SIG~: Performance Interface for Schaefferian Sound-Object Improvisation

Israel Neuman
Iowa Wesleyan College
Division of Fine Arts
isneuman@gmail.com

ABSTRACT

Pierre Schaeffer's theory of sound objects is a milestone in the historical development of electronic music. The TARTYP plays a central role in this theory. The TARTYP, however, is not widely accepted as a practical tool for musical analysis and composition, in part due to the large number of confusing and vague terms it introduces. This paper suggests a focus on Schaeffer's sound recordings that exemplifies the TARTYP as a source for aural learning of this taxonomy and an improvisational approach that explores the practical applications of the TARTYP to real-time composition and computer improvisation. A software based on the TARTYP generative grammars and a performance system supporting this improvisational concept are presented along with specialized graphic notation of TARTYP sound objects set in animated scores. Finally the paper describes performance practices developed for SIG~, a Schaefferian improvisation group based in Iowa City.

1. INTRODUCTION

Pierre Schaeffer's theory of sound objects is a milestone in the historical development of electronic music. The TARTYP (*Tableau Récapitulatif de la Typologie*), i.e., Summary Table of the Typology of Sound Objects, plays a central role in this theory. It is a schematic representation of a taxonomy of sound objects that demonstrates the premises of the Schaefferian theory. While Schaeffer's ideas set the path for major trends in electronic music, the TARTYP itself was not widely accepted as a practical tool for musical analysis and composition. Its impracticality is in part attributed to a large number of confusing and vague terms introduced by this theory [1], which I also address in prior related work [2, 3]. The challenge in understanding the TARTYP terminology is highlighted by the number of different translations and revisions introduced by Schaefferian scholars for the same TARTYP terms (see [1, 4, 5, 6]).

Dack [7] cites two reasons for the difficulty of understanding and translating the Schaeffer's terminology. The first one is that Schaeffer had to coin many new terms and expressions in the development of his theory. The second is that Schaeffer used "standard words in non-standard ways", e.g., the word *facture*. Dack discusses in detail the difficulty of translating and explaining the TARTYP term facture, which represents "an important concept in Schaeffer's theory." This term receives great attention also from Thoresen [1] and Normandeau [6]. Thus, the discussion of the TARTYP is centered mainly on explaining the semantics of Schaeffer's terminology through the use of natural languages, rather then any application of the TARTYP itself.

In contrast, Schaeffer devoted a great deal of attention to the construction of sound examples that demonstrate his ideas. The TARTYP sound-object classes[1] are exemplified in [8] with up to three examples per sound object class (see also [9]). One way to move beyond the debate on the semantics of the TARTYP terminology and to put this theory into practical use is to focus on Schaeffer's sound examples as the main source for aural learning of the TARTYP taxonomy, limiting the role of the textual explanation. Improvising musicians are accustomed to learning from audio examples. Transcriptions of recordings and imitation of performances are common tools among improvising musicians for mastering a musical language. An improvisation ensemble may therefore provide an ultimate environment for attempting to put the TARTYP to practical use.

SIG~ is a Schaefferian improvisation group based in Iowa City, Iowa, founded by the author for the purpose of exploring the practical applications of Schaeffer's TARTYP to real-time composition and computer improvisation. The group's focus is not on exact reproduction of Schaeffer's sound examples, which include a variety of sound sources not always suitable for generation in a real-time environment. Instead, members of the ensemble use these examples to imitate the behavior of sounds and to create their individual interpretation of the TARTYP sound objects, i.e., sounds that have the same defining characteristics. The mastery of the TARTYP musical language in SIG~ is supported by software designed specifically for this ensemble using the Pd-extended and Processing environments. A core element of the SIG~ performance system are generative grammars derived from the classification of sounds in the TARTYP which I have previously presented in [2, 3]. In this paper, I present the SIG~ performance system along with its soft-

[1] A sound-object class, a term coined by the author in [2, 3], is the class of all sound objects with the same TARTYP defining characteristics represented in the TARTYP by an alphanumeric symbol.

ware, interfaces and components. I also present the animated scores incorporated in the system's interfaces in which I have used specialized TARTYP graphic notation introduced elsewhere [1]. Finally I describe the ways by which the members of SIG~ interact with this system and its animated scores as well as the set of new performance practices that result from this interaction.

