Video Game Audio Prototyping with Pure Data

Leonard J. Paul[1]

[1] Vancouver Film School, 1380 Burrard Street,
Vancouver, BC, V6Z 2H3, Canada
{info@lotusaudio.com}

Abstract. The seventh generation of home video game consoles hold great promise over the previous generation for new interactive real-time audio techniques to provide new levels of audio immersion for the player [4]. With this additional complexity comes the difficulty of developing audio behaviors that will best support the artistic goals of the game sound designer. Prototyping allows the game sound designer to use a set of tools to rapidly experiment with interactive audio generation without the involvement of a game audio coder. This rapid process of real-time audio experimentation is necessary to suit the restrictive bounds of most game development schedules and budgets. This paper investigates the tools readily available to anyone working in game audio who wishes to challenge the capabilities of the current video game consoles.

Keywords: Video game audio, interactive audio, adaptive audio, game coding, game prototyping, video game, PD, Pure Data, Open Sound Control, OSC.

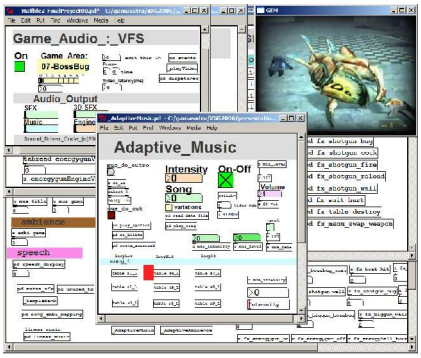

Fig. 1. Screenshot of software used to demonstrate game audio prototyping using Pure Data.

1 Introduction

The seventh generation of game platforms grants the game sound designer new freedom and power in interactive audio, but new knowledge is required to effectively harness the audio capabilities of the current platforms. With platforms such as Microsoft's XNA [16] and Sony's use of existing standards such as Open GL and Collada[14], the industry has come to realize that it is not only a challenge to utilize

the advancements in hardware but also provide a platform to accelerate and ease the learning-curve on the game development process.

1.1 Prototyping

Prototyping helps to prove ephemeral characteristics of games such as game-play, replay value and other qualities difficult to review without the use of an interactive prototype [19]. It's a good approach to view the prototype as a toy and have fun with it [10]. Prototyping allows the attrition of features to occur earlier in the project which helps to focus the game design and saves valuable development time. It can take a while to learn how to effectively prototype game audio, especially for sound designers coming from a film background as prototyping is largely unnecessary in film audio work. As with any prototyping, one should try not to try to force the prototype to evolve into the final product, as it should be rebuilt to become more robust [19]. When prototyping entire game ideas it was found that not being afraid of failure, not spending too much time on the prototype and finding several different ways to solve a certain problem were highly effective techniques, which could similarly be applied to prototyping game audio [10]. One of the most important choices when prototyping is the platform chosen to realize the prototype.

1.2 Pure Data

Pure Data (PD) is an open source cross-platform graphical programming language for Windows, Mac OS X and Linux [21]. PD is a powerful rapid prototyping environment for audio in comparison to C++ as the learning curve for artists such as game sound designers is much more gentle with PD. Using PD can give the game sound designer additional creative control as well as gaining valuable skills and vocabulary to better work with sound coders when implementing the prototypes.

Optimization of an interpreted language such as PD is of prime concern for games, but with research being done into the compilation of PD code, such as the compilation of PD code to Java in the pd2j2me project[24], the issue of optimization becomes less of an issue. The PD engine can be quite easily integrated with other projects as shown by the PD internet browser plug-in[1] and the PD VST plug-in[22]. If direct code integration isn't an issue, then the advantages of commercial packages, such as Max/MSP[6] and Reaktor[18], are that the documentation and support for their products can make the learning curve easier than PD for some game sound designers.

1.3 The Future of Game Audio

In the future, the distinction between prototyping and implementation may fade altogether as audio implementation tools begin to more closely resemble the tools previously reserved for prototyping. Game studios are beginning to realize how important defining the behaviors of game audio is, as expressed by Clint Bajakian, currently Senior Music Supervisor at Sony Computer Entertainment America, "Artist creates conditions, rules and procedures, not necessarily the audio itself"[3]. This

notion of creating generative audio is also being currently explored by well-known musicians such as Brian Eno for Will Wright's game *Spore* [13]. It is even possible that PD is being used in the process of making *Spore* as the Electronic Arts studio that is developing the game was "looking for PD people"[7].

2 Technology

To demonstrate the possibilities of audio prototyping for console games, a sound driver has been implemented utilizing PD. The initial version was developed in 2003 [19] and has enjoyed continual development by the author as part of the game audio curriculum at the Vancouver Film School.

