Electronic dramaturgy and computer-aided composition in Re Orso

Marco Stroppa, Jean Bresson

Electronic dramaturgy and computer-aided composition in *Re Orso*

Marco Stroppa and Jean Bresson

Re Orso (King Bear) is an opera merging acoustic instruments and electronics.[1] The electronics were realised at IRCAM with the assistance of Carlo Laurenzi. The libretto, written by Catherine Ailloud-Nicolas and Giordano Ferrari, is based on a fable by Arrigo Boito.

Every moment of the opera is exclusively inspired by and tightly related to the prescriptions of the libretto and the intimate structure of the drama: there are no vocal, instrumental or electronic sounds that do not have a deep connection to and a musical justification in the dramaturgy. In addition, an important compositional objective was that the electronic material be endowed with a clear dramaturgic role, in order to be perceived as a character on its own (actually, several characters) with a personality that develops during the action.[2]

Preceded by a short *exordium*, *Re Orso* is divided in two parts of approximately 45' (five scenes) and 30' (three scenes) separated by an *intermezzo storico*. The ensemble leaves the pit at the end of this first part and the singers remain alone with the accompaniment of electronic sounds. Voice and electronics are therefore essential elements of the dramaturgy and of the composition. Both have been written and organised with computer-aided compositional tools. This text explores some of the representative OpenMusic patches developed for this project.

---

[1] Commissioned by Françoise and Jean-Philippe Billarant for IRCAM, the Ensemble Intercontemporain, the Opéra Comique in Paris, La Monnaie in Brussels, and a French *Commande d'Etat*. It was premiered at the Opéra Comique in May 2012.
See `http://www.opera-comique.com/fr/saisons/saison-2011-2012/mai/re-orso`

[2] For this reason the word "electronics" does not appear in the subtitle of the opera, and is replaced by its respective roles: *Re Orso, Musical Legend for 4 singers, 4 actors, ensemble, invisible voices and sounds, spatialisation, and acoustic totem.*

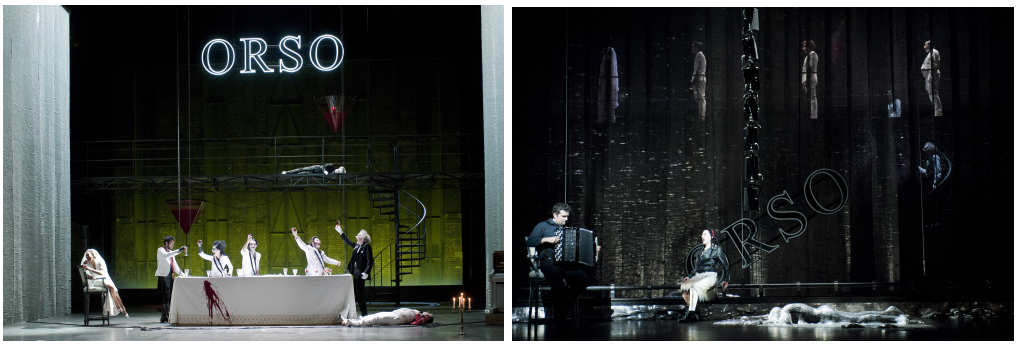

Figure 1. *Re Orso.* Opéra Comique, Paris (May, 2012). Photos: Elisabeth Carecchio

Compositional environment

Re Orso is organised around a "compositional environment" consisting, at the most basic level, of 72 "formant chords"[3] and 3 scale structures,[4] as well as of several rhythmic cells and dynamic profiles. The music deals with this environment according to a panoply of rules that will not be analysed here.

This pre-compositional environment is used to generate the basic musical structures of the piece, which the composer calls *Leitgestalt*. A *Leitgestalt* is defined as a cognitive morphology [11] playing a dramaturgic role; that is, an acoustic and theatrical event that impinges upon our cognition and hence can be recognised in other parts of the piece when it comes back, even if it is transformed. For instance, a downward *glissando* always means a kind of fall (moral, physical, psychological, etc.), large or small depending on the extent and details of its realisation.

Re Orso brings a variety of electronic materials into play:

- Imaginary voices (FOF-based synthesis using the OM-Chant library [3]);
- Imaginary sounds (synthesised using the OMChroma library [1]);
- Imaginary instruments (physical modelling synthesis using Modalys);
- Imaginary ensemble of the *intermezzo storico* (recorded and remixed ensemble);
- Mysterious voices (recorded voices processed with AudioSculpt's cross-synthesis);
- Mechanic doubles of the two *buffo* roles:
  - Hand-controlled puppet (vocal sounds processed with SuperVP-Trax),
  - Computer-controlled, singing Disklavier;
- Real-time processing of the main female role's voice (Chromax spectral delay [5]);
- Overall coordination and concert setup (Max with Antescofo).

[3]Chords algorithmically derived from databases of formant data (see footnote 11) transcribed from sung vowels (5 or 8 formants) approximated in a chromatic space. In total, *Re Orso* uses a data base of 44 5-note chords and 28 7-to-10-note chords.

[4]A "scale structure" is a succession of absolute pitches that encompasses the playing range of the ensemble. These structures are based on some intervallic and symmetric properties.

The voices (4 singers and 4 actors) and instruments are amplified.[5] The sound projection uses an 8-channel frontal setup (3 loudspeakers placed on the stage, 3 loudspeakers hung above the stage, plus one hidden below the stage and one oriented downward from the ceiling) and an *acoustic totem*, a column of eight loudspeakers that appears from below the floor in the middle of the stage at the end of the piece.

OpenMusic has been used both for writing instrumental parts and for the generation of electronic sounds. In particular, the OM-Chant library allowed the connection of formalised compositional processes to vocal sound synthesis using the FOF technique and the Chant synthesiser.

Sound models

The *sound model* is a fundamental concept in *Re Orso* as well as in most previous works by Marco Stroppa. It is, with respect to the composition of sound, something similar to the idea of *Leitgestalt*, or more precisely, to the concept of *Musical Information Organism* developed by the composer in the 1980s for his musical material: a group of sounds "that consists of several components and properties of varying complexity, maintaining certain relationships and giving rise to a particular form" [9]. This concept implies that different sounds corresponding to the same model can be recognised as possessing the same identity and, therefore, can express the same sonic potential [10]. Technically, this concept of sound model is implemented in the OMChroma library as a class (called *cr-model*), which aims at facilitating the generation of families of sounds [4].

Control of additive synthesis

One on the most used *Leitgestalten* in the opera is the "phantom" of a bell. A representative example of it is the series of synthetic bell-stroke sounds that can be heard at the end of the *intermezzo storico*. Dramaturgically, these sounds introduce to the audience the increasing fear of death in the King's mind. The dark sound of the the great tenor bell at Winchester Cathedral was taken as a reference and used as a starting point for the generation process.[6] Through a sequence of strokes, the timbre of the synthetic bell becomes increasingly dark and more ominous.

The OpenMusic patch shown in Figure 2 implements this process. The starting material is the spectral analysis of a bell-stroke recording, imported as an SDIF file at the top left of the patch. This file contains a *chord-sequence* analysis, performed with AudioSculpt software, based on time-markers hand-positioned in the spectrogram in order to parse the different states of the spectral evolution of the sound.

The segmentation and spectral analysis data are used to instantiate the *cr-model* object labelled "model 1" whose contents are shown at the right. This model provides a reservoir of time/frequency structures that can be processed in the time and frequency

[5] The amplification has three main purposes: first, it allows for a better fusion between the acoustic sounds and the electronic materials; second, it gives the stage director total freedom of movement on stage, without risking that the singers are not heard during a *tutti*. Finally, used as an acoustic microscope, it can bring to the audience softly whispered sounds that would otherwise be inaudible.