2. BACKGROUND

The TARTYP is a tabular representation of a sound object typology that was first published in [10]. This table classifies sound objects based on their characteristics in the time and frequency domains, and it introduces an alphanumeric notation for sound objects. Its structure suggests interrelationships between sub-collections of sound objects. Schaeffer introduced in the TARTYP new terminology that was essential for the development of his theory. Most studies of the TARTYP offer translations and revisions of the table and its terminology (see [1, 5, 6]). Michel Chion's *Guide to Sound Objects* [4] is lexical collection of Schaefferian terminology, including TARTYP terms, which is widely accepted by Schaefferian scholars. Its 2009 English translation [4], however, sparked new discussions regarding the semantics of this terminology [7]. Thoresen [1] aims towards the development of a practical analytical tool of electronic music based on Schaeffer's ideas. He reduces the number confusing TARTYP terms and suggests a set of graphic symbols that notate sound qualities portrayed by the TARTYP classification.

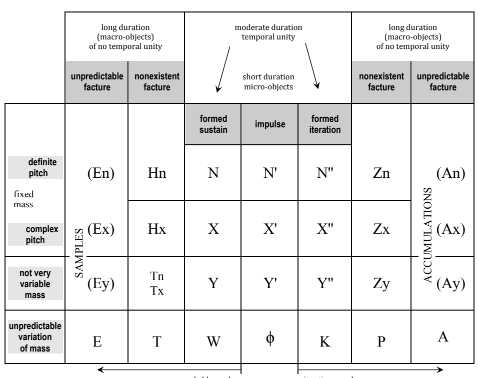

Figure 1. Highlighted in gray are time-domain terms along the upper row of the table and the frequency-domain terms along the leftmost column in Pierre Schaeffer's TARTYP (after [4]).

Sound-object characteristics are specified in the TARTYP at the margins of the table, with time domain characteristics along the upper row of the table and the frequency domain characteristics along the leftmost column (see Figure 1). Combinations of characteristics at the horizontal and vertical planes of the table describe the sound-object classes notated in the body of the table. Schaeffer divides the table into sub-collections of sound-object classes that are only partly notated in the table yet are well documented [4]. Figure 2 displays this division of the TARTYP into six sub-collections. I have previously described generative grammars derived from the TARTYP classifications of sound objects and its defining elements as well as from the structure of the table and its sub-collections [2, 3]. The rewrite rules of these grammars use the time and frequency terms specified at the margins of the table as terminal symbols. For each of the sub-collections of the table, I define a grammar in which a terminal equals a subset of the notated sound-object classes. A set of rewrite rules in a sub-collection grammar defines a space consisting of a large but finite number of paths where each of these paths can be composed out as a sequence of sound objects. In addition, I define a Table grammar that references each of the sub-collection grammars and establishes a unified hierarchical space of sound object sequences.

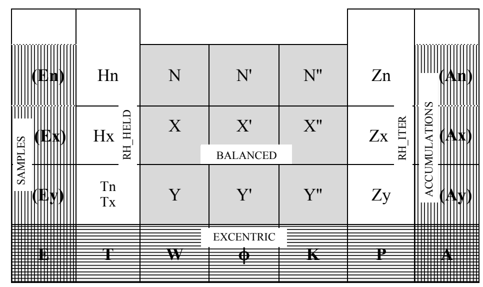

Figure 2. The division of the body of the TARTYP into sub-collections of sound-object classes.

