Building Dharmapala: a closed sound environment for exploratory improvisation, made with Pure Data

Marcus Alessi Bittencourt

Universidade Estadual de Maringá - CCH-DMU
Av. Colombo, 5790 - Bloco 8, sala 2
87020-900 - Maringá, Paraná - Brazil
phone: (55)(44) 3261-4092

mabittencourt@uem.br

ABSTRACT

This paper presents a behind the scenes analysis of the construction and process of composition of a musical work of mine for live electronics, Dharmapala, which presents itself as a closed improvisational sound environment, implemented with the computer software Pure Data. We will present the inspiration and concept for the piece, followed by explanations on its internal construction, its implementation in Pure Data, and remarks and instructions on the live performance of the piece.

Keywords

live electronics, computer music, improvisation, Pure Data.

1. INTRODUCTION

In my work as a composer, I have always been concerned with the design of sound and musical objects as well as of instruments of music in a Schaefferian sense [1]. I also have a habit of regarding these musical objects of mine as something like sound characters, which sonorously behave in very specific ways, unfolding in time their inner characteristics. When several of these characters are put together, their own properties and their very interaction serve important structural roles in the compositional logic of my pieces, actually to the point of them being the very source of it. Usually, I tend to work those ideas in the context of very precise scores and sequences of musical fabric. Nonetheless, I sometimes feel the urge of dismissing the constraints of working with a fixed text and go after preparing something more relaxed, more plastic, usually in the context of my own explorations as an improviser. Here I point out that the concept of improvisation does entail a thorough knowledge of models, of basic procedures which serve as guides to the improviser in the process of concocting a composition live, in real time. The musical piece presented here works within this text-free context, but never failing to set a precise field of operations for the musician, with precise rules of conduct and consequences for the player's actions. These features guide and stimulate the performer to interact with his own musical utterance. In this improvisational context, every little thing played by the musician can't be taken back by him and has a musical consequence which escapes his immediate control. All the musician can do is interact with it, learn its aural logic, use it for his own expressive benefit, and hopefully guide it to a meaningful development. The creation of such arena of musical exploration is the inspiration behind this piece.

PdCon09, July 19-26, 2009, São Paulo, SP, Brazil.

2. BACKGROUND OF THE PIECE

2.1 On its Musical Concept

The Dharmapala composition is to be regarded as an exploratory improvisational environment in a way very much indebted to Pauline Oliveros' work with "Deep Listening" [4] and her seminal electronic improvisational works from the 60s like "*I of IV*" and "*Bye Bye Butterfly*". This audio engine pursued a design that aimed at endowing the output sounds with a recognizable unity of mood and pathos, according to the type of sounds present in the input. Led and stimulated by the live musician, the machine is called to output sounds that seem to behave in very specific and inter-related ways, thus creating the opportunity for the development of a sense of form and direction to the piece. To accomplish this, the Dharmapala engine was thought out as a system of interconnected sound chambers resembling a sort of a digestive system, where the incoming substances would peregrinate through several stages, undergoing chemical transformations of all sorts, being distilled into their primary elements or combined into bigger molecules, traveling from one chamber to the next. Of course (and hopefully), I never intended to take this metaphor all the way to its unfortunate logical conclusions regarding the final product. The piece was also thought out as a collection of several movements, not necessarily to be played all in the same event. In each movement, the musician should feed the machine with the sounds of one single instrument or voice, and the piece will magnify the qualities of this instrument, transmuting it into something else, hopefully fierce and impressive, with lots of opportunity for nuances.

2.2 On the Name Dharmapala

Dharmapala, "defender of the righteous path" in sanskrit, denotes a member of a group of wrathful Buddhist deities depicted in the iconography as powerful beings with multiple heads, arms and legs, crowned with skulls and parading fanged teeth with their vicious mouths wide open. They are usually shown girdled with chopped-off heads, amidst columns of fire and trampling over their prostrated defenseless enemies. Not exactly evil, their function is to destroy the passions of the mind and protect the Dharma, the righteous path [3]. There are eight Dharmapalas:

Yama (The God of Death), Mahakala (The Great Black One), Yamantaka (The Conqueror of Death), Vaisravana (the God of Wealth), Hayagriva (The Horse-necked one), Palden Lhamo (The Goddess), Tshangs Pa (White Brahma) and Begtse (The God of War). All are considered wrathful manifestations of Bodhisattvas: enlightened beings that out of compassion refrain from entering Nirvana to undertake the mission of saving others. The composition was named after these wrathful deities due to the finalized machine's ability to build with ease not only very menacing, ear-grinding sonorities, but soft, delicate, meditative-like sounds as well, all this in a very plastic way and, if the musician is apt, with drama and panache.

