SoDA: A Sound Design Accelerator for the automatic generation of soundscapes from an ontologically annotated sound library

Andrea Valle
CIRMA/StudiUm - Università di Torino
andrea.valle@unito.it

Paolo Armao
Freelance Sound Designer - Zero dB
paolo.armao@gmail.com

Matteo Casu
CELI srl
casu@celi.it

Marinos Koustomichalis
CIRMA/StudiUm - Università di Torino
marinos.koutsomichalis@unito.it

ABSTRACT

In this paper we describe the SoDA project that aims at automatically generating soundscapes from a database of annotated sound files. The rationale of the project lies in the ubiquitous requirement for sound designer to produce backgrounds that include specific multi-layered sound materials. SoDA provides an ontologically-annotated database that allows the sound designer to describe the desired soundscape by keywords: by referring to the database, the system then delivers a resulting audiofile.

1. INTRODUCTION

Sound design is now a term encompassing various domains and applications, from non-interactive audiovisual products to real-time scenarios including virtual reality, multimedia installations, soundscape generation for urban and architectural design. Still, the core of the practice is related to the audiovisual context, where the term originated and that acts as a reference in many aspects for new developments. In the audiovisual production pipeline, sound designers have to carry out various tasks, but a rough distinction can be made in relation to sync sounds/background sounds. Sync sounds have to be edited in relation to specific events defined by the story and/or in relation to specific visual cues (they are event-bound, so to say). Background sounds –typically called Ambience (or Atmosphere) in film post- production– are not event-bound, rather they provide a general sense of space/time/mood that is crucial (even if in many occasions unnoticed by audience) to the semiotic behaviour of the audiovisual text. While foreground sounds are indeed strictly related to action, the production of background sounds requires a different working attitude, focusing on a large, undefined time scale, and thus their organisation shares many aspects with soundscape studies, and consequently with soundscape-related practices. In this sense, ambience creation is the link between audiovisual production and other domains of sound design. Because of this connection, in this paper we describe a project that tackles various issues related to the creation of ambiences in a non-real time context (that is, mostly in relation to audiovisual production), but that can be easily (both on the theoretical and technical side) extended as a general framework for soundscape generation.

2. AMBIENCE AND SOUND DESIGN PRACTICES

Assuming film production as a reference, it is worth briefly discussing sound design practices in order to highlight emerging issues to be addressed. Although approaches may differ depending on professional cultures, budgets and tools used for content creation, it is also true that a combination of practices defines a standard in this field [1]. Ambiences require lot of work to be carried out, typically involving the use of many sound files that are composited by a well-established layering technique (REF). Figure 1 shows

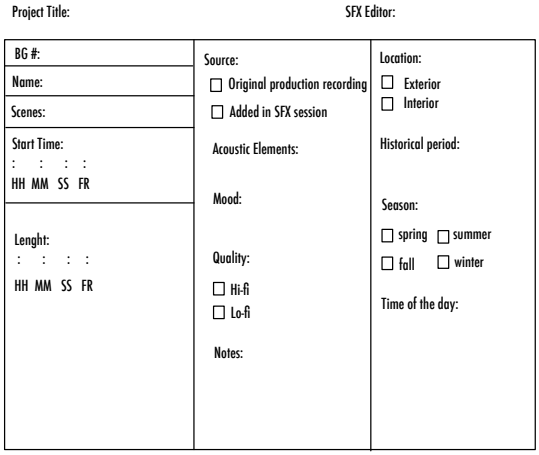

Figure 1. Spotting log for creating ambiances

an example of a tool that sound designers use in support of ambience design development for feature film sessions. This model is a customisation of the SFX Spotting Log originally proposed by Robin Beauchamp [2] used at ZerodB studios. Once all the features that define an ambience are found, the sound designer starts recording or looking acoustical elements that compose the aural scene in his/her