[6] This is an homage to Jonathan Harvey's *Mortuos plango, vivos voco*, which uses the same bell at the beginning of the piece.

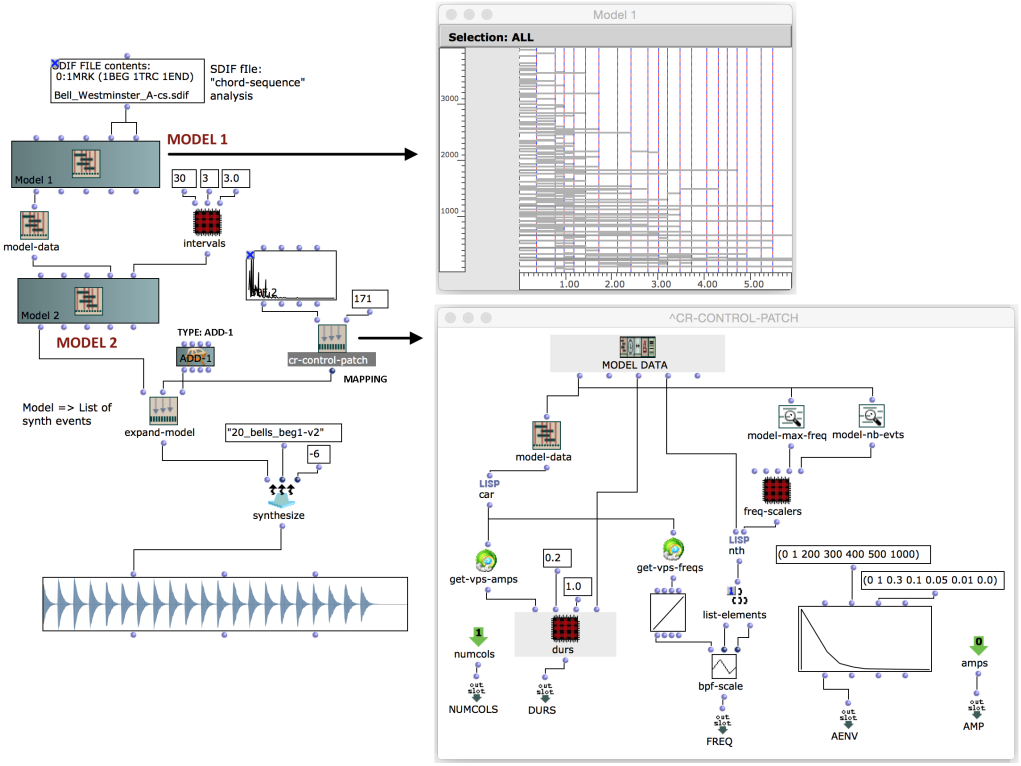

Figure 2. Patch synthesising the bell strokes played at the end of the *intermezzo storico*.

domains (leading to another *cr-model* labelled "MODEL 2") and finally converted into a set of sound synthesis control parameters.

The *expand-model* function performs the conversion of the successive segments from MODEL 2 into a list of instances of a given class of "synthesis event" from the OMCHROMA library (in this case, the class *add-1* is used, which corresponds to a simple additive synthesis). The function therefore returns one *add-1* instance for each segment in the input *cr-model*. Along with the *cr-model*, the second argument of *expand-model* is a patch (*cr-control-patch*), which determines the details of the mapping performed between the *cr-model* segments and the sound synthesis parameters.

The editor window of *cr-control-patch* is also visible in the figure. In this special patch, the synthesis parameters (or "slots" of the class *add-1*) are represented by the *out-slot* arrow boxes at the bottom. The mappings here are of several kinds:

- The number of "components" of each synthesis event (NUMCOLS—in this case with the class *add-1*, the components corresponds to *partials* in the additive synthesis process), as well as their respective amplitudes (AMP), are derived from the two "standard" patch inputs.
- The durations (DURS) and frequencies (FREQ) of the components are determined according to programmed relations with the data contained in the *cr-model* segment (represented by the *model-data* box at the top).

- The amplitude envelopes of the components (AENV) are not dependent on any external data and are specified statically by the BPF visible at the bottom right of the patch.

The *expand-model* function applies this *cr-control-patch* iteratively, updating the *model-data* at each iteration with the values from the current segment, and producing a new instance of *add-1*.[7] The *synthesize* function at the bottom of the figure collects this list of *add-1* instances and runs the actual synthesis process. For each state of the spectral evolution of the original stroke sound (that is, for each segment of the *cr-model*, and each iteration within *expand-model*) a new, increasingly dark bell stroke is synthesised.

Abstract modelling and gestures in the Disklavier part

At the beginning of the 5$^{\text{th}}$ scene (Part 1), the dramaturgy calls for both a Troubadour and his mechanical double (a Yamaha Disklavier—computer-controlled robotic upright piano) to sing two arias. Making a piano sing may look like a utopian challenge: while examples of a speaking piano were realised in the past with special hardware added to the instrument,[8] to our knowledge nobody had yet made this instrument *sing* on stage.

The notion of sound model, previously used to parameterise sound synthesis, was applied here to generate a score for the Disklavier. The classical approach consisting of analysing sounds to derive instrumental scores, developed and thoroughly explored by spectral composers, can be advanced further when combined with symbolic processing in computer-aided composition programs. The process was long, painstaking, and somewhat tedious but yielded a spectacular result on stage.

To proceed, the composer chose to use the popular aria *La Donna è mobile* from Verdi's *Rigoletto*. Several performances were compared until one was found (Alfredo Kraus, 1979) whose spectral analysis was the most interesting for this purpose. As a Disklavier cannot have the same expressive freedom as a real singer, the first task was to "straighten" the metre of the original recording by superposing an audible beat to it as a reference (♩. = 138), and by time-stretching some parts of the audio so that they adequately fit with the downbeats. In total, this extended version contained 36 measures at 3/8 and one at 4/8, whereas the original had 34 measures at 3/8. OpenMusic was then used to generate markers (six per pulse, resulting in a density of 13.8 attacks/sec). The AudioSculpt *chord-sequence* analysis of the transformed sound performed using these markers identified a maximum of 60 partials per time segment (see Figure 3). At this stage, approximately 45000 notes were generated, which was not only too much for the Disklavier, but also still sounded too mechanical.

Empirical testing showed that a density of 10 notes per pulse was the upper limit that a Disklavier could reasonably afford to play without becoming unstable, and eventually, crashing. Given the amount of data contained in the original analysis, a manual data reduction would have been extremely inefficient. Programs were therefore written in Lisp and used in OpenMusic to perform dynamically evolving processing and filtering over the whole sequence and to confer a more human character upon the instrument (see Figure 4).

[7] In functional programming a function like *expand-model*, taking another function as one of its arguments, is called a *higher-order function*.

[8] See for instance Peter Ablinger: `http://ablinger.mur.at/speaking_piano.html`

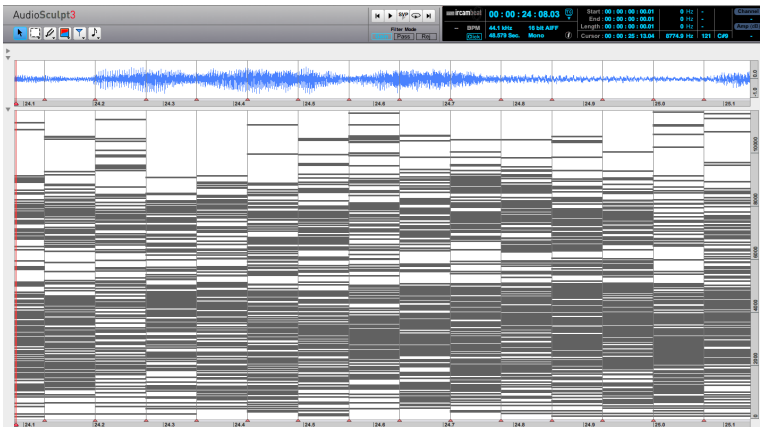

Figure 3. Zoom on 1s of the *chord-sequence* analysis of *La Donna è mobile* in AudioSculpt.