3. DISSECTION OF DHARMAPALA

3.1 Foreword

I will now dismantle part by part the rather huge labyrinth which became, little by little, the Dharmapala engine. True to the concepts enumerated previously, it is composed of five stages, or rather audio chambers, all connected in a very specific and special manner. As a visual aid to the dissection, I included schematics of the several stages. Throughout them, the expression i-rnd(x,y) means a randomly chosen integer number between x and y, and the expression f-rnd(x,y) means a randomly chosen floating-point number between x and y. A thick line-arrow denotes a single channel of audio, a medium-thick double line denotes a pair of channels in a stereo setup, and a thin line-arrow indicates data or action streams. Some of the algorithms feature a starting action point, which is marked with a star.

3.2 The Moonshiner Stage

The first audio chamber I devised was thought out as a kind of sound distillery, hence the name "Moonshiner". Although it receives audio (in this case, directly from the adc~), it is only used as subject for spectral analysis by an FFT listening unit, and the sound it produces is entirely synthesized by oscillators. The overall sound result of this stage is an ever-changing, rhythmically pulsating web of timbre colors, all derived from the incoming sounds. These colors pile up and linger for a good while, due to a sometimes violent feedback loop, and slowly undergo a process of color replacement by the new incoming timbres. The core of this stage is an automaton which takes snapshots of the pitch spectrum of the current incoming audio and hands over the data to the oscillator banks in the form of a list of the amplitudes and frequencies of the eight strongest partials. It performs this task rhythmically according to a duration scale. Here we use a rather Stockhausenesque scale of durations [9] based on a harmonic-series of a 50 msec fundamental and its seven first harmonics. Upon choosing one duration figure, the automaton has 66% of probability of keeping taking snapshots at that duration figure rate, and 33% of probability of picking another figure in the rhythmic scale for the spectrum sampling task. The synthesis is accomplished by two banks of eight oscillators each (one for every partial in the FFT data), each bank producing one of the channels of the stereophonic field. For the sake of a "shinning", ever-changing aura-like sound effect, each oscillator has its amplitude independently varied through a swelling machine that continuously adjusts its volume during a time fragment chosen randomly between .05 and 1 second to an amplitude value chosen between -100 and 0 dB. The audio produced goes still through a limiter, to boost some soft sounds inside the timbres and square off at -6 dB some of the loud waveforms, giving a rougher distorted sonority [5] [7], then goes through a reverb unity, set to a large reflective room, and finally leaves the Moonshiner stage. Before going to the limiter, the audio is also sent through a delay line of 1 second in a feedback loop returning to the FFT listening unit. This will growl back at the performer if he is not attentive, but that is the actual spirit of the thing. Anyway, the added limiters do take care of the possible distortions, making them more colorful and gritty than painful.