personal library. The ambience is typically built from discrete elements, that is, audio samples stocked in collections ("libraries") organised in various ways. Sound post-production studios develop sound libraries by purchasing commercial libraries and by employing sound editors to record and create private libraries or to create original sounds through synthesis. With regard to audio libraries, the access to sound material typically requires the input of keywords in the search engine, usually referring to a certain desired figurative feature (e.g. fan, cornfield-daytime environment, etc.). Examples of widely diffused libraries on the market are Hollywood Edge [1], SoundIdeas [2], Blastwave FX [3]. Also, online sites for consultation and purchase of single audio samples can be taken into account, such as SoundDogs.com [4], SoundSnap.com [5] and the free community Freesound.org [6]. The classification criteria governing the organisation of the keys are typically heterogeneous and based on practice rather than on explicit formal definitions. The lack of a clear communication protocol between the library producer and the sound designer results in a time-consuming research in the library (thus decreasing efficiency) that does not always produce the desired results (thus decreasing effectiveness). In order to solve this issue, support tools are commercially available, that allow the sound designer to organise audio materials from different libraries in a custom library, and to define his/her own keywords. An example is Soundminer [7]. An alternative method to classification criteria exploits metadata associated with sound files, as defined by the RIFF (i.e. AIFF and WAVE) and MP3 file format (the last not to be into account, given the scant use in the professional field). Metadata can be used as search keys for files, requiring the sound designer, however, to provide a classification of individual samples. Dedicated softwares, like Metadigger by Sound Ideas [8], make it possible to search for samples through the use of metadata. In essence, the work of organizing sound materials falls largely on the sound designer. In addition, the customization of the library increases the productivity of the designer owner, but does not allow to transfer knowledge, since the criteria for classification chosen by the individual sound designer are potentially even more heterogeneous than those of departure. As dedicated human resources are needed to organise sound libraries, this activity rarely happens in everyday activities of medium/small sound post-production studios. Hence, the habit of creating a new library for each project, potentially requires a relevant amount of resources both in terms of time and storage space. In some sense, custom created libraries might be just the first step toward the same accessibility issue emerging with commercial libraries, as it is just a matter of time and work to have them increasing to such a dimension that they cannot be managed anymore by the sound designer. Thus, on one side commercial libraries requires a relevant amount of time to be explored; on the other side libraries assemble from custom recordings, while fulfilling the needs of the very moment, in order to be reused in future projects require an analogous amount of time to be annotated with semantic metadata.

To sum up, the work of sound designer relies heavily on sound libraries (be they commercial or custom-made), but their usage is far from being optimal, as the retrieval process is often unsatisfactory in terms of efficiency and effectiveness, and this sub-optimality propagates through the whole production pipeline.

[1] http://www.hollywoodedge.com/
[2] http://www.sound-ideas.com
[3] http://www.blastwavefx.com/index.html
[4] http://www.sounddogs.com/
[5] http://www.soundsnap.com/
[6] http://www.freesound.org/
[7] http://www.soundminer.com/
[8] http://www.sound-ideas.com/metadigger.html

3. SOFTWARE SOLUTIONS

In sound design, libraries of sound samples stands on the data side. They have a software counterpart in Digital Audio Workstations (DAW, a typical example being Avid Pro Tools), environments that allow to load and edit sounds in relation to a fixed timeline and that can be used as host environments for third party DSP unities, the so-called plugins. While DAWs allow the automation of control by recording and playing back control gestures, they are typically controlled by hand. That is, in the case of ambiences that we are considering, the sound design specifies by means of a GUI which sounds have to be inserted, at which time, and how they are processed. Each ambience results from the layering of multiple sounds, typically from few to dozens. This is a time-consuming task, where layers have to be processed and spatialised in a time-varying way, in order to provide liveness and verisimilitude. As these variations are not related to specific events, they are typically managed in a procedural way, and thus they could be automatised. Solutions from three other domains address the issue of automated compositing of background sounds (event-unbound, so to say).

First of all, sound design for gaming industry has to face the issues of interaction that requires –in relation to background sound– to specify some rules to generate audio as it is not possible to know in advance e.g. duration. The terms "procedural audio" is widely used into gaming industry starting from Farnell's seminal book *Designing Sound* [3]. From a theoretical perspective, procedural audio indicates the use of various techniques for audio synthesis and processing used in the domain of computer music, firmly established and already available, to generate sounds in real time for gaming environment. Such an approach is now gaining a high momentum in sound design for gaming and is also entering in the classic film scenario. In case of textural sounds, procedural audio aims at defining an acoustic behavior that can drive the synthesis, be the latter based on algorithms or on playback/manipulation of atomic sound sample. As an example, in the case of the rain, rather than using audio samples of non-specifiable duration, it can be much more efficient and effective to collect short samples of dropping sounds and to re-generate the overall texture of rain. In game industry procedural audio is actually available through specialised extensions for middlewares, software environments that bridge the software development

level with the hardware level. In input, middlewares [9] provide the developer –in this context: the sound designer– a set of tools and interfaces to work on sound. In output, they generate opportune software elements (e.g. libraries, referring to audio data) to be integrated into targeted applications (e.g. more general game development environments) and platforms (from OS, to mobiles, to consoles), typically over various multichannel configurations (e.g. stereo, 5.1, 7.1 etc).

In the field of softwares for audio-visual sound design, some solutions are intended as alternative tools for the composition of background sounds and effects. Among the most common: MetaSynth [10] and Soundbuilder [11], a list to which special plug-ins for DAW can be added. Metasynth provides the user with a graphical control that allows her/him to literally draw the sound, an approach that facilitates a quick design of the sound material. Soundbuilder is proposed as a tool for creating soundscapes. In contrast to the method based on layering imposed by DAW, Soundbuilder approaches the composition considering the physical space of the scene. In both cases, their peculiarity is to go out from the constraints imposed by the DAW multitrack paradigm (layering of single samples) proposing different approaches. In any case, the materials obtained from these are usually managed within a DAW.