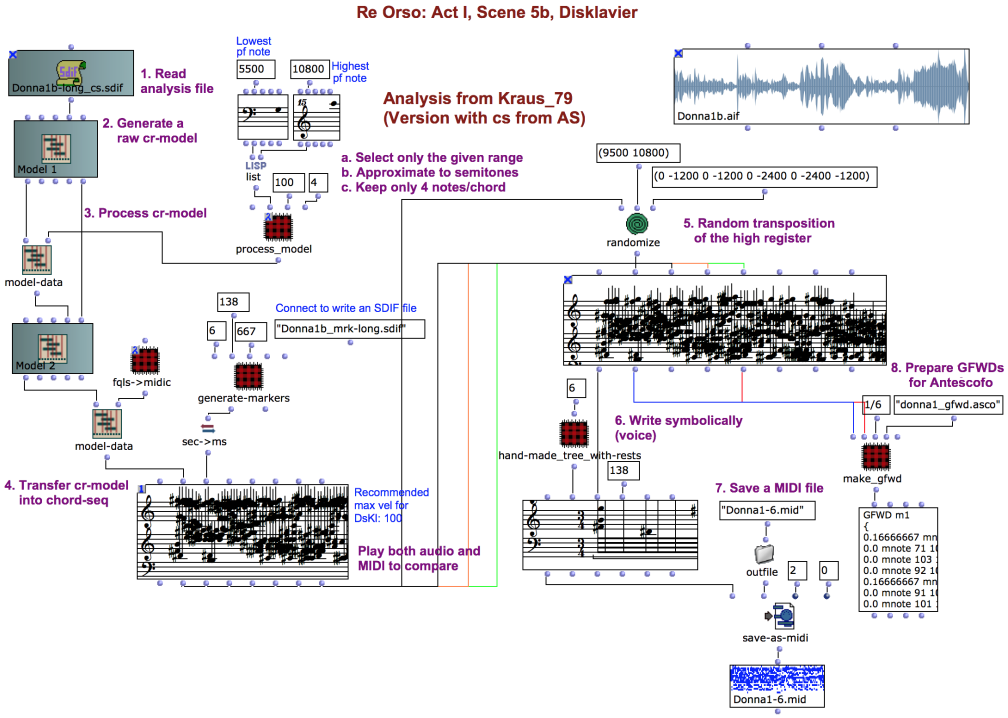

Figure 4. Processing a sound model to generate a score (MIDI file) for the Disklavier.

The principal processing routines implement the following operations:

- Approximate the data to semitones and eliminate repeated notes.
- Cut out pitches above the range of the piano keyboard.
- Eliminate pitches below G3 so as not to take into account possible analysis errors.
- Underline the difference between *f* and *pp* using higher registers for the *f* passages.
- Intensify the impression of a *crescendo* by gradually modifying the number of notes in the piano's upper octave and a half (i.e. during a *crescendo*, the higher notes in this range appear progressively).[9]
- Add some randomness (i.e. select keys around the original ones) in the higher pitch range. This allows deviations from the original B-major tonality, but also avoids too many successive repetitions of the same keys—which can be dangerous for the Disklavier.

The final MIDI editing was done with Digital Performer (see Figure 5).

Pedal automations are added (*una corda* or mute), especially in the *pp* passages. This not only changes the sound, but also creates a theatrical effect, as the piano on stage has no lid and viewers can see the entire mechanics shifting closer to the strings.

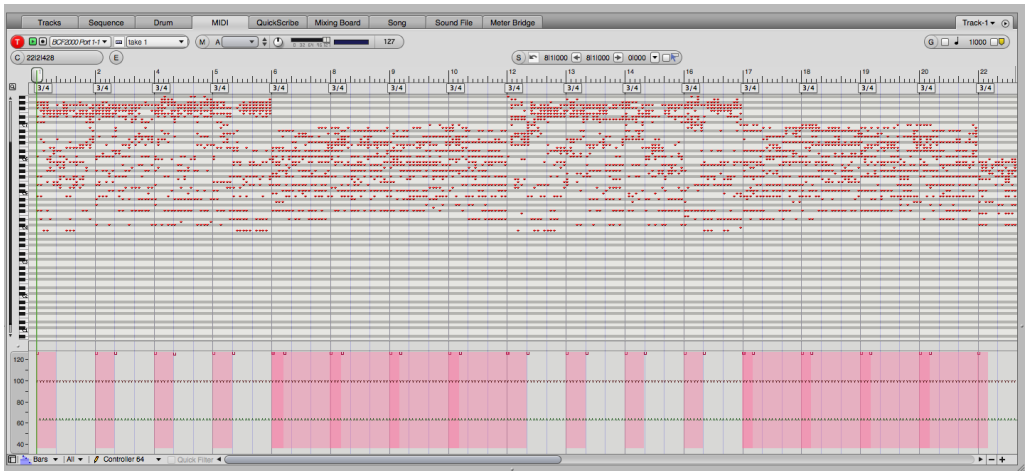

Figure 5. MIDI file of the beginning of the first song of the Disklavier in Digital Performer. Notice the spectral difference between *f* passages (e.g. in the first four 3/4 bars, with more notes in the high register) and *pp* passages (e.g. bars 5 to 8, with reduced pitch range), as well as the pedal controllers: short sustain-pedal strokes (represented by the dark rectangles) in the *f* passages, followed in the *pp* passages by a combination of *una corda* (wider rectangles) and shorter sustain-pedal strokes.

[9] It is known that the auditory perception of a high sound dynamic is accompanied not only by a higher amplitude, but especially by a wider spectrum. In addition, due to the semitonal approximation, the higher part of the spectrum tends to sound like a chromatic cluster; so filtering out some notes in the soft passages proves to be helpful musically.

Imaginary voices and the control of Chant synthesis

Computer-generated singing voices are a major component of the dramaturgic material of *Re Orso*. Their implementation in OpenMusic was probably the greatest challenge and most innovative research carried out for the opera. They have been realised using the Chant synthesiser [8] controlled by the library OM-Chant.

The composer had long been attracted by the musical expressivity and dramatic potential of the early examples of sung voice synthesis that had been developed at IRCAM in the 1980s using Chant and its control environment Formes [7] (like Jean-Baptiste Barrière's *Chréode*, or the voice of God in Harrison Birtwistle's *The Mask of Orpheus*, realised by his assistant Barry Anderson). Today most of this technological framework has disappeared with the evolution of computer environments. This project allowed us partially to reproduce in OpenMusic some of the temporal and spectral processes that were devised and used at that time.

The Chant synthesiser

The Chant program was initially developed for generating realistic singing voices, although other original sounds based on this voice simulation model could also be synthesised with it. In this program, sounds are produced by a number of parallel FOF generators[10] and filter modules. A FOF generator outputs a periodic train of finely enveloped sine-wave grains, producing the effect of a vocal *formant* in the spectral domain.[11] The FOF parameters (frequency, amplitude, attack, release time, etc.) determine and control the frequency, amplitude, and shape of the formant, while the period of grain generation determines the fundamental frequency of the output sound.