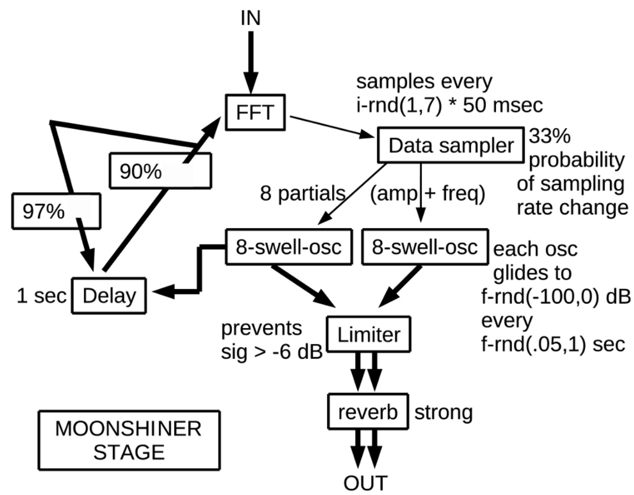

Figure 1. Schematic of the Moonshiner stage

3.3 The Ring Modulator Stage

Ring modulation being a most favorite token in my bag of tricks, I couldn't let it out of Dharmapala. This is also the only stage that allows incoming sound from the adc~ to somewhat pass directly to the dac~, although in a garbled, ring-modulated way. The special feature here is that the frequency of the RM modulator is chosen based on spectral data extracted via FFT from the very incoming sound itself, which will also act as the RM carrier waveform. This, at least in principle, ensures that we will have mostly quasi-harmonic resulting timbres for the ring-modulated sounds [7] and this in a real-time adaptive away, as the incoming sound changes. Another idea I pursued was to have the outgoing sound coated with a nice moving inter-related lilting melodic pattern of timbres, as if the sound spectrum whistled and danced through its own partials. To accomplish this, the FFT data is sampled by a rhythmic automaton similar to the one of the Moonshiner stage, but working with a duration scale based on a fundamental figure of 100 msec (this puts this audio chamber working on a tempo half the speed of the Moonshiner stage one). Instead of sending a list of partials every snapshot, the automaton sends only one of the partial frequencies, randomly chosen among the eight in the data. In order to avoid some unfortunate frequency values and to yet throw in some graininess and a slight increase of inharmonicity to the result, the frequency is validated for use only if it stays between 5 and 5000 Hz and it is transposed two octaves down. Only then the value is used by the ring modulator as frequency for a sinusoidal modulator waveform. The carrier waveform is, as mentioned before, the actual incoming audio from the adc~, and the ring-modulated signal is mixed fifty-fifty with its original unprocessed source. It is then sent to an automated stereo panning and amplitude sweller which glides to a randomly chosen panning value between 0 (left channel) and 1 (right channel) and an amplitude value between -2.5 and 0 dB, through the course of a time fragment between .5 and .75 seconds. Before leaving this audio stage, the signal is limited at 0 dB, a precautionary measure to prevent occasional overflowing, and receives a mild reverb patina.