The automatic generation of soundscapes is the focus of three projects. The European project Listen [4] coordinated by the Fraunhofer Institut für MedienKommunikation is focused on the generation and control of interactive soundscapes, but it is specifically targeted to an innovative, media-oriented experimentation on augmented reality. Its main goal is to create a new medium: the immersive audio-augmented environment (IAAE). Listen does not include an explicit modelling of the soundscape and does not provide production tools for sound design. Tapestrea [5] is intended to create "environmental audio" in real-time, but does not define any explicit relationship between sound and space, nor it is possible to interact with the user. GeoGraphy [6] is designed for the real-time simulation of existing soundscapes, starting from a database containing sound materials and other information. It can work interactively and in real-time, it includes the modelling of a virtual listener, but the organisation of audio materials is based on specific data structures ("graphs"), potentially very complex to handle. The underlying model of GeoGraphy is designed for a specific purpose and it does not easily merge into the sound design workflow. In the direction of a simplification of the configuration required to the user, Schirosa [7] discusses an extension for GeoGraphy that allows the generation of realistic soundscapes with a higher level methodology (e.g. by specifying semantic constraints that drive the automatic generation of graphs).

Finally, the use of the computer as a support tool in creative processes, that is through the delegation of high-level competences and not just in terms of low-level processing, is at the center of computer-assisted composition. Software specially designed for this purpose, all based on the Lisp language and with graphical interface, are OpenMusic [8], PWGL [9], Common Music [10]. Their features include the ability to manipulate symbolic representations of the musical fact, together with tools for audio signal analysis. These solutions are indeed eminently musical, thus they do not address the problems of sound design. Still, the idea of constraint-based composition can indeed be relevant for bringing an intelligent support to the creative production sound design.

[9] Two main solutions are available on the market: Wwise by Audiokinetic and FMOD by Firelight Technologies.

[10] http://www.uisoftware.com/MetaSynth/index.php

[11] http://www.soundbuilder.it/

4. AN OVERVIEW OF SODA

In the following sections we introduce the SoDA architecture and its implementation. Figure 2 (left) provides a general formalisation of sound design practice as we have discussed it. First, the sound designer retrieves sound files form an already existing archive (be it created from scratch or commercially available). Then, the resulting sound files have to be organised following a certain schema (e.g. layering). Finally the tracks have to be processed and mixed so that the final audio is available. Taking into account

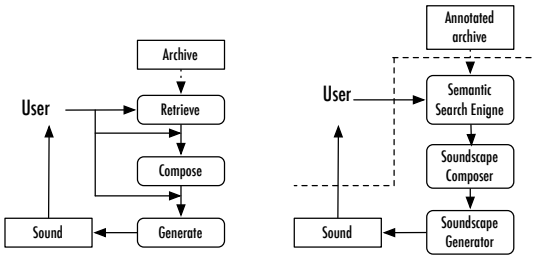

Figure 2. General formalisation of sound design for ambiances (left) and automatisation in SoDA(right).

the previously discussed issues, SoDA aims at providing – with respect to the creation of ambiances– a twofold computational "acceleration" (hence its name) to sound design practice: on the selection of relevant sound elements to be composited and on their organisation. The situation is shown in Figure 2(right), where the slashed line indicates the automated aspects. On one side, in SoDA sound files are annotated by human experts (but not exclusively) with tags preserving relevant information, such as..., and stored into an archive. Then, SoDA features a Semantic Search Engine that allows to retrieve sound files from an input query by the user. On the other side, SoDA features an automated Soundscape Composer that allows a rapid, tuneable, creation of soundscapes of unspecified duration [12]. The Soundscape Composer is passed the results from the Semantic Search Engine and is provided with algorithms for semantically-informed, automated layering. Finally, these data, which are placed at the organisational level, are passed to the Soundscape Generator which is responsible for the final audio rendering.

[12] Hence on we will refer to background sounds or ambiences more generally as soundscapes.

In the following we will first discuss the semantic engine for retrieving sound files, and then two components that implement the strictly audio related parts. Finally we will describe the actual implementation.

5. ANNOTATION AND SEMANTIC SEARCH

In order to enhance semantically the retrieval of sound files, we have defined an annotation schema starting from the way in which metadata are annotated in state-of-the-art libraries. Among the libraries used by sound designers we have taken into consideration Sound Ideas Series (6000, 7000 and 10000), World Series of Sound, Reniassance SFX. The search tools for audio documents taken into account are instead SoundMiner [13] , Library Monkey [14] , Basehead [15] , Audiofinder [16] , Apple iTunes (that, even if not intended for professional use, is used by small studios). An example is the metadata tagset from Renaissance SFX, that includes: identifier, title, length, type of ambience, position (stationary or echo), movement (direction of the sound), description, category (with respect to an internal classification). Regarding how documents are usually tagged in commercial sound libraries two observations can be made:

- firstly, the "description" field often seems to include information that could better be served in dedicated fields — one example is information about the location represented in the sound file
- secondly, libraries rarely follow standard and controlled vocabularies: each library use a specific structure for the fields of its documents and for their values — this is a problem in terms of interoperability between libraries as well as in terms of ease of use for a user (in our case, a sound designer). One example is given by the various formats for dates, or by the different ways to tag the genre