During the synthesis process, a Chant "patch" (specific configuration of different available units or "modules"—FOF generators, filters, etc.) runs continuously and its parameters are modified by external controllers. In the latest version of the synthesiser, these controls are specified via files encoded in SDIF format, where the values and evolution of the parameters are set and stored as time-tagged *frames*. The parameters' changes and the state of the synthesiser are not necessarily set synchronously at every point in time: Chant performs an "off-line" rendering and systematically interpolates between user-specified values of the parameters at a given, global control rate. Between the specified values, linear interpolations produce smooth continuous transitions.

This is an original approach to sound synthesis, as the control "primitive" is not a single event (what we might call a "note") like in most existing systems. In Chant the succession of states, smoothly connected to each other, generates monophonic sounds allowing for subtle expressivity and the control of *legato* details that are needed for realistic singing-voice synthesis (a continuous paradigm that can be related to the notion of "phrase").

[10] FOF = *Fonction d'Onde Formantique*, or Formant Wave Function; see [6].

[11] A *formant* is an amplitude modulation of the spectrum at a specific frequency and with a specific shape and bandwidth, characterising the observation of voiced sounds.

The OM-Chant library

OM-Chant provides tools to format control structures and create SDIF files adapted to the control of Chant, as well as utilities to facilitate setting Chant parameters (database of formant values, implementation of predefined control rules, etc.). As we mentioned previously, the control can be arbitrarily distributed in time, either very sparsely (in which case the synthesiser will interpolate between the specified values), or precisely sampled, describing fine evolutions of parameters via processes of arbitrary complexity.

OM-Chant inherits from Chant "continuous" control paradigm, but also provides discrete timed structures (called *events*). Following the model of the OMChroma library, a number of synthesis *event* classes are defined, which correspond to the different available modules and controls in a Chant synthesis patch: FOF banks (class *ch-fof*), fundamental frequency (class *ch-f0*), filter banks (class *ch-flt*), etc. An instance of a Chant class is an array or a matrix of values that determines the evolution of the corresponding module parameters over a given time interval. Continuous evolutions can be specified for all or part of the synthesis parameters during this time interval: auxiliary tools are provided to generate such evolutions (e.g. the *vibrato* that may occur within the time intervals of the events), or to control transitions between successive or overlapping events [2].

This consideration of timed events together with the description of their morphologies and articulations in a phrase combine abstract musical constructs and "continuous" specifications, drawing an intermediate path where continuous control can be associated with powerful and expressive time specifications.

Synthesis of vocal sounds in *Re Orso*

One demand of the libretto was the invention of imaginary voices: sounds that can be recognised as vocal, but that no human voice could ever produce. Several families of imaginary voices were created, among them:

- Phonemes comprising the succession vowel/consonant/vowel;
- *Messa di voce* sounds (*crescendo* followed by a *diminuendo*);
- Ethereal voices, containing formants wandering around in the spectrum;
- Humorous voices in the *coloratura* register.

Vocal sounds generated with OM-Chant appear in various passages of *Re Orso*. The patch shown in Figure 6, for instance, generates a sequence of sounds corresponding to the first time the King hears the Worm's voice, in the second scene of the opera. Two main objects (or *events*) are instantiated in this patch: *ch-fof* and *ch-f0*. The combination of these two objects determines the set of parameters required for the control of a bank of FOF generators (formant values and fundamental frequency).

On the left, the *ch-fof* values (formant frequencies, amplitude, bandwidths, etc.) are determined according to a chosen vowel (e.g. soprano "o", alto "e", etc.) thanks to the *vowel-formants* utility. A set of "rules" is then applied, such as *autobend* (which shifts the first and second formants' central frequencies as a function of the fundamental frequency), *comp-formants* (which adds a resonance in the lower region of the spectrum), or *autoamp* (which automatically adjusts the relative amplitudes of the formants). Other control parameters used are directly derived from the patch inputs, such as *<win>*,

<wdur>, *<wout>* (which determine the shape of the envelope of the FOF grains), or *<bw-scale>* (an adjustable formant bandwidth scaler).

At the right of the figure the *ch-f0* object controls the fundamental frequency over the duration of the sound synthesis process. The base pitch is here modulated (using *param-process*) by both *vibrato* (controlled by frequency and amplitude envelopes) and *jitter* (aleatoric perturbation, also controlled by frequency and amplitude values).

The *synthesize* function at the bottom of the patch collects a list of CHANT synthesis events (in this case, one *ch-fof* and one *ch-f0*), formats an SDIF control file, and performs an external call to the CHANT synthesiser in order to produce the sound file.

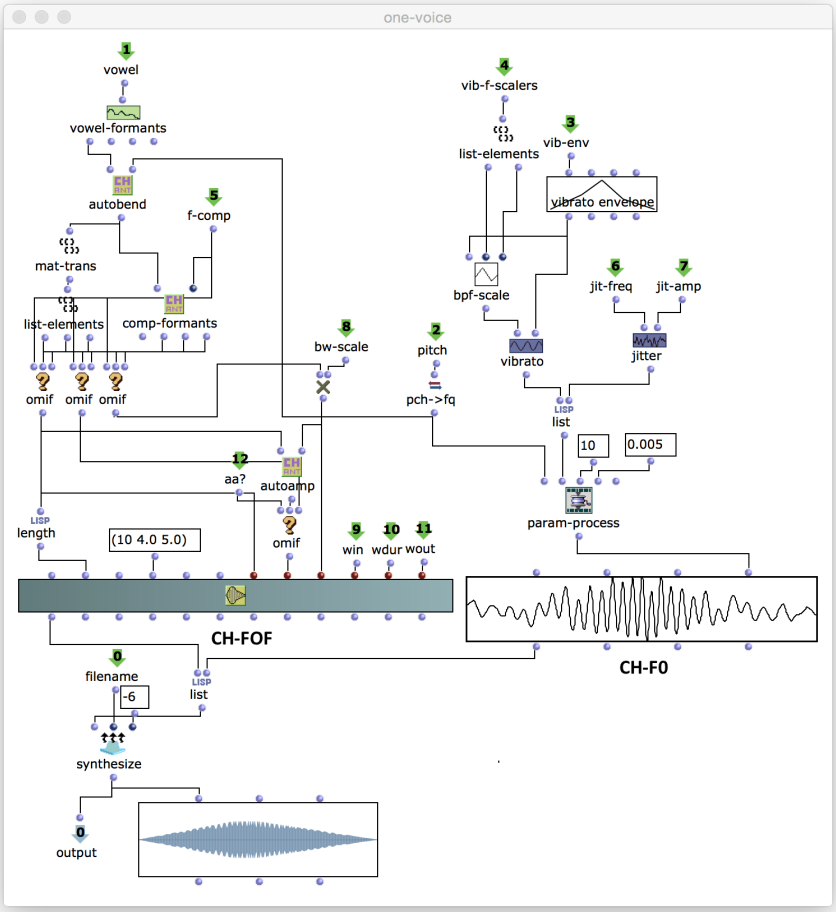

Figure 6. Synthesis of a *messa di voce* with OM-CHANT. Imaginary voices (Part I, Scene 2).

This patch is actually quite generic, and leaves many open parameters (input arrow boxes in Figure 6) that are set in Figure 7 in order to produce a series of different sound files. This series of sounds implements a progressive evolution from "realistic" sounds (formants and fundamental frequency in the range of human voices, natural *vibrato* curves, etc.) to more synthetic/imaginary ones, all produced by the same program.