I have used a generate-and-test algorithm designed to produce a set of rewrite rules. When a legal rule set is produced, a second algorithm is used to extract a path or a sequence of terminals, using a recursive rule-expansion technique. These algorithms were implemented as four Java classes and embedded in the Max/MSP or the Pd-extended environments to create grammar objects of the mxj type (or pdj in Pd). A grammar object creates (in real time) a set of rules in one of the grammars and then constructs (in real time) multiple paths from the same set of rules. The object receives a list of arguments in the left inlet and posts the set of rules in the Pd window. Following the generation of the rule-set, with each subsequent "bang" received in the left inlet the object outputs a path or a sequence of terminals as a symbol list. In the following sections I describe how the TARTYP generative grammars and the grammar object are incorporated as core elements in the SIG~ performance system and its interfaces.

3. INDIVIDUAL PERFORMER INTERFACE

The Individual Performer Interface (IPI) is a combination of applications developed in the Pd-extended and Processing environments that support performer-computer interaction based on the TARTYP taxonomy. The central element of the IPI is a TARTYP grammar engine that incorporates one or more grammar objects (see Figure 3). The latter is used to generate rule sets and paths within a specific TARTYP sub-collection grammar. Each IPI incorporates a specific set of grammar objects and therefore enables a performer-computer interaction associated with a sub-collection of TARTYP sound-object classes. The goal of the IPI is to allow the performer to create his or her interpretation of the sound-object classes of a sub-collection and to incorporate these sound objects in an interactive human-computer improvisation. Besides the TARTYP grammar engine, an IPI includes additional elements that support interactive human-computer improvisation such as a sound-object sampler, a spectral data extraction tool, signal processing tools and an animated score.

As mentioned above, a grammar object generates rule sets and outputs paths or a sequence of terminals as symbol lists. The grammar engine stores these paths or symbol lists using a Pd-extended patch with a **coll** object [2, 3]. The grammar engine uses the stored paths to generate time-based sequences of sound objects from the sub-collection associates with the IPI based on this sub-collection's grammar. For example, the IPI of the Balanced object sub-collection would include the performer's interpretation to the nine sound-object classes at the center of the TARTYP (N, N', N", X, X', X", Y, Y', Y"). The Balanced grammar uses a set of six terminals each of which is equivalent to a subset of the Balanced object sub-collection as specified in formula (1).

DEFINITE = {[N | N' | N"]+} (1)

COMPLEX = {[X | X' | X"]+}

VARIABLE = {[Y | Y' | Y"]+}

IMPULSE = {[N' | X' | Y']+}

FORMED_ITER = {[N" | X" | Y"]+}

FORMED_SUS = {[N | X | Y]+}

A path generated by the grammar object and stored in the **coll** object includes a sequence of terminals, e.g., IMPULSE, IMPULSE, IMPULSE, IMPULSE, FORMED_SUS, FORMED_SUS. The grammar engine translates such a path to a sequence of sound objects by randomly selecting a specific sound object from the subsets of Balanced objects equivalent to each of the terminals in the path. Hence the path above may be translated into the following sequence of sound objects: <N', N', X', Y', X, N>. Each of the alphanumeric symbols in this sequence represents the performer's interpretation for the sound-object class as described in the TARTYP. The grammar engine first constructs the sequence as a string of notation symbols that are presented to the performer in the animated score discussed in Sections 5 and 6. Following the construction of the sequence the IPI plays the sequence in a temporal organization derived from the temporal and spectral characteristics of the sound objects.