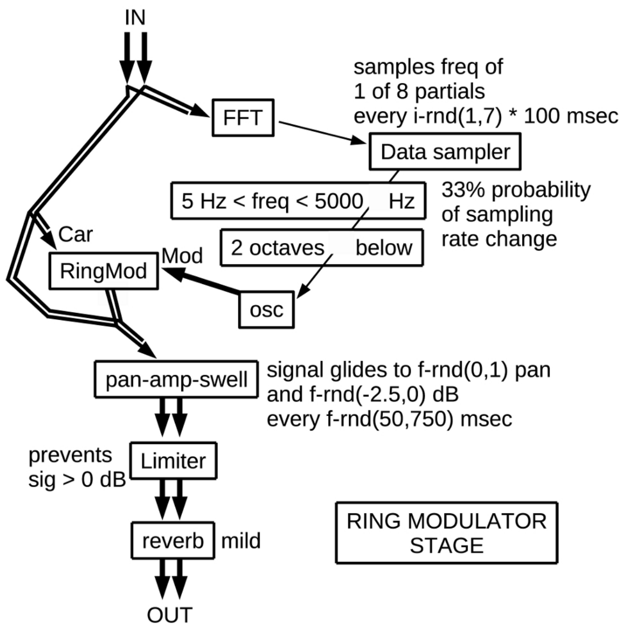

Figure 2. Schematic of the ring modulator stage

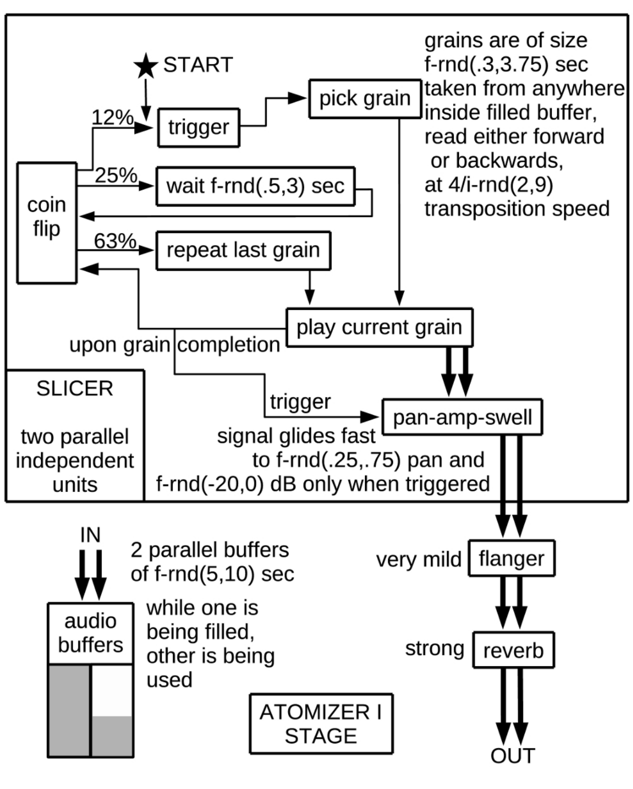

Figure 3. Schematic of the Atomizer I stage

3.4 The Atomizer I Stage

This is the first in a series of sound chambers whose objective is to work with the process of memory recall. The idea here is to work with an instant type of recollection, not literally but in dreamlike nonlinear terms. If we are to be able to do such type of nonlinear editing with audio buffers in the context of a live-electronics performance, care has to be taken not to simultaneously read and write the same buffer or else your desired audio memories could become glitchy or would overwrite themselves and be unable to be recalled independently. I also felt it was not desirable to have deaf moments when sound wasn't getting recorded at all. To implement this idea, we set up a variable-size two-container filling mechanism with two audio buffers of variable sizes, so we would have some variety in how far ago the memories actually came from. These buffers are both initially set to a write-only status, so reading (and therefore editing) is only possible after at least one container has been completely filled. The algorithm that controls this audio storage operation first picks a size for the first buffer to be filled, randomly chosen between 5 and 10 seconds, then opens the audio input for listening and starts filling it (the Atomizer I stage, as we will see later on, is fed with audio coming from the Moonshiner stage). When the first buffer gets full, it is marked read-only, the algorithm sets the second buffer as write-only and proceeds to fill this second buffer just as described before, first picking a size for it, then allowing audio to enter its premises. When the second buffer is filled, the game is reversed: the second buffer is marked read-only, the first buffer is set as write-only and emptied of its contents. The algorithm then resumes the process from the top. This way, we always have a buffer available with audio to read from, which here represents a continuous flux of short-term memory of sounds heard from the Moonshiner stage some 5 to 10 seconds ago. Once such memory flux starts, two parallel and identical sushi-chef like automatons will chop, edit, mix and remix live those sound memory bits and pieces. Initially, each automaton (called "slicer" in the schematics) defines what I call here a grain (the sushi slice, if you must), which is a specific sound object constructed from material taken from the memory flux. To manufacture this grain the automaton first chooses a fragment of .3 to 3.75 seconds taken from anywhere inside the current read-only buffer, then it chooses if it will be played forward or backwards and finally sets for it a very specific playing speed. This transposing speed choice is based on the following principle: if we consider the original pitch level of the grain as being the fourth subharmonic of an undertone series, the possible transposition levels for the grain are the ones corresponding to the pitch levels of the 2nd, 3rd, 4th, 5th, 6th, 7th, 8th and 9th subharmonics of such series. To precise what will happen here, the possible transposition factors will be +12, +4.98, 0, -3.86, -7.02, -9.7, -12 and -14.04 semitones, approximately. This method proposes a magnifying effect on the pitched contents of the incoming memory flux, creating a denser combination of harmonies sharing a natural, just-intonation minor mood (for this concept, see [2] and [6]), used in a rather arpeggiated setting. Upon definition of a grain, the slicer enters a loop mode where it keeps continuously throwing dice and performing, according to the odds, one the following actions: it can immediately play the current grain (63% chance), it can wait silently a time between .5 and 3 seconds (25% chance), or it can define a brand new grain and play it (12% chance). This sound design clearly yields results indebted to the old-school *sillon fermé* technique [8], but with the insertion of variable, asymmetrical silent gaps in between.

Every iteration of a grain will be independently given a stereo panning position and a gain level, ranging between .25 and .75 pan and -20 and 0 dB. Before reaching the exit of this audio stage, the combined sounds from the two parallel slicers undergo a mild flanging effect coloring process and receive a thick layer of reverb.