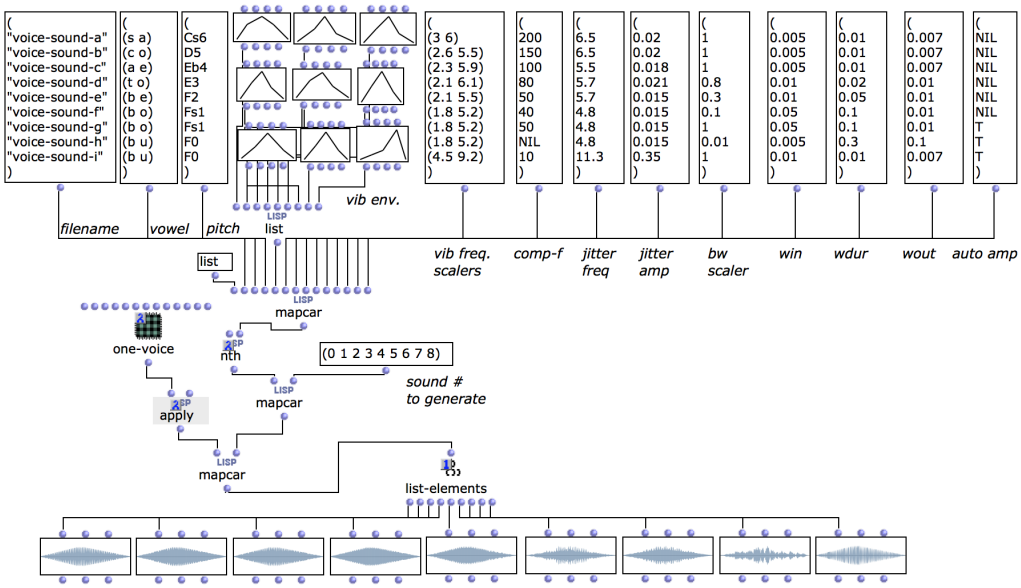

Figure 7. Iterative call of the patch from Figure 6 with variable set of control parameters.

The last sounds and set of parameters in Figure 7 are of particular interest. Very low fundamental frequencies, on the order of a few Hertz, are perceived as trains of impulses rather than pitches, while the short attacks and growing resonance times of the FOF grains produce the effect of percussive, almost bell-like sounds.

In FOF synthesis the control of the formants' bandwidths (and their shape in general) is important for the perceived result. At the limiting case, an infinitely narrow bandwidth (0Hz) is a simple *partial* (sinusoidal signal), and the FOF synthesis can then produce spectra resembling additive-synthesis sounds. We will get back to these interesting properties in our last example.

An example of "phrase" generation: From a cell phone to *Queen of the Night*

At the very beginning of the opera, the ring of a mobile phone is heard through the loudspeakers followed by a message asking the audience to turn their mobile phones off, as is often the case before a performance. However, this time the message is pronounced live by one of the singers (the Worm) in the language of the country where the piece is performed. This indicates that the opera has already started, albeit unconventionally. Just after the announcement the cell phone rings again and gradually turns into an increasingly eccentric voice (crazy melodic contours, fast tempo, extreme range, and exaggerated expressive inflections), until it hints at some passages of the *Queen of the Night* synthesis, a famous simulation realised by Yves Potard at IRCAM in the early eighties. It is a humorous start for the piece; let's examine it more closely.

The first step was to find the right ring. The one selected was a rapid alternation of a major third (D6/B♭5), which had several advantages:

- Sonically simple, yet rhythmically lively.

- It contains both the pitches D (*Leitgestalt* of the King, D=*Re*=King) and B♭ (which is in the pitch-class range of the *Leitgestalt* of the Worm, A4-C♯5). However, both pitches are located one octave higher, that is, in the vocal range of the King's coerced wife (Oliba, high soprano). A lot of theatrical meanings for a simple ring!
- The range and vocal speed were compatible with those of the *Queen of the Night* sound synthesis process.

The patch generating one of the sounds played in this passage is presented in Figure 8. The OMLOOP named *notes-to-chant* contains a patch similar to the one in Figure 6. Instead of synthesising a sound at each iteration, it generates timed CHANT events (*ch-fof* and *ch-f0*) corresponding to the short tones of the cell phone (at the beginning) or to slightly longer sung vowels with an expressive *vibrato* (at the end). The collected sequence of events is synthesised as a whole phrase (outside the OMLOOP). Automatic interpolations are therefore performed between the successive CHANT parameter changes, which produce smooth progressions between the distinct sound features, and implement the gradual embodiment of the initial ring into a human voice. While the short tones sound quite synthetic and mechanical at first, as soon as a vocal-like *vibrato* is added they suddenly acquire more organic and natural characteristics.

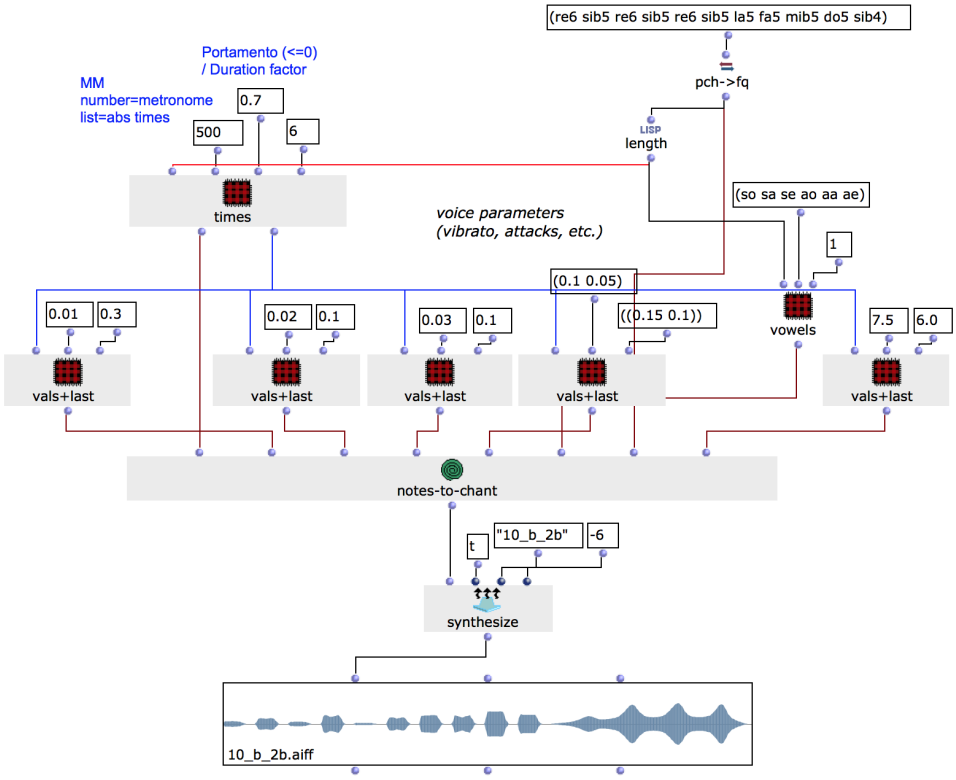

Figure 8. Generating a sung sequence (or "phrase") from a list of CHANT events: cell-phone/queen-of-the-night at the beginning of *Re Orso*.