On the administrative and technical side, the Audio Engineering Society (AES) has published two standards [17] : AES57-2011 ("Audio object structures for preservation and restoration") and AES60-2011 ("Core audio metadata"). There are also less specific (but freely available) standards, such as the Dublin Core, in its variants ("application profiles" [18]). Other, more domain-specific, examples, are EBUcore (AES60) and MPEG-7 (including the audio section ISO/IEC 15938-4:2002). With the advent of semantic technologies for the web (linked data[11], semantic web[12]) and RDF-based vocabularies, semantic interoperability has become one of the *desiderata* of data models. In this regard, a tendency which showed up is the confluence of different domain-specific vocabularies in more general ontologies used for data integration. In 2012 the WWW consortium (W3C) mapped some of the most used schemas for media objects in the Ontology for Media Resources [19] , recommending it for the annotation of digital media on the web. The W3C specification is not bound to a serialisation in a particular language, so it can be used as a general schema for the SoDa project.

Within the SoDA project, the Semantic Search Engine heavily relies on classical techniques borrowed from Information Retrieval (IR)[13], whose main task is finding relevant "documents" on the basis of the user's information needs, expressed to the system by a query. The query can be structured or non-structured (i.e., a textual query). The process of feeding the search system with documents has to be preceded by the task of annotating documents (files) with metadata, which are then used by the system to provide features useful for the user, such as faceted search (i.e., the capability of filtering documents by selecting orthogonal features) and query expansion (a technique in which the engine recognizes relevant concepts in the user query and trigger an "expanded query" on the documents). The intelligence showed by the overall behaviour of the system is due to several factors. Among them, the two most relevant are first a a good annotation of the documents; secondly, the disposal of a knowledge base that helps the system in recognising concepts within the metadata and in the user's query (so that the search is *mediated* by the knowledge base).

Obviously, the process of annotation imposes trade-offs between scalability and accuracy: a structured manual annotation, which uses a tagset compliant to the knowledge base used by the system, will likely lead to better IR results, but it will hardly scale to situations in which new documents are created and must be annotated by hand. A good compromise is hence to conceive the document as a set of different fields, some of which can be partially structured (e.g. contain free text, which is easy for a human to manage) and others can be pre-compiled by means of an automated process. Therefore, choosing the right tagset for the annotation is quite a crucial task: the tagset should cover the project needs but also be kept simple, in order to be usable. Moreover, it is a good choice to keep it interoperable with existing standards. In SoDA we conceptually divide the metadata into three sets:

- administrative metadata: provenance information, copyright, date of creation, etc. ;
- technical metadata: file format, length, number of channels, and spectral information;
- content metadata: data about the content described by the document;

Administrative metadata are typically created together with the creation of the audio file. In the SoDa project, scalability on technical metadata is guaranteed by a custom utility developed in SuperCollider (see later) that processes the library in batch mode and pre-compiles technical metadata on each document. Content metadata are compiled by a human, but –to gain scalability– permit unstructured text in some of the fields.

In Figure 3 we give the set of tags used in SoDA by means of a fake but realistic example. Our tagset takes into account both standard practices (see Figure 1) and the needs

[13] http://store.soundminer.com
[14] http://www.monkey-tools.com/products/library-monkey/
[15] http://www.baseheadinc.com
[16] http://www.icedaudio.com
[17] http://www.aes.org/publications/standards/
[18] http://dublincore.org/documents/profile-guidelines/
[19] http://www.w3.org/TR/mediaont-10/