3.5 The Atomizer II Stage

As one could have guessed by its name, the Atomizer II stage shares lots of features and functionality with its sibling, the Atomizer I, but with some crucial differences. The first one is that the audio buffers here have sizes ranging between 10 and 20 seconds, making the available continuous flux of memory be of a medium-term quality rather than a short-term one. This stage, like its brother, also works with two parallel sushi-chefs, here called "choppers" to distinguish them from their "slicer" counterparts. Instead of a grain, the chopper manufactures what is here called a loop: an ordered and looping string of 1 to 3 grains, each grain consisting of a fragment of size 1.3 to 3.75 seconds taken from anywhere inside the current read-only buffer. This makes the Atomizer II work with quite bigger fragments of sound than the Atomizer I. Each chosen grain is set to play always forward and with a playing speed chosen by a just-intonation minor procedure similar to the one used for the Atomizer I. If there we had results of a rather arpeggiating character, here we have a more scalar end result, working with the 5th, 6th, 7th, 8th, 9th, 11th, 12th and 13th subharmonics of a series where the original pitch level figures as its eighth subharmonic. The just-intonation pitches produced here are more foreign to our equal-tempered scale, with the possible transposition settings being +8.14, +4.98, +2.3, 0, -2.04, -3.86, -5.5, -7.02 and -8.4 semitones, approximately. As one can see, here the pitch range magnification of the grains is considerably smaller than before and in a stepwise fashion. The dice-throwing loop mode of the "chopper" also has different actions and probabilities, namely: it can immediately play the next grain in the loop queue, circling along to the next item in the list (57% chance), it can wait silently a time between .5 and 3 seconds (29% chance), or it can define a brand new loop of grains and play its first grain (14% chance). The resulting mix from the slicer is then sent to an automated stereo panning and amplitude sweller which glides suddenly to a randomly chosen panning value between .25 and .75 and an amplitude value between -20 and 0 dB every time interval of .3 to 2 seconds. Before reaching the exit of this audio stage, the combined sounds from the two parallel choppers go through a delay line of 450 msec with 60% of high-pass (6.5 KHz) filtered feedback, then one of the stereo channels is dephased 100 msec from the other, to create a more spacious stereo image, and the result undergoes some medium size reverberation. Another important difference between the Atomizers is that while number I works with the synthesised blots of sound color output by the Moonshiner stage, number II receives audio directly from the adc~, completely unprocessed. Thus, here is the closest we can get in Dharmapala to hearing the original adc~ input sounds in an unprocessed manner.

3.6 The Swell Orchestra Stage

The function of this quite simple stage is to create, every time interval between 10 to 30 seconds, a single synthesized, hazy, crescendo-diminuendo style sound object event lasting from 5 to