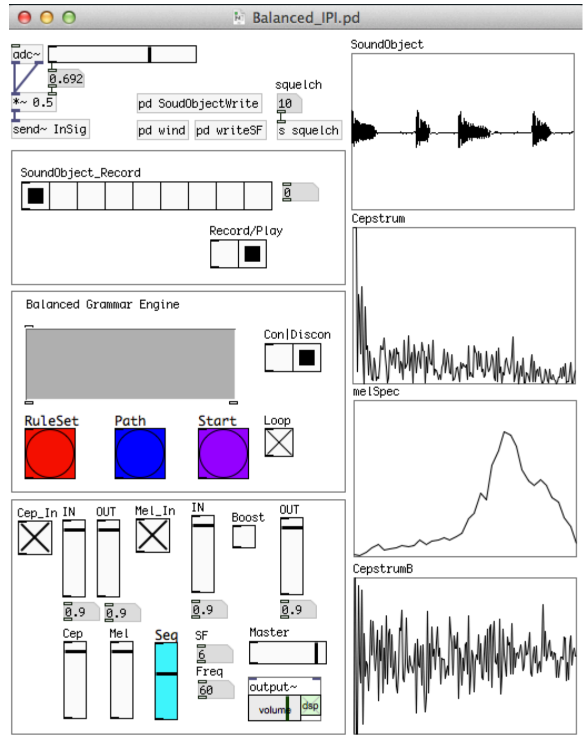

Figure 3. The IPI's Pd-extended application with a TARTYP grammar engine

The IPI has two modes: recording mode and playing mode. In the recording mode, the performer records his or her interpretation for the sound-object classes that are included in the TARTYP sub-collection which is associated with the performer's IPI (e.g., a performer using a Balanced IPI records interpretations for the sound-object classes N, N', N", X, X', X", Y, Y' and Y"). During this sampling stage the IPI extracts spectral data from the recorded sounds using timbral feature extraction objects from the Pd externals package timbreID [11]. This data is used to construct a spectral signature of the recorded sound object as well as in the processing of the live sound. In particular, each sound is analyzed for features extracted by a Fast Fourier Transform, as well as cepstrum and Mel frequency computations. Additional mapping is applied to these features to fit the needs of the live sound processing (e.g., cepstrum features characterized by a wide range of negative and positive numbers are mapped to positive numbers 0 to 1 for the purpose of spectral filtering). The IPI includes a script that writes in real time hundreds of text files including the data computed every few milliseconds in regard to each sound object. A collection of such text files associated with a sound object contains the spectral signature of this object.

After the completion of the sampling of sound objects, the performer changes the IPI to the playing mode. In this mode the IPI constructs sequences of sound objects. The IPI uses the spectral signatures created in the recording mode to process the live signal of the performer as well as the signal created by the playback of sound-object sequences. In particular, features from spectral signatures are loaded into spectral filters that filter both the live and playback signals. In addition, cepstrum features are mapped to control speed transformations of the playback of sound objects. Both the sequencing of sound objects and the signal processing correspond to the sequences generated by the grammar engine: the IPI plays the sound objects that match the grammar engine's sequence, and it selects features for the signal processing from the spectral signatures of the sound objects played in the sequence. This activity is framed in a temporal organization that is implied by the sequence itself. For example take the sequence mentioned above <N', N', X', Y', X, N>. The first sound to be played in this sequence is the performer's interpretation of N'. Throughout the duration of this sound spectral filters and speed transformations are controlled by features from the spectral signature of N'. Following the repetition of N' as dictated by the sequence, the performer's interpretation of X' is played and throughout its duration spectral filters and speed transformations are controlled by features from the spectral signature of X'.

4. GROUP PERFORMANCE SYSTEM

In the TARTYP generative grammars, the sub-collection grammars are unified by a Table grammar that reflects the structure of the TARTYP. The Table grammar is a collection of rule sets whose terminals correspond to the six names of the six TARTYP sub-collections (see Figure 2). A path generated in this Table grammar consists of sequencing invocations of the other six grammars accordingly. Schaeffer groups the sub-collections of Sample, Accumulation and Excentric objects into a larger collection also called Excentric [4]. In the TARTYP grammars this larger collection is implemented as a Sub-table grammar as shown in Figure 4. The latter represents the larger sub-collection of Excentric objects (the title Sub-table is used to prevent duplication). Hence, the term Excentric represents in the grammars only the smaller sub-collection of Excentric objects in the bottom row of the table.