Integration of sound models and continuous voice transitions: "Death of the King's Voice"

Re Orso dies at last. But the composer wanted to extend the death of this character to encompass the death of his voice as well. How can a voice (as a sound) die? The passage "Death of the King's Voice" summarises a number of central notions from the compositional strategies described in this text and in the opera in general. It is realised through a single FOF synthesis process lasting one and a half minutes. Figure 9 shows the main patch to generate this sequence.[12]

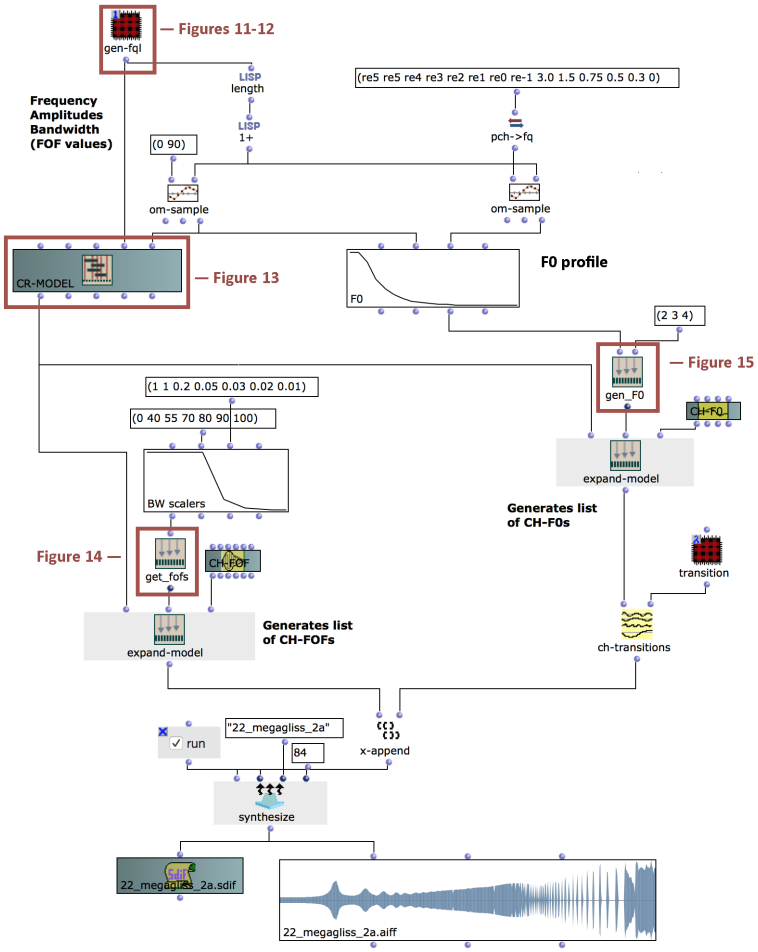

Figure 9. Main OpenMusic patch generating the "Death of the King's Voice". The contents of the framed boxes are detailed in Figures 11–15.

[12]In this and the following figures the patch has been simplified for the sake of clarity. The presented values are also not the exact ones used for the sounds played during the opera.

The resulting sonogram (Figure 10) reveals a number of interesting details of this process. The fundamental frequency of the FOF generator starts at a high D (D5, 585 Hz) and plunges into sub-audio values (0.3 Hz, that is, one stroke each 3.3 s). This fundamental frequency is easy to track (or hear) at the beginning, and then disappears in the low register. The "plunge" was structurally divided into octaves (D5-D4, D4-D3, etc.). Within some of these octaves (D4-D3/tenor register, and D2-D1/deep bass register) a *vibrato* was applied, conferring a strong vocal connotation to the *glissando* in these selected parts of the sound.[13] At the same time five vocal formants, that are clearly visible at the beginning, seem to collapse into independent strokes from approximately 40" after the beginning of the process. The bandwidths of the formants progressively decrease, from approximately 80-120 Hz down to very small values (less that 1 Hz at the end), and make the formants increasingly more resonant. The strokes (or duration of the grains) get longer and longer, from 7 ms to several seconds, in order to let the resonance be heard. At the end the spectrum looks much like an additive synthesis sound. The spectral characteristics (frequencies, amplitudes) of this final part are derived from the bell-sound model described earlier.[14]

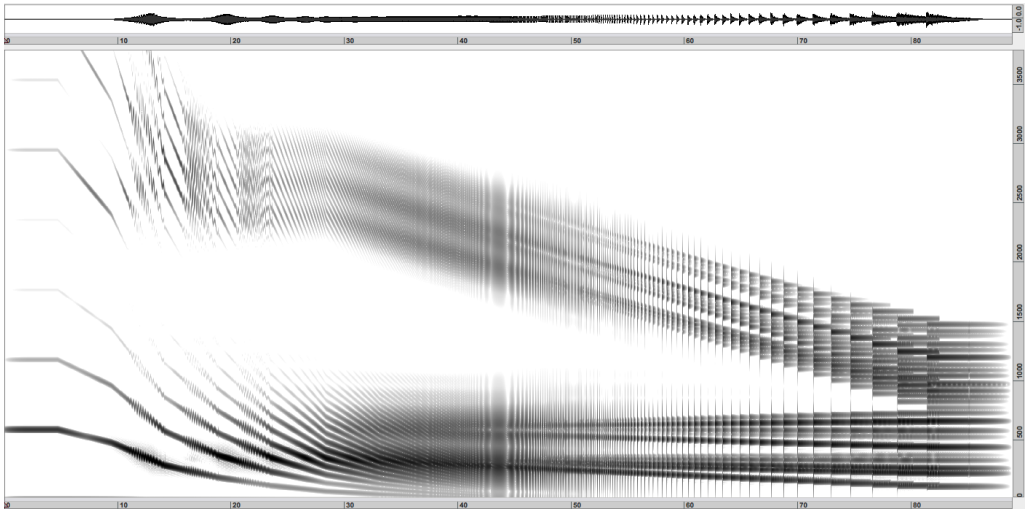

Figure 10. Sonogram of the "Death of the King's Voice" (1'30").

The final bell sound is made up of 25 partials which "appear" in the middle of the synthesis process out of the five formants of the initial vocal sound. 25 FOF generators therefore actually run throughout the entire process, organised in five groups of five formants at the beginning (each group of five is centred on the same frequency, but four of them start with a null amplitude), which progressively spread out to the 25 different frequency/amplitude values of the target sound spectrum as their bandwidths decrease.

[13] However, as classical singers do not often use such long *glissandi*, and normally do not add *vibrato* while performing a *glissando*, this example already suggests a kind of imaginary voice even for a more "vocal" moment.

[14] One might also perceive an indirect reference to the famous passage in Karlheinz Stockhausen's *Kontakte* where a high pitch (E5) starts a downward *glissando* ending with slow, isolated, reverberated impulses tuned to E3.

The sub-patch at the upper-left of Figure 9 (*gen-fql*) produces the sequence of formant data. This part of the process is detailed in Figures 11 and 12.

The sequence starts with formant values selected from a vowel database (Phase 1). During this initial phase the synthesis process renders smooth transitions between the vowel "u" of a soprano (labelled "su"), an alto ("au"), a tenor ("tu"), and, finally, a bass ("bu"). The function *duplicate-formants* does exactly what it says: it duplicates the formant data so that the total number of formants equals the target number of additive partials required at the end of the process.

The sequence ends with the values of the bell-sound spectrum (Phase 3). These values are generated from spectral analysis data in the *west_bell* sub-patch visible in Figure 12. A *cr-model* object is created from a partial tracking analysis and a sequence of hand-positioned markers (two separate SDIF files exported from AudioSculpt). Only one of the segments is selected as a source for frequency and pitch material, and bandwidths are computed from the frequencies using the *autobw* function of the OM-Chant library.

In between (Figure 11, Phase 2) the *interpolation* function generates 12 intermediate states of the FOF generators. Figure 13 shows the contents of the resulting *cr-model* object, considered the "knowledge base" (or "skeleton") of the synthesis process.