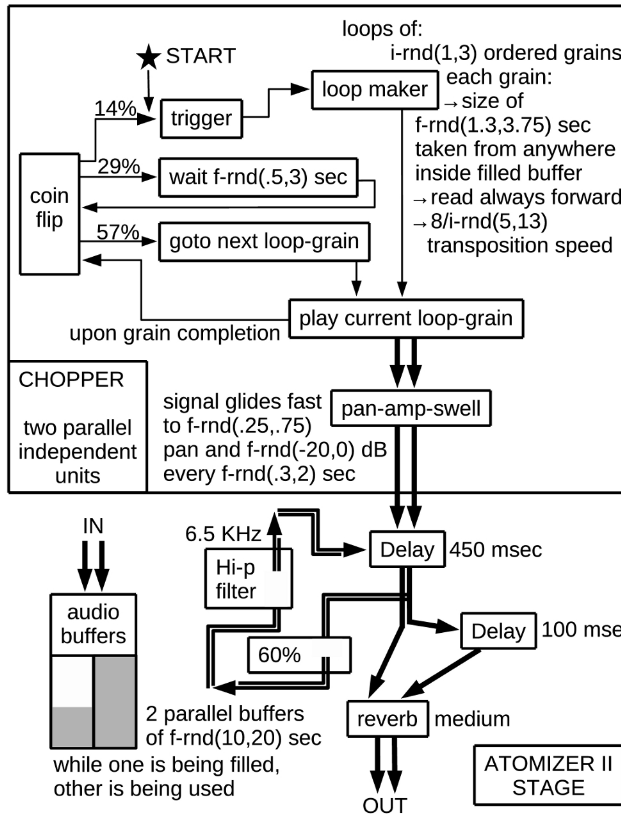

Figure 4. Schematic of the Atomizer II stage

15 seconds. This sound object represents a medium to long-term timbre memory of the collective past of other main Dharmapala stages and will have its pitch spectrum constructed from the data of a single FFT snapshot containing the amplitudes and frequencies of the eight strongest partials, taken somewhere between 10 to 30 seconds ago in the past of the input of this stage. The synthesis is accomplished by two banks of eight oscillators each, all identical in structure to the ones used in the Moonshiner stage. Each bank will produce one of the channels of the stereophonic field, and each oscillator has its amplitude independently varied through a swelling machine that, during the course of .05 to 1 second, adjusts continuously its volume to an amplitude between -100 and 0 dB. The audio produced by the banks of oscillators goes through a limiter at -6 dB, which serves to boost some soft sounds and square off some of the loud ones, and then goes to a potentiometer which is the actual responsible for manufacturing the sound object. Always closed when idle, the potentiometer serves to shape the envelope of the object during the course of its chosen duration to the form of a steady crescendo followed by a steady diminuendo of equal size. The last stop for the output signal is to receive a very thick layer of reverb.