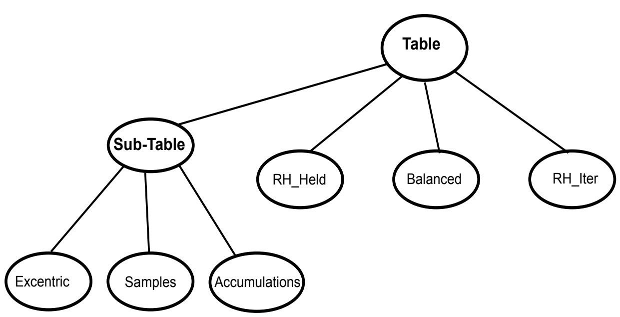

Figure 4. Grammar-based hierarchical structure enabled by Table and the Sub-table grammars. This tree-like hierarchy unifies all the types of grammar.

The tree-like hierarchy shown in Figure 4 is fundamental to the approach taken in the design of the SIG~ performance system. This group performance system includes three major components: individual performer interfaces (IPIs); an independent improvising processor (IIP); and a group network supported by a server (see Figure 5). The IIP runs an application similar to the IPI yet without interaction with a live performer. The IPIs and the IIP, each of which are in a specific TARTYP sub-collection grammar, are unified in the group performance system by a Table grammar engine which is running on the supporting server. The Table grammar engine ensures a structural organization that reflects the structure of the TARTYP. The group network supported by the server is used to enable communication between the Table grammar engine, the IPIs and the IIP.

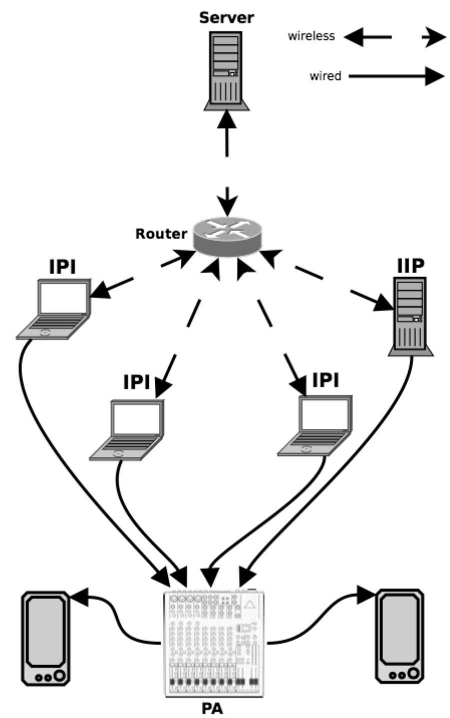

Figure 5. A system diagram of the SIG~ performance system

In the hierarchical structure shown in Figure 4, the Table and the Sub-table grammars provide the background level of the structural organization while the other grammars are the middle ground of this organization. Note that musical foreground is determined by the realization of paths as sequences of sound objects and in the case of SIG~ by the performers' actions. A path extracted in the Table or the Sub-table grammars would invoke the extraction of paths in the middle ground grammars. Similarly in the group performance system following the extraction of a path in the Table grammar engine messages are sent through the network to the IPIs and IIP that invoke the generation of new rule-sets