explicitly expressed by the sound designers involved in the project. Among these are the number of channels, the location, and information about the context: season, time of the day, meteo. We also tried to stay as compliant as possible to the *Ontology for Media Resources* as for the names and types of the fields. Among content data, the tag `typeOfSoundObject` is related to a phenomenological appreciation [14], and allow to define Atmospheres (long sounds, with no begin/end), Sound Objects (atomic sounds), Sequences (composite sounds that nevertheless show a unique identity). This tag is relevant for soundscape composition (see later, an analogous classification is proposed in [6]). We observe that some of the content data

```
<doc>
      <!– management metadata –>
    <field name="identifier">1234</field>
    <field name="creator">Machiavelli Music</field>
    <field name="collectionName">My Great Collection</field>
    <field name="copyright">2013 Machiavelli International Musical Images - All rights reserved</field>
    <field name="recordDate">2013-11-12</field>
    <field name="releaseDate">2013-11-15</field>
    <field name="description">dogs barking loud in the streets of turin</field>
    <field name="title">dogs barking</field>
      <field name="url">http://fakedomain.com/docs/1234</field>
      <!– technical metadata –>
    <field name="bitDepth">24</field>
    <field name="channels">6</field>
    <field name="duration">00:00:12</field>
    <field name="samplingRate">192000</field>
    <field name="typeOfShot">closeup</field>          <!– closeup, mid-shot, long-shot –>
      <!– technical metadata –>
    <field name="centroid">520</field>                <!– in hertz –>
    <field name="complexity">0.8</field>
    <field name="dissonance">0.5</field>
    <field name="loudness">3.2</field>                <!– in sones –>
    <field name="onsets">3.2</field>                  <!– array of seconds –>
    <field name="onsets">2.5</field>
    <field name="onsets">1.9</field>
    <field name="pitch">100</field>                   <!– hertz –>
    <field name="sharpness">0.3</field>
    <field name="slope">12</field>
    <field name="spread">1</field>
    <field name="weightedSpectralMaximum">100</field>
      <!– content metadata –>
    <field name="createdIn">torino</field>            <!– place of the recording –>
    <field name="depictsFictionalLocation"></field>   <!– place depicted by the recording –>
    <field name="season"></field>                     <!– spring, summer, autumn or winter –>
    <field name="timeOfTheDay"></field>               <!– morning, noon, evening, night –>
    <field name="content">dogs</field>                <!– content associated to the file (e.g. the source of the sound) –>
    <field name="periodStartYear"></field>
    <field name="periodEndYear"></field>
    <field name="typeOfSoundObject">atoms</field>          <!– sound object, sequence, atmos–>
</doc>
```

Figure 3. An example of SoDA document.

cannot be properly represented by using a flat tagset, but needs some structure: one example is given by the locations, which constitutes a taxonomy (New York is a place in the United States, which are part of North America). Physical objects, present together with the locations in the text descriptions, generally have a similar structure: e.g., birds are animals. The SoDA Semantic Search Engine uses as knowledge base an OWL artifact[15] [20] .

The ontology is hence constituted by two main taxonomies: locations and objects, among which there are physical as well as abstract objects (including events). The hierarchical relations between objects and between locations are used by the search engine to perform query expansion; sets of synonyms attached to each individual or class are also used at query time by the engine. The total number of individuals in the ontology is roughly 1000 (≈300 locations and ≈700 objects). The ontology has been built with a bootstrap process: a set of locations has been extracted from GeoNames (in particular, countries of the world with major cities), while the part on physical objects is in fact the Proton Ontology, an Upper Level Ontology by OntoText [21] . The ontology also keeps track of information about the type of sound emitted by the object – e.g. if the sound is repeatable.

Thanks to this data organisation, together with basic linguistic analysis (in particular, tokenization and lemmatization) the semantic search engine permits to find children as well as syntactical variants of the concepts expressed in the user query, gaining a simple yet effective set of IR features. Thanks to the linked data links between GeoNames and DBPedia[16] demonyms are also used by the engine: "french" finds "france" (and children locations) and viceversa.

6. SOUNDSCAPE GENERATOR AN SOUNDSCAPE COMPOSER

As shown in Figure 2 SoDA's design involves a Soundscape Generator (SSG), an autonomous and highly featured soundscape synthesis engine capable of modelling soundscapes with a variable degree of realism, and a Soundscape Composer (SSC), that acts as a mediator between the former and other parts of SoDA in oder to completely automate soundscape generation from user query.

SSG is designed as a versatile and multi-featured audio engine, that can work as an autonomous unit and that can be at the same time integrated modularly into SoDA. Several features had to be addressed while developing SSG. SSG is capable of modelling complex 3D spaces and of providing localisation of sound events within them and with respect to their acoustic properties. SSG works both in real and non-real time, in order to be used for both quick tests and experimentation as well as for its specialised role or in SoDA. It delivers sound in a variety of formats (mono, stereo, multichannel, etc). Finally, it can emulate and reconstruct complex soundscapes just by means of minimal sonic material. By addressing all these issues, SSG is innovative if compared with similar systems.

SSG proposes a conceptually straightforward model for soundscape generation. As shown in Figure 4, SSG generates audio by taking into account three elements: a "Space", a "Listener", and a "Decoder". These elements are coordinated and integrated by a "Renderer", that is also the external interface of the system.

The Space is intended as a model of the desired sound space. In this sense, SSG is related to soundscape modelling softwares rather than to DAW-like or abstract sound environments. The Space (see in general Figure 5) is built as an aggregate of an arbitrary number of individual "Zones", each provided with its individual geographical, acoustic and sonic profiles. Profiles are intended as sets of features, passibile of future improvements and enhancements. The Geographical profile refers to spatial boundaries and absolute positioning in 3D virtual space. The Acoustic profile

[20] Using a widespread terminological abuse, we here use the terms "knowledge base" and "ontology" interchangeably.