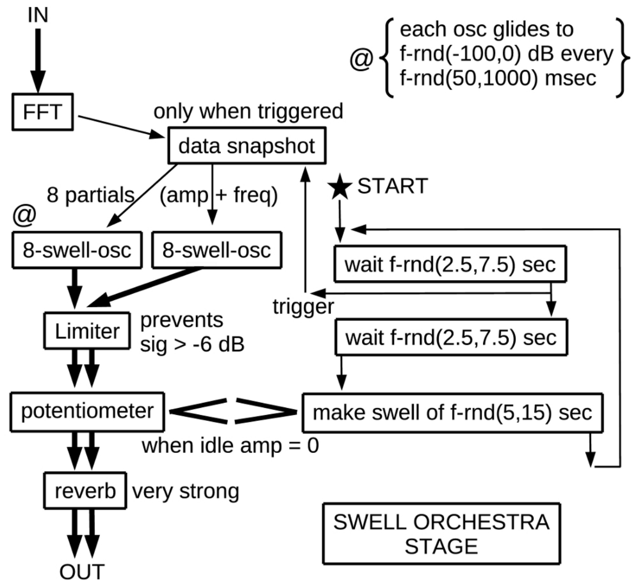

Figure 5. Schematic of the Swell Orchestra stage

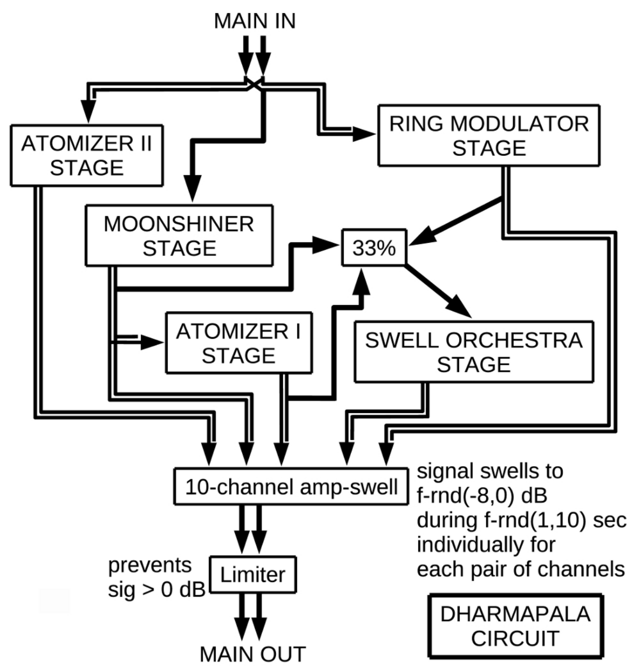

Figure 6. Assemblage of the Dharmapala circuit

3.7 Assemblage of all Stages

The incoming adc~ input is distributed directly only to the Moonshiner, Atomizer II, and Ring Modulator stages. The Atomizer I stage receives its input from the synthesized sounds output by the Moonshiner stage and the Swell Orchestra stage receives its input from the equally combined output from the Moonshiner, Atomizer I and Ring Modulator stages. The stereo outputs of all five stages are routed to a 10-channel automatic amplitude swelling machine that keeps adjusting individually the volume of each incoming stereophonic pair in a continuous manner to an amplitude value chosen between -8 and 0 dB, during the course of time fragments between 1 and 10 seconds. After this automatic variable mixing stage, and right before being fed to the dac~, the signal is yet again hard limited at 0 dB, as a precautionary measure to prevent occasional overflowing.

4. THE PURE DATA IMPLEMENTATION

4.1 General Description

Dharmapala consists of a single pd patch containing literally hundreds of sub-patches and abstractions of varied complexity levels. It is a huge maze whose construction required a good deal of trial-and-error sessions, crafting each audio stage individually, testing it with simple control sounds, correcting it when it plainly didn't work or didn't perform aurally as expected, adapting it, improving it until it performed adequately, safely for live performance and according to the musical concepts intended both collectively for the piece and singly for each stage. The version of Pure Data used to compose Dharmapala was the one distributed by CCRMA for the Fedora 10 Linux, pd-0.39.3-1. The patch makes use of some external libraries, mainly the ZOOexternals, a collection of experimental objects programmed by myself during the last few years, and the LADSPA plugin~ object from the flatspace library. The LADSPA plugins used are Steve Harris' swh plugin flanger (named "Flanger"), and LADSPA's default hpf plugin (named "Simple High Pass Filter"). Regarding the ZOOexternals, it is extensively used throughout the patches. The main objects used are X_limiter~, a simple hard limiter, X_Lbuffer and X_Lbufferplayer~, a pair of objects consisting of an audio buffer for recording directly to the computer RAM and a player that can access fragments of the buffer contents nonlinearly, with a fade-in, fade-out applied, backwards or forward and in any speed transposition, freeverbit_st~, a pd implementation of the known freeverb reverb algorithm, int_between and float_between, objects that pick random numbers from a given range as integers or floating-point numbers, respectively, Zlist, a storage device for pd lists, plus objects to operate these lists (Zlist_size, Zlist_element, Zlist_shuffle, Zlist_append, Zlist_sublist), fork, a data stream forking device after the jMax fork object, and Zgate, a variable multiple outlet routing device. Faithful to a common construction practice of mine, the Dharmapala patch has a colorful main interface that hides all the complexity of its sub-patches, presenting crucial information in a simple way to the performer and providing all the controls and switches necessary for the live musician to operate the whole patch. I have to confess that, for a person interested in studying these patches, their readability is really problematic, requiring a mammoth effort from the reader, mostly due to the non-breaking patch-chords in pd (such patch-chord folding ability is the only Max/MSP feature I really envy). Nonetheless, once assembled, the whole patch amazingly does work as intended (or so I believe at this moment), and it has been safely producing some amusing live music in concerts.

4.2 The Dharmapala Interface

As mentioned earlier, the main colorful pd patch of the composition serves as interface between the performer and the automated audio engine. The interface contains switches to turn the audio engine on or off, a stopwatch to help the musician pace himself, VU meters and sliders to control audio input and output. Some commands can be triggered by plain touching the keys of the computer keyboard (with, of course, no "enter" required), and for that we have a toggle box to switch on or off the keyboard listener. The input has a monitor control, that allows the input to be heard directly in the output, useful only as a pre-concert regulation device. The input can be pre-processed through a fierce limiter, which first hard limits the input signal at -20 dB, then amplifies it at +20 dB. This is very useful, if not mandatory, if we are to deal with very percussive incoming sounds, like a Brazilian berimbau for example, for the transients of the percussive attacks can generate garbage FFT data and produce uninspiring shrieks in some of the sound-synthesizing audio stages of the composition. The main keyboard keys used for fast access control are: "n", to start the audio engine, "z", to signal it to stop, "0", to reset the stopwatch, "=", to start the stopwatch, "-", to stop it. The performer, in emergency situations, can raise or lower the input and output volumes with the keys "1" and"q" for the master volume, and "3" and "e" for the input volume. For safety of operation, the keyboard listener patch waits idle 300 msec after any key has been pressed.