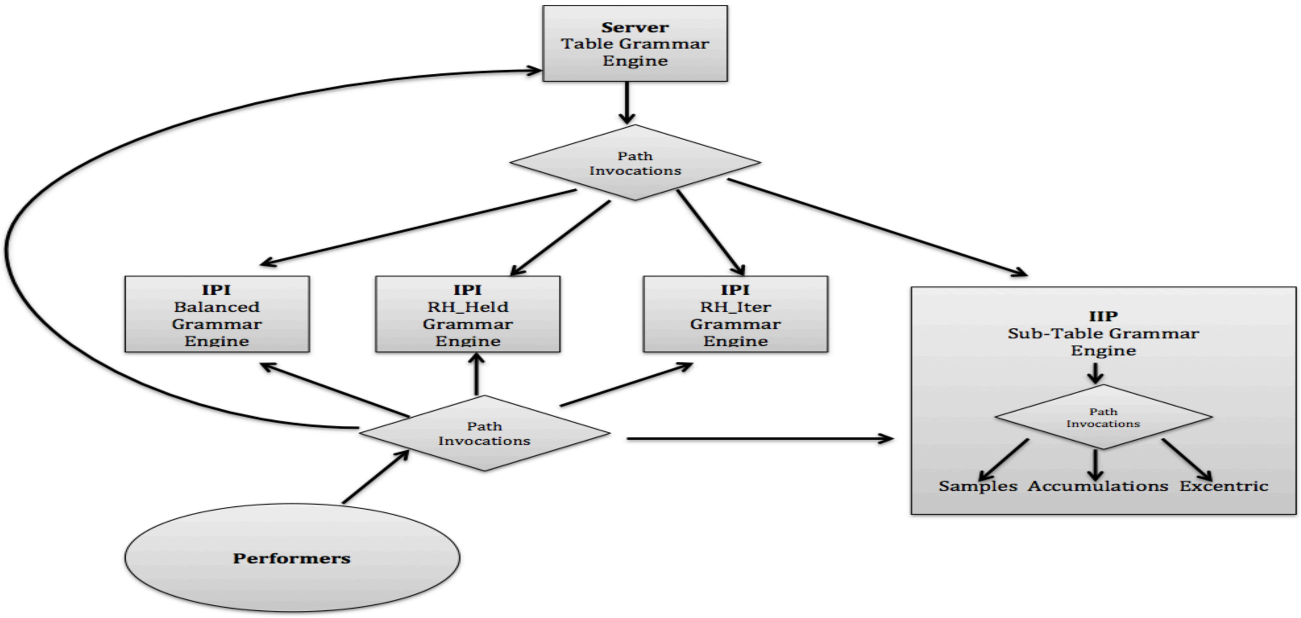

Figure 6. A diagram of the communication network incorporated in the SIG~ performance system

and the extraction of new paths. At the same time, performers may chose in any time to extract a new path in their own IPIs as well as to invoke the generation of a new rule set and the extraction of new paths in the Table grammar engine. Figure 6 is a diagram of the ensemble's communication network.

The TARTYP taxonomy describes some sound objects as more suitable for musical performance, namely the Balanced, RH_Held and RH_Iter sound objects, while other sounds, the Excentric, Sample and Accumulation sound objects, as less fitting or unfitting for music. Dack [5] however points out that much of today's electronic music is based on the Excentric, Sample and Accumulation type of sounds. In SIG~, I have taken the initial yet flexible approach of placing the Balanced, RH_Held and RH_Iter subcollections in the musical foreground and middle ground and the Excentric, Sample and Accumulation subcollections in the musical background. The musical foreground and middle ground are controlled mainly by the performers and the IPIs are in the Balanced, RH_Held and RH_Iter grammars. The role of the IIP is to manage Excentric, Sample and Accumulation sound objects at the musical background. Its application incorporates a Sub-table grammar engine. This Sub-table grammar engine is invoked through the network by the Table grammar engine or by one of the performers. In turn, the Sub-table grammar invokes the extraction of paths in the Excentric, Sample and Accumulation grammars.

The underlying structure generated by the TARTYP grammars implies a basic level of a temporal organization derived from the fact that a path extracted in the Table or the Sub-table grammars would be advanced in relation to the paths extracted in the middle ground grammars. If for example the path BALANCED, RH_HELD, BALANCED, RH_ITER is extracted in the Table grammar, when the first terminal is read it activates a path in the Balanced grammar. The second terminal will be read only when the path in the Balanced grammar ended. Nevertheless the temporal organization of a SIG~ performance is determined by and large by selections of system presets which are motivated by compositional choices as well as the real-time interaction of performers with the system.

