre (Fred Frith, Bill Laswell, and Charles Hayward) in which long phrases spin out over a steady pulse, without implying a regular bar line. This stretching, groping quality seemed a good fit for the gameplay of *EndWar*, and it was a conscious move away from the inherent periodicity of loop-based structures. In fact, even in such a highly dynamic system as DirectMusic, phrase variations recur at regular intervals, illustrating how hard it can be to escape from this pervasive looping paradigm.

Related Projects and Future Work

The *EndWar* music system has no direct descendants at Ubisoft; *EndWar 2* was cancelled after a few months of production, and to my knowledge no subsequent game has used the system. If the system were to be revisited in a game, I would be interested to see it expanded to allow for several hierarchical layers of nested pools (not just the two levels implemented in *EndWar*, although that already provided significant terrain to explore). The underlying metronome could be expanded to allow tempo fluctuation within one level. There is also much potential in the idea of incorporating real-time pitch manipulation, allowing greater reuse of elements and reducing memory usage.

I have continued to explore several of these ideas in subsequent works outside of the game industry. My six-channel sound installation *Kaleidoscope Music* (2009, first presented at Beijing's Today Art Museum) algorithmically generates rhythmic phrases that are intermittently deployed in a manner similar to *EndWar*'s cells (although there is no prerecorded audio in this piece, only filtered environmental sound, streamed into the computer in real time). Another piece entitled *Mobile 4* (2011, premiered at the San Diego Museum of Art) also explores real-time phrase generation, this time displayed in scrolling notation on networked laptop screens for acoustic musicians to interpret. And in *Food Opera: Four Asparagus Compositions* (2012, premiered at Harvard's Graduate School of Design), a collaboration with chef Jason Bond of the restaurant Bondir, I composed a real-time algorithmic soundtrack for a four course, asparagus-based tasting menu; here again, several layers of short phrases were intermittently deployed to create a rich, evolving sonic texture. I believe this mode of non-linear musical organization represents fertile soil for continued compositional investigation.

Acknowledgements

Thanks to the audio team of *EndWar*: Yassine Abouelfalah, Ou Yuan Jun, Yu Hai Xiang, Ding Jun Mei, Du Ke Jia, Song Wei Wei, Wang Yi Chen, Chen Ai Qi, Wang Ge, with additional help from Zhang Lei and Romain His. Thanks also to creative director Michael De Plater, producer Audrey Leprince, and associate producer Nathalie Paccard, for their vision and support in this somewhat experimental venture. Thanks to AI programmer Vinh Dieu Lam, who was critical in actually getting the music system to respond intelligently to the rest of the game.

References

Collins, Karen. 2008. *Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design*. Cambridge, Mass.: The MIT Press.

Marks, Aaron. 2009. The *Complete Guide to Game Audio*. Burlington, Mass.: Focal Press.

For more information about audio development for *Tom Clancy's EndWar*, including video clips of the music system in action, please visit http://www.benhouge.com/writings/?p=628.