2.1 Pure Data Sound Driver

The current version of the software supports the playback of sound effects, speech, adaptive music, engine loops and outputs 5.1 surround sound. The surround sound portion of the software supports volume and high frequency attenuation by a user-defined roll-off curves and doppler-shift. Although the current PD sound driver isn't currently publicly available, an older version of the code can be found on Gamasutra.com [19] and the surround sound support can be found online on the author's website at VideoGameAudio.com. This software has been built for instruction and to allow maximum transparency by being coded entirely in PD. In industry, one would only need to implement the areas in PD that one wished to prototype and have the existing sound driver produce the remaining sound.

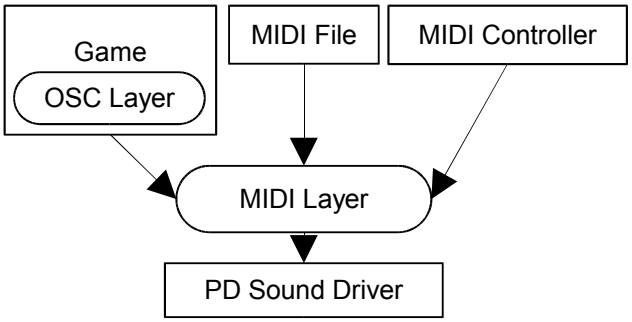

Fig. 2. Game event data flow figure showing the possibility of input from the game, a midi file or a MIDI controller.

2.2 Event Input

Figure 2 shows the different options of sending events to the PD sound driver. The events that drive sound generation could come from a running game (such as a *Half-life 2* game modification) that has an Open Sound Control (OSC)[5] layer added to it

which communicates the game parameters via network to the sound driver. The sound driver can also be triggered by a MIDI file that has specially coded events that simulate game events in synchronization to game captured footage. This method has been used for some time for instruction as it has the advantage that a game is not required to run in parallel with the game audio code. The obvious disadvantage is that the generation of audio is linear, even though the audio is being generated in real-time similar to an actual game. The last method of input of events is by a MIDI controller which could likely be used best for debugging and mixing.

The manner in which the student projects are realized are very similar to the process that industry goes through when prototyping. In both scenarios, students and professional game sound designers are working on a relatively small-scale project for the purposes of learning under time-pressure.

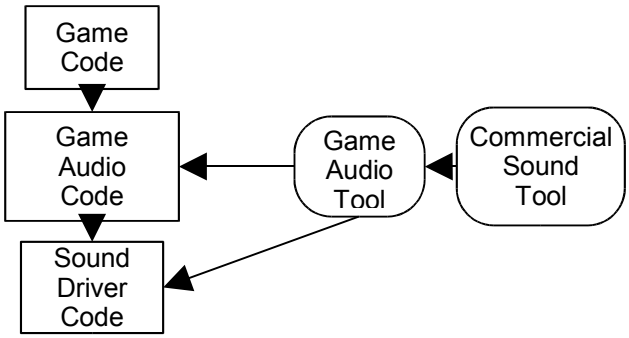

Fig. 3. Game event data flow figure showing the possibility of input from the game, a midi file or a MIDI controller.

2.3 Game Audio Tools

A commercial sound tool such as Steinberg's *Nuendo*[26] or Digidesign's *ProTools* [8] is used by the game sound designer to create content. The audio files are brought together in a game audio tool such as Sony's SCREAM[25], Audiokinetic's Wwise[2] or Microsoft's XACT[16]. Depending on the level of functionality of the game audio tool, they allow for different levels of definition of an actual sound's behavior. The more of the sound's behavior that the game sound designer is able to describe, the less specifics the game audio coder will need to guess about. The nature of the definition of the sound's behavior can range from a static description (such as the amount of pitch shift to randomly utilize) or allow for much more detailed dynamic scripting of a behavior (such as how the engine loops transition between one another in a car engine model). Most current game audio tools are best at describing parameter ranges rather than allowing for the definition of dynamic behaviors. PD has an advantage in that it is able to implement a large number of possibilities with a high-degree of flexibility. The drawback of having too much flexibility is that it can be difficult for the game sound designer to get interesting results without spending a significant amount of time learning PD. However, once a few good prototypes are constructed it is easy to reuse elements and create a library of commonly used elements.

3 Conclusion

To this date, the PD sound driver software has been driven by a MIDI data stream that triggers game events in real-time in conjunction with captured game-play footage. The obvious extension to this would be to drive the sound driver by real game events in response to playing a game. Work is currently being done to integrate an OSC layer into a *Half-life 2* game modification which could be ported to most other similar mods with little difficulty. If the events and the game visuals were recorded by the PD engine, the valuable diagnostic of being able to review game audio events at a refined level is a very useful tool. Another enhancement would be to have the sounds driven by real-time midi controller data such as a keyboard or a mixing surface. The recording of the mixer data can be recorded to provide mix presets that can be recalled by the game to suit particular audio mix situations, such as ducking the music and sound effects during a cut-scene or adding reverb when the player enters a tunnel similar to the technique used on the *Scarface* video game[12].