[21] http://www.ontotext.com/proton-ontology

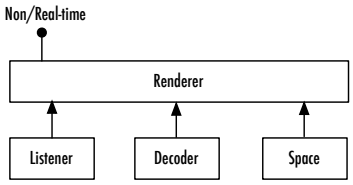

Figure 4. Soundscape Generator (SSG) structure.

refers to modelling physical phenomena (with a variable degree of realisms) such as reverberation, resonance, absorption, etc. Finally, the Sonic profile refers to the type of sound events that may occur within a Zone: these are modelled as arrays of Sources. SSG allows for different

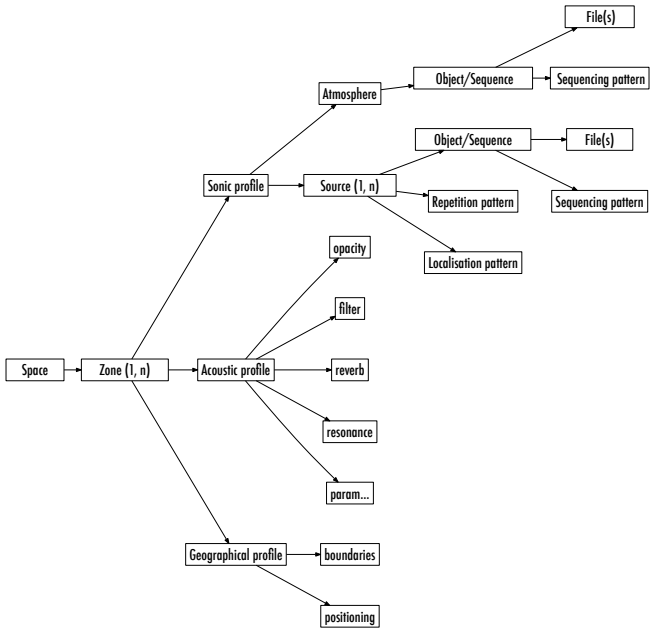

Figure 5. SSG: structure of Sound space.

kinds of Sources, all of which are nevertheless conceived as audio "Sequences" (or as simple sound "Objects") of some sort - their only differences lying in their spatial positioning and directionality. A Source is a data structure containing audio data as well as information about when and where they should be reproduced. Directional Sources may be either fixed in space or allowed to move in predefined trajectories. Other types have been also catered for, such as for instance "Cloud", that facilitates the modeling of events such as crowds, rain, etc. Cloud is to be understood as individual directional sounds that appear as if switching to random positions within a certain area and with respect to a density factor. Another special kind of Source are Atmospheres (see later), that are intended to represent non-directional background sound (the so-called "air" in sound designer's jargon).

Each individual Source is triggered for playback with respect to a pattern-based mechanism. Patterns are high-level data structures of arbitrarily complexity that define in a very synthetic way a temporal behaviour, whether one-shot or repetitive, deterministic or stochastic, and that can be arbitrarily combined linearly or recursively [22]. SSG provides the user with a conceptually straightforward way to model highly sophisticated events with minimal elements based on the numerous available patterns that may represent linear sequences, random selections from lists, probability-based number generators, random walks, mathematical constructs, and others. This feature is one of the major innovations of SSG when compared to similar systems. Indeed, rather than random permutations of sound sources, real soundscapes are characterised by specific spatial and temporal patterns. As an example consider a dog that barks in an irregular but not totally random way. It would be highly inefficient to mimic in detail such a behavior using traditional sequencing methodologies. Yet in SoDA it is trivial to model a highly realistic and ever-permuting dog barking by means of relying to patterns that describe its phenomenological properties and applying it to just a few one-shot bark recordings. A possible non-deterministic bark sequence based on 2 different barks could be the following, for instance: Bark A or Bark B or Silence should be played back for a random duration between 0.5 and 2 seconds and this whole pattern should be repeated a random number of times between 3 and 10. Then this entire sequence is the sonic element of a Source which in turn would use more pattern-based logic to repeat it irregularly in time and while our virtual dog is moving spatially.

SSG also implements different kinds of Listeners' profiles, depending on their spatial motion. It is possible to model listeners with different spatial positioning and ambulatory behavior as well as having different listening sensitivity. There are models for listeners that are fixed in space, ones that follow predefined trajectories and lastly ones that would randomly wander in space. Thus, by means of having a Listener move through Zones with different acoustic/sonic profiles, highly complex and specialized soundscapes may be easily modeled, e.g. the soundscape of a building from the perspective of someone who is running through its various rooms.

Finally, the third element of SSG is the Decoder, that manages the desired output format. Internally, SSG relies on ambisonics spatialisation algorithms [18] so that, given the opportune decoder, the same audio stream may be decoded to one of the standard formats such as mono, stereo, 5.1, 7.1 or even for reproduction from an arbitrarily configuration of speakers in 2D or 3D space - with canonical ones giving better results.

Thus, the Renderer takes into account all the properties defined for its Space, composes them with the properties of the Listener and generates audio in compliance with the Decoder.

While it is relatively easy to model sophisticated soundscapes with SSG, its usage still requires the user's intervention. On the contrary, SoDA asks for a fully automated soundscape composing paradigm. Soundscape Composer (SSC) has been, therefore, conceived as that part of the

[22] These kinds of data are extensively supported in the SuperCollider language, used in the implementation, see [17].

overall system that would interact with SSG and configure it accordingly to the results of the Semantic Search Engine. SSC address several tasks:

- Given the results of the semantical analysis it chooses and subsequently models a Space with adequate geographical and acoustic features.