Figure 7. Interface screen for Dharmapala

5. THE PERFORMANCE OF THE PIECE

The musician should elect an instrument to play, out of a list of eight possibilities (I have not yet compiled a complete listing, but the number eight refers ideally to the eight Dharmapalas). As we will see later, my first two concert-proven instrument suggestions are the berimbau and a vintage analog oscillator. Very importantly, one has to figure out a way of sending the sound of the instrument to the computer without feedback from the live electronics. For this, contact microphones come to mind, and this is what I use for the berimbau. To play the piece, basically, after the performer sets his input audio levels and regulates the loudness/softness response of the output (no frying people's ears, please), all he has to do to play the piece is to turn the audio engine on by pressing the "n" key. The performer has to remember that the more he feeds the machine with sounds, the more excited it will get, the more complex its sound output will be. The musician has to listen and react to the sounds produced, conducting the flow of musical activity to interesting places. With the way Dharmapala works, the suggested obvious musical form to be tried is a structure that starts calmly, pianissimo and quasi-niente, evolves through various stages of activity, at the improviser's discretion, guidance and ingenuity, and finally dies out slowly and calmly back to a pianissimo, quasi-niente. The provided stopwatch serves to help the musician keep track of the overall dimensions of the performance. When done, the performer should press the "z" key, which will automatically initiate a gentle and slow fade out (to prevent the musician to inadvertently start new sound activities) and, upon silence, the audio engine will be automatically turned off. That's it.

6. FINAL REMARKS

And so, I arrive at the end of these observations on Dharmapala. The lack of a score for it does make these explanations the only readable form of tracking down for myself and for others what I have intended this composition to be. Regarding my experience performing this piece, its first live concert trials presented two takes on the main concept. In the first movement (Dharmapala I: Yamantaka), the instrument used is the Brazilian berimbau, captured with a contact mike and played in two ways: beaten with its usual stick and stone and played with a violin bow. In the second take (Dharmapala II: Palden Lhamo), we feed the machine with the sounds of a vintage analog oscillator, played live by the musician through the operation of its dials, buttons and switches. The former is a vigorous piece, virile and intense, the later is delicate, subdued, but not without its energetic moments. I consider patcher programs like Pure Data to be the most efficient way nowadays to build a project like this one. With a profound commitment to the open-source community, I elected Pure Data as one of my daily workhorses. I particularly appreciate the way it first opens to us: a blank canvas just waiting to be filled with ideas. It is also extremely reliable and of very safe operation in live contexts. Its handling does require some good patience and ingenuity from us though, but this is usually a most pleasant affair.

7. REFERENCES

[1] Bittencourt, Marcus Alessi. *Acousmatics, Sound Objects and Instruments of Music*. In *Proceedings of the Spark 05 Festival of Electronic Music and Art*. Minneapolis, Minnesota, USA. University of Minnesota Twin Cities, 2005, 46-49.

[2] Costère, Edmond. *Lois et Styles des Harmonies Musicales*. Presses Universitaires de France, Paris, 1954.

[3] Kumar, Nitin. *Wrathful Guardians of Buddhism - Aesthetics and Mythology*. 2001. http://www.exoticindiaart.com/wrathful.htm (last accessed 01-02-2009).

[4] Oliveros, Pauline. *Acoustic and Virtual Space as a Dynamic Element of Music*. Leonardo Music Journal, Vol. 5 (1995), pp. 19-22.

[5] Puckette, Miller Smith. *The Theory and Technique of Electronic Music*. World Scientific Press, Singapore, 2007.

[6] Riemann, Hugo. *Ideas for a Study "On the Imagination of Tone"*. Journal of Music Theory, Vol. 36, No. 1 (Spring, 1992), pp. 81-117.

[7] Roads, Curtis. *The Computer Music Tutorial.* The MIT Press, Cambridge, MA, 1996.

[8] Schaeffer, Pierre. *A La Recherche d'une Musique Concrète*. Seuil, Paris, 1952.

[9] Stockhausen, Karlheinz and Barkin, Elaine. *The Concept of Unity in Electronic Music*. Perspectives of New Music, Vol. 1, No. 1 (Autumn, 1962), pp. 39-48.