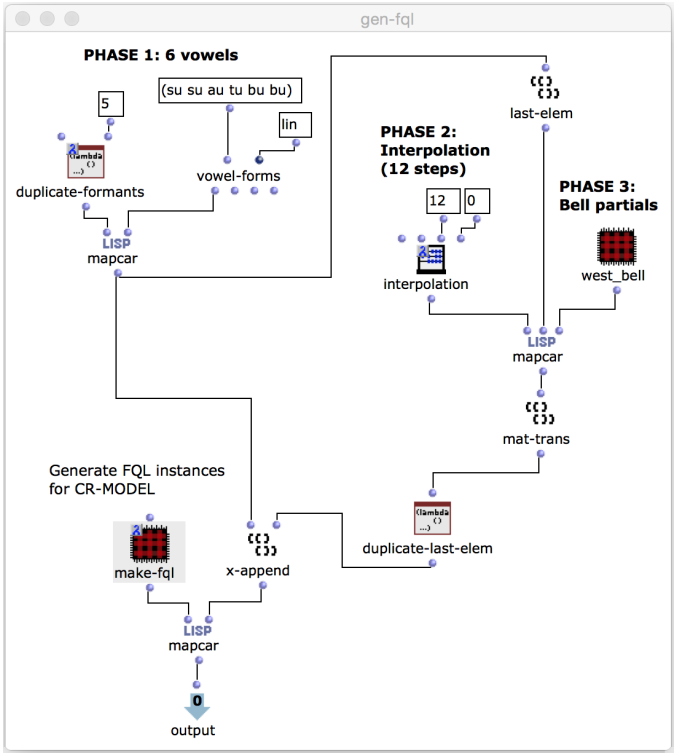

Figure 11. Generation of the sequence of formant data for the "Death of the King's Voice" synthesis process. Left: formant values extracted from a vowel database. Right: formant values derived from the spectral analysis of the Winchester Cathedral bell recording (see Figure 12). Middle: interpolation of the formant values.

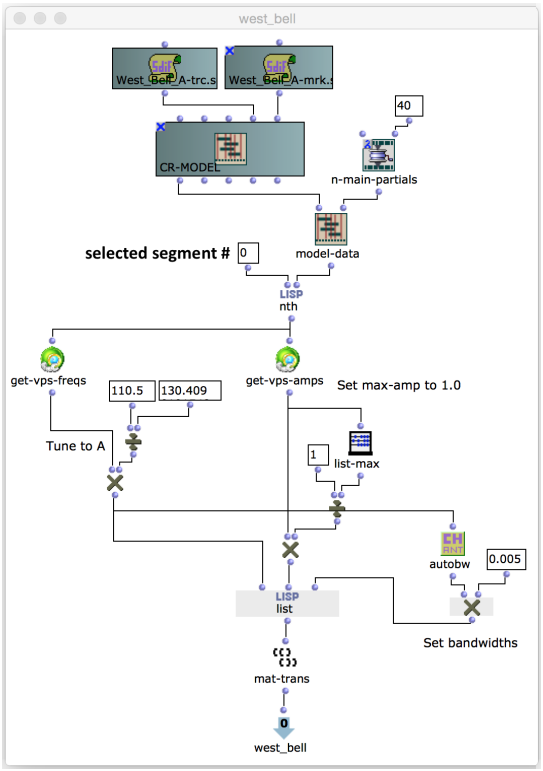

Figure 12. Extracting formant values from the analysis of the bell recording. Note that the original sound has a spectrum similar to C minor, but for the opera, a spectrum similar to A minor was needed, hence the multiplication of the frequency values ("Tune to A").

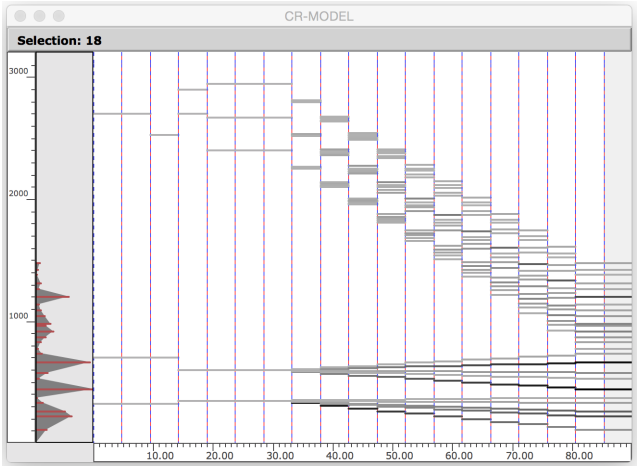

Figure 13. Contents of the *cr-model* object from Figure 9.

Thanks to the combinations of higher-order operators and the library of specialised tools available, this whole process, lasting almost 1'30", is actually implemented as a single "note" in the synthesiser. Once it is triggered, this note develops automatically, thanks to its inner logic, the knowledge base, and the programmed rules contained in the patch.

The *expand-model* box in Figure 9 generates the actual OM-CHANT events driving the sound synthesis process, starting from the *cr-model* data. As with *cr-control-patch* in Figure 2, this time the *get_fofs* sub-patch (see Figure 14) determines a mapping between the contents of the *cr-model* segments and the different parameters of the generated synthesis events (in this case, the slots of the *ch-fof* class: FREQ, AMP, BW, WDUR, WIN, WOUT, etc.)

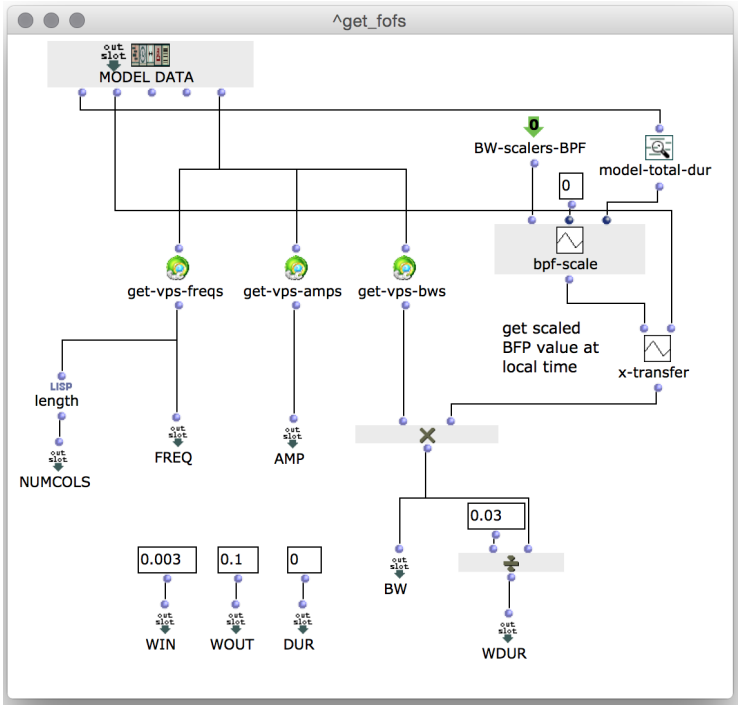

Figure 14. Control patch mapping the *cr-model* data to the parameters of the *ch-fof* synthesis events in Figure 9.

The "BW scalers" curve defined in the main patch is used to scale the bandwidth values of the formants during this mapping process. When the sound is clearly vocal (at the beginning) this factor is 1.0 (that is, the initial values are used), then it rapidly decreases from 0.02 to very small values (until 0.01, which is $1/100^{\text{th}}$ of the original bandwidth) for the last part of the process (remember that narrow formants get closer to additive *partials*). Using a multiplication factor rather than absolute values has the advantage that the wider formants will still have, proportionally, a larger bandwidth and therefore decrease more rapidly, which is exactly what happens with the resonances of a bell sound. As mentioned previously, the duration of the FOF grains (WDUR) also increases during the process.