In SoDA, the Space consists of a singleton zone having a fixed listener at its center. This is an example of the constraints that the Composer imposes to the Generator: in fact, a soundscape for SoDA (think about the reference to the ambiance in sound design) is intended as a background with some possible moving sound sources but where the listener is not moving [23]. Once the space has being designed, SSC will populate it by selecting sounds from the library and with respect to the results of the semantical analysis.

- SSC associates the Space with at least one Atmosphere.

The Atmosphere (a value from the TypeOfSoundObject tag) will be constructed from the files that are tagged as ambiences. It loops constantly to provide an acoustic background for the soundscape, ensuring a continuous layer so that no silent gaps exist in the final audio output.

- Individual Sources are then created by means of combining similar sounds (similarity here is defined with respect to their classification and their semantic properties) into audio sequences.

The last step involves a number of decisions on the temporal and spatial behaviour of the overall soundscape. A general leading criterium can be defined as "balance". The generated soundscape should be generally balanced in spatial, temporal, spectral, contextual and dynamics-related respects. Note that "balance" refers to that compromise between a certain variance (essential element of a realistic soundscape) and an well defined average state (for the results to be quantified and validated). Here, "realism" is an semiotic/empirical criterium that depends on sound design practice and cultural expectations. Apart from this general criterium, SSC has to also guarantee that the individual Sources generated not only satisfy the desired overall behaviour of the soundscape but are also meaningful in their own sake. That is to say that each individual Source corresponds to an actual real-life object and behaves in an appropriate way.

- As a default behaviour, that is, unless explicitly specified by some tags, SSC treats all available sounds as static, non-repeatable and mostly occurring in close proximity to the Listener and in the horizontal plane defined by his or her ears.

- Sounds that are tagged as "sequences" are never repeated (contrasting "sound objects") since in real life it is rare that complex sequences of sounds are repeated;

- SSC takes into account sounds tagged as "in motion" by spatialising them according to set of rules defined not in the document but in the ontology for certain classes (e.g. motor vehicles);

- In case of sound objects, SSC reconstruct meaningful, generative sequences out of them again by referring to set of rules available in the ontology their classes.

7. THE CURRENT IMPLEMENTATION

With respect to the architecture shown in Figure 2 and to the components described in the previous sections, the actual SoDA application implements a web service based on a client/server model (Figure 6). The user submits a query to the web interface. The query is intercepted by the SoundScape Composer, which forwards the query to the Semantic Search Engine. This forward mechanism is required as the Composer imposes some constraints to the semantic search in order to fill the minimal requirements for a soundscape synthesis. In particular, by using the faceted search, the Composer asks for a atmosphere and for n other sounds which constitute the atomic ingredients for the generation of the soundscape. The Semantic search engine answers with a set of documents from the Semantic index that relevant to the query. These documents refer to audio files. Each document is returned together with its metadata and with the associated individuals/classes in the ontology associated. The Composer is hence provided with information by the Semantic Search Engine and is thus able to retrieve the audio files from the Audio storage and parametrises the Generator that synthesises the soundscape. The resulting audio file is finally made available to download to the notified user.
Both the Composer and the Generator are at the moment entirely developed in SuperCollider, as the latter allows a high-level environment for audio programming together with an efficient non/real-time audio server [19]. In SoDA, SuperCollider is installed as a web server application. The Semantic Search Engine is a proprietary software by Celi srl, a Java webapp built on top of several open-source libraries, among which Apache Lucene [24] for the indexing and search, and Apache Jena [25] for managing RDF. The engine exposes REST APIs, and hence can be used by the other components via HTTP, keeping software integration at a minimum complexity.

8. CONCLUSIONS AND FUTURE WORK

By coupling semantic annotation and automatic generation SoDA aims a providing the sound designer a tool for fast prototyping that allows a trial-and-error methodology.
On the semantic side, SoDA both proposes an annotation schema and provides an annotated library. Through the annotation schema the library can be extended and customised, still allowing the search engine to operate with

[23] This example also demonstrates that SSG is designed as an abstract module with respect to SoDA.

[24] http://lucene.apache.org/ .