PD allows for extensions to utilize other languages such as Java[17]. This capability enables PD to have access to functions that would be better realized as a procedural piece of code rather than the modular coding style that PD tends to encourage. It is also very easy for coders to create custom objects in C++ for PD using helpful existing frameworks such as flext [11] and extend PD's functionality as desired. PD also communicates quite well with Flash using the flashserver object [15]. Very rapid development of game elements could be done with the combination of using Flash for the visuals and PD for the audio.

PD is a very powerful tool to aid sound designers in exploring new possibilities for game audio on modern game consoles.

References

1. Alonso, M., Geiger, G., & Jorda, S.: An internet browser plug-in for real-time audio synthesis. WEDELMUSIC '04: Proceedings of the Web Delivering of Music, Fourth International Conference on (WEDELMUSIC'04), (2004) 23-26. from http://dx.doi.org.proxy.lib.sfu.ca/10.1109/WEDELMUSIC.2004.4
2. Audiokinetic: WaveWorks interactive sound engine. Montreal, QC, Canada (2006) Retrieved January 30, 2007, from http://audiokinetic.com/
3. Bajakian, C.: The future of game audio production. Paper presented at the O'Reilly Mac OS X Conference 2004, Santa Clara Ballroom (2004) Retrieved January 30, 2007, from http://conferences.oreillynet.com/presentations/macosx04/clint_bajakian.ppt
4. Bridgett, R., & Paul, L. J.: Establishing an aesthetic in next generation sound design. Gamasutra, (June 20, 2006) Retrieved January 30, 2007, from http://www.gamasutra.com/features/20060621/bridgett_01.shtml#
5. CNMAT: Open sound control. USA (2006) Retrieved January 30, 2007, from www.cnmat.berkeley.edu/OpenSoundControl/
6. Cycling '74: Max/MSP and jitter. USA (2007) . Retrieved January 30, 2007, from http://cycling74.com/
7. Danks, M., & Bogart, B.: SPAM: EA is looking for pd people (August 26, 2006) Message posted to http://lists.puredata.info/pipermail/pd-list/2006-08/041486.html
8. Digidesign: Protools. USA (2006)

9. Fristrom, J.: Manager in A strange land: Books are good. Gamasutra, (2003) Retrieved January 2007, from http://www.gamasutra.com/features/20031031/fristrom_01.shtml
10. Gabler, K., Gray, K., Kucic, M., & Shodhan, S: How to prototype a game in under 7 days: Tips and tricks from 4 grad students who made over 50 games in 1 semester. Gamasutra, (January 2007) . Retrieved January 30, 2007
11. Grill, T.: Flext - C++ layer for cross-platform development of Max/MSP and pd externals (2005) from http://grrrr.org/ext/flext/
12. Jackson, B.: From scarface to simlish. [Electronic version]. Mix (Oct 1, 2006) Retrieved January 30, 2007
13. Johnson, S.: The long zoom. [Electronic version]. The New York Times (October 8, 2006) Retrieved January 30, 2007
14. Khronos Group: Collada (2007) Retrieved January 30, 2007, from https://collada.org/
15. Matthes, O.: Flashserver (2006) Retrieved January 30, 2007, from http://www.nullmedium.de/dev/flashserver/
16. Microsoft: XNA game studio express 1.0. USA (2007) Retrieved January 30, 2007, from http://msdn2.microsoft.com/en-us/directx/aa937791.aspx
17. ml: Java external plugin for pure-data (2006) Retrieved January 30, 2007, from http://www.le-son666.com/software/pdj/
18. Native Instruments: Reaktor 5 (2006) Germany
19. Paul, L. J.: Coding vs. Aesthetics. Paper presented at the 2003 GDC Written Proceedings, San Francisco, CA, USA (2003) Retrieved January 30, 2007
20. Paul, L. J.: Audio prototyping with pure data. Gamasutra (May 28, 2007) . Retrieved January 2007
21. Puckette, M. S.: Pure data. San Diego, CA, USA (2007) Retrieved January 30, 2007, from http://crca.ucsd.edu/~msp/software.html
22. Sarlo, J.: Pdvst (2004)
23. Puckette, M. S.: The theory and technique of electronic music (Draft: December 30, 2006 ed.). USA (2006) World Scientific Publishing Co. Pte. Ltd. Retrieved January 2007, from http://www.crca.ucsd.edu/~msp/techniques/latest/book.ps
24. Schiemer, G., & Havryliv, M.: Pocket gamelan: A pure data interface for mobile phones. *NIME '05: Proceedings of the 2005 Conference on New Interfaces for Musical Expression*, Vancouver, Canada (2004) 156-159.
25. Sony Computer Ent. America: *SCREAM*. USA (2005) Retrieved January 30, 2007, from https://www.cmpevents.com/GD05/a.asp?option=C&V=11&SessID=3817
26. Steinberg: *Nuendo*. Germany (2006)