The most difficult task in the design of this synthesis process was to find musically and perceptually convincing values for the successive intermediate states of the changing parameters. The interesting moments (for example, when the sound begins perceptually to become a bell) are very sensitive and usually the optimal values for these parameters are located within small, cross-dependent intervals. We had to proceed through intensive trial-and-error to find them.

The linear interpolation of the fundamental frequency, for instance, did not work well in the logarithmic space of pitches (the *glissando* is too slow at the beginning, and quickly moves down to "unmusical" values at the end). A linear pitch interpolation yields better results, but then the process tends to sound quite mechanical. Using linear frequency interpolation on portions of the *glissando* within each octave (see "F0 profile" on Figure 9) seemed to give the more interesting musical results.

The *ch-f0* events controlling the evolution of the fundamental frequency of the synthesis process are generated on the right of Figure 9. Here *expand-model* and the *gen_F0* mapping patch only use the time information from the *cr-model* and take data from the fundamental frequency curve. The details of *gen_F0* are visible in Figure 15. This patch mostly consists of a choice of whether *vibrato* is applied to the current segment—see the selection list (2 3 4) in Figure 9. If the current segment falls within the selection, the frequency value is turned into a linear ramp to the next frequency in the list, to which *vibrato* is applied using *param-process*.

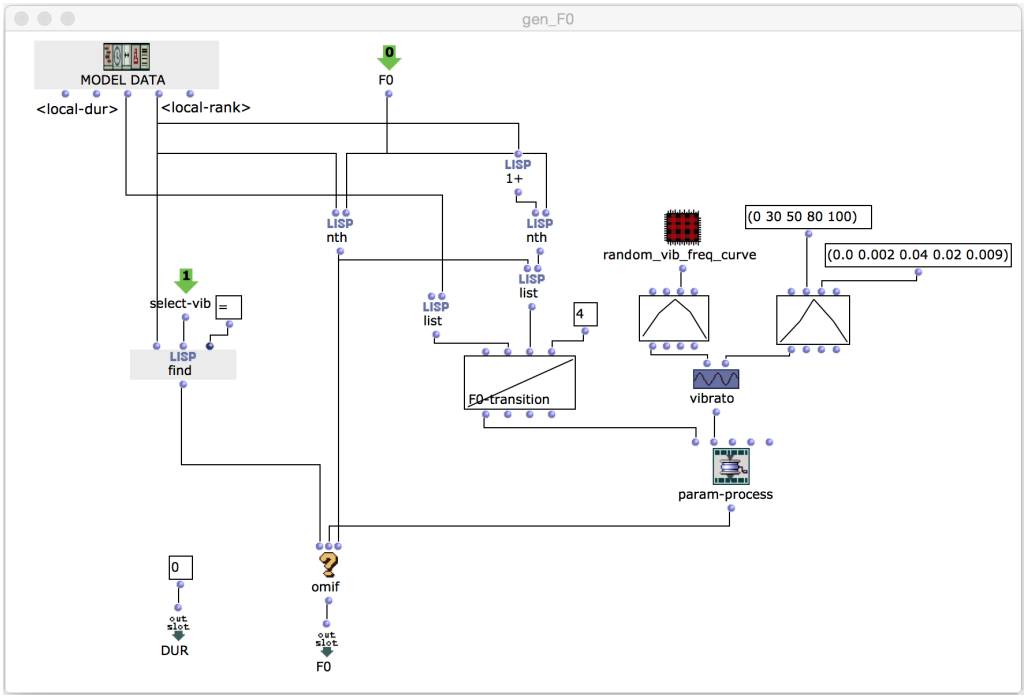

Figure 15. Mapping between values of the *cr-model* of Figure 9, the frequency curve, and the *ch-f0* slots. This patch applies (or not) the *vibrato* function depending on whether the current segment is among the selected indices allowed to pass from *<select-vib>*.

Finally, the *ch-transition* function visible in Figure 9, associated with the *transition* patch, controls the smooth evolution of the whole process, ensuring transitions between the "continuous" *vibrato* curves and static values of the fundamental frequency. The *transition* patch performs a number of tests on the successive pairs of *ch-f0* events generated in the previous part of the process (are the two events continuous/BPF objects? do they overlap? etc.) and makes decisions about the data to be produced as an output to the synthesis process (see [2] for more details on this process).

Figure 16. *Re Orso*: death of the king. Photo: Elisabeth Carecchio

Conclusions

Re Orso allowed us to explore several exciting æsthetic and scientific concepts. During the performances we realised that the electronics could, indeed, play the role of a dramatic character, certainly invisible to the audience, but likely to tell a "story" and to evolve emotionally as the opera unfolds.

The CHANT and OMCHROMA paradigms, both present in the compositional process, proved to be complementary: while OMCHROMA mainly deals with sound structure at the (discrete) level of the notes, OM-CHANT was naturally suited to address issues of phrasing, which are crucial for the synthesis of sung voice and for conferring to the synthetic sounds a high degree of expressivity. With these tools, we were able to produce sounds of high acoustic quality within a reasonable computation time and, especially, to give to the whole sonic realm the unity for which the composer was searching.

We are aware that we only skimmed over certain aspects of the synthesis system, and that much more experience is needed fully to exploit its full potential. The constant guidance of an æsthetic perspective (that is, synthesising a certain sound family because of the needs of the libretto) helped us to focus on the most salient musical directions. This was a crucial aspect of the experience.

References

[1] Carlos Agon, Jean Bresson, Marco Stroppa. "OMChroma: Compositional Control of Sound Synthesis". *Computer Music Journal*, 35(2), 2010.

[2] Jean Bresson, Raphaël Foulon, Marco Stroppa. "Reduction as a Transition Controller for Sound Synthesis Events". In *FARM—Workshop on Functional Art, Music, Modeling and Design, ICFP'13*. Boston, 2013.

[3] Jean Bresson, Marco Stroppa. "The Control of the CHANT Synthesizer in OpenMusic: Modelling Continuous Aspects in Sound Synthesis". In *Proceedings of the International Computer Music Conference*. Huddersfield, 2011.

[4] Jean Bresson, Marco Stroppa, Carlos Agon. "Generation and Representation of Data and Events for the Control of Sound Synthesis". In *Proceedings of the Sound and Music Computing Conference*. Lefkada, 2007.

[5] Arshia Cont, Carlo Laurenzi, Marco Stroppa. "Chromax, the other side of the spectral delay between signal processing and composition". In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. Maynooth, 2013.

[6] Xavier Rodet. "Time-domain Formant-wave Function Synthesis". *Computer Music Journal*, 8(3), 1984.

[7] Xavier Rodet, Pierre Cointe. "FORMES: Composition and Scheduling of Processes". *Computer Music Journal*, 8(3), 1984.

[8] Xavier Rodet, Yves Potard, Jean-Baptiste Barrière. "The CHANT Project: From the Synthesis of the Singing Voice to Synthesis in General". *Computer Music Journal*, 8(3), 1984.

[9] Marco Stroppa. "Musical Information Organisms: An approach to composition". *Contemporary Music Review*, 4(1), 1989.

[10] Marco Stroppa. "High-Level Musical Control Paradigms for Digital Signal Processing". In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. Verona, 2000.

[11] Marco Stroppa. "Auf der Suche nach formalen Polyphonien. Zwischen Musiktheorie und Neurowissenschaft". *Musik & Ästhetik*, Heft 01(1), 2012.

Acknowledgements: The authors would like to thank Thibaut Carpentier, Arshia Cont, Nicholas Ellis, José Fernandez, Raphaël Foulon, Gilbert Nouno and Xavier Rodet from IRCAM, Marlon Schumacher from CIRMMT/McGill University, and Jean-Baptiste Barrière for their invaluable help and advice.