[25] https://jena.apache.org

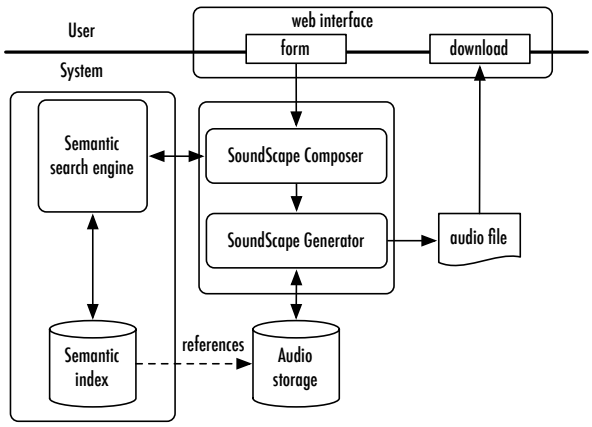

Figure 6. Overall organisation of the application.

it. Concerning soundscapes, as these are generated automatically, their production cost and time is very limited if compared to traditional soundscape practice of handmade composition with GUI. Thus, automation allows to match various production situations, from fast prototyping in preproduction to low-budget post-productions. Moreover, the reference to a space in SSG allow to automatically obtain a spatial coherence between various sound materials. Finally, the generative nature of SoDA is indeed a major point, as it allows to obtain always different results, even from the same user query.
Evaluation of the results has been at the moment conducted in an informal but constant way (as our team includes professional sound designers), and more systematic tests are planned. Such a feedback is indeed very relevant for SoDA, as the project aims at providing the sound designers a tool that can be used in real-world situations. Even if modifying an already annotated library could be a huge task, still the system can be easily improved by operating at different levels. In fact, on one side, the ontology of the Semantic Seach Engine can be modified or integrated to fit new requirements without altering the tagset. On the other side, the Soundscape Composer can be quickly tuned to take new requirements into account.

Acknowledgments

The SoDA project is founded by Regione Piemonte - Polo d'Innovazione per la Creatività Digitale e Multimedialità, III programma annuale. We thank our partners Vittorio Di Tomaso and Andrea Bolioli (Celi), Vito Martinelli (Zero dB) and Pietro Giola (Machiavelli).

9. REFERENCES

[1] D. L. Lewdall, *Practical Art of Motion Picture Sound*, 3rd ed. Burlington, MA: Focal Press, 2007.

[2] R. Beauchamp, *Designing Sound for Animation*, 2nd ed. Burlington, MA: Focal Press, 2013.

[3] A. Farnell, *Designing Sound*. Cambridge, MA: The MIT Press, 2010.

[4] O. Warusfel and G. Eckel, "LISTEN-Augmenting everyday environments through interactive soundscapes," *Virtual Reality for Public Consumption, IEEE Virtual Reality 2004 Workshop, Chicago IL*, vol. 27, 2004.

[5] A. Misra, P. R. Cook, and G. Wang, "Musical Tapestry: Re-composing Natural Sounds," in *Proceedings of the International Computer Music Conference (ICMC)*, 2006.

[6] A. Valle, V. Lombardo, and M. Schirosa, "Simulating the soundscape through an analysis/resynthesis methodology," in *Auditory Display*, ser. Lecture Notes in Computer Science, S. Ystad, M. Aramaki, R. Kronland-Martinet, and K. Jensen, Eds. Springer Berlin Heidelberg, 2010, vol. 5954, pp. 330–357.

[7] M. Schirosa, J. Janer, S. Kersten, and G. Roma, "A system for soundscape generation, composition and streaming," in *Prossime distanze. Atti del XVIII CIM - Colloquio di Informatica Musicale*, A. Valle and S. Bassanese, Eds., AIMI - Associazione Informatica Musicale Italiana. DADI - Dip. Arti e Design Industriale. Università IUAV di Venezia, 2011, pp. 115–121.

[8] G. Assayag, C. Rueda, M. Laurson, C. Agon, and O. Delerue, "Computer-assisted composition at IRCAM: From PatchWork to OpenMusic," *Computer Music Journal*, vol. 23, no. 3, pp. 59–72, 1999.

[9] M. Laurson, V. Norilo, and M. Kuuskankare, "PWGLSynth: A visual synthesis language for virtual instrument design and control," *Computer Music Journal*, vol. 29, no. 3, pp. 29–41, 2005.

[10] H. Taube, "An introduction to Common Music," *Computer Music Journal*, vol. 21, no. 1, pp. 29–34, 1997.

[11] T. Berners-Lee, "Linked data - design issues," *W3C*, no. 09/20, 2006. [Online]. Available: http://www.w3.org/DesignIssues/LinkedData.html

[12] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific American*, vol. 284, no. 5, pp. 34–43, May 2001. [Online]. Available: http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21

[13] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[14] P. Schaeffer, *Traité des objets musicaux*. Parigi: Seuil, 1966.

[15] W. OWL Working Group, *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation, 27 October 2009, available at http://www.w3.org/TR/owl2-overview/.

[16] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic Web Journal*, 2014.

[17] R. Kuivila, *The SuperCollider Book*. Cambridge, MA: The MIT Press, 2011, ch. Events and Patterns, pp. 179–205.

[18] D. G. Malham and A. Myatt, "3d sound spatialization using ambisonic techniques," *Computer Music Journal*, vol. 19, no. 4, pp. 58–70, 1995.

[19] S. Wilson, D. Cottle, and N. Collins, Eds., *The SuperCollider Book*. Cambridge, MA: The MIT Press, 2011.