5. GRAPHIC NOTATION

The IPI includes an animated score using notation based on Thoresen's [1] graphic notation of Schaefferian sound objects. Thoresen presents this notation as part of a revision of the TARTYP meant to create a more practical analytical tool. He chooses to replace Schaeffer's alphanumeric notation with graphic symbols. As known from traditional notation, very few graphical elements can concisely describe multiple musical dimensions such as pitch, rhythm, articulation and dynamics. In the first part of his revision process, Thoresen reduces the TARTYP into a minimal representation of 18 symbols representing only part of the 29 sound-object classes originally appearing in the TARTYP. In the second part of the revision, Thoresen introduces an expanded diagram with more than 62 symbols that account for the reduced objects but also for many additional variations of the original objects. Hence, Thoresen does not present a one-to-one replacement of Schaeffer's alphanumeric symbols with matching graphic symbols. Nevertheless, Thoresen's notation is consistent in handling the sound-objects characteristics used in the TARTYP: a circle represents pitch-based sound (*definite pitch*) while a square represents a noise-base sound (*complex pitch*); a solid line represents sustainment whereas a dashed line represents iteration; a dot represents short duration (*impulse*) where as longer line represents longer duration (*measured duration*); an angled line (dashed or solid) represents glissando (*variable*).

The IPI's animated score is an application written in the Processing programming language that communicates with the Pd-extended portion of the IPI through a UDP socket. The role of the animated score in the IPI is to convey to the performer musical structures, i.e., sound-object sequences which are generated by the grammar engines. Performers may interact with sound object sequences in different ways which are discussed in the following section. These sequences of sound objects emerge from the TARTYP generative grammars and therefore include sound objects from all 29 sound-object classes originally appearing in the TARTYP. The notation used in the animated scores replaces the 29 alphanumeric symbols used by Schaeffer in the table with 29 matching graphic symbols.

Figure 7 presents the notation used in the animated score. The figure is a table diagram displaying 29 graphic symbols. Each cell of the table includes one of Schaeffer's original alphanumeric symbol, and the graphic symbol that replaces it in the animated score. The graphic notation in the figure is closely based on Thoresen's notation. It maintains the same basic principles of symbolic representation of sound objects characteristics used by Thoresen (i.e., a circle represents pitch, a square represents noise, etc.). The graphic symbols used for Balanced objects (the nine objects at the center of the TARTYP) as well as the Accumulation and Sample objects (the right-most and left side columns) are closely based on Thoresen's notation for these objects in his minimal representation. The notation of the Excentric objects (the bottom row) extends the basic idea of Thoresen's minimal representation. In regard to the RH_Held and RH_Iter objects (the columns to left and right of the Balanced objects), I have reduced the number of variations suggested by Thoresen to a single graphic symbol per sound-object class.

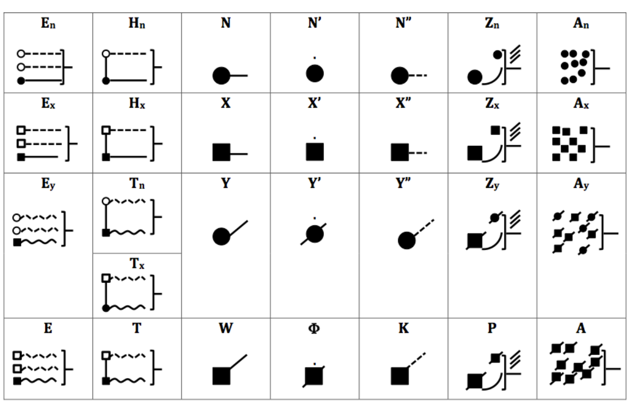

Figure 7. Overview of the SIG~ animated score notation. Graphic symbols based on [1] (reproduced by permission of the author).

The graphic notation of the animated score is presented to performers together with the sound files of Schaeffer's sound examples. Figure 8 shows an excerpt from the notation key presented to the performers. This key associates the sound files which are marked by the alphanumeric symbols with the graphic symbols. The notation key includes also short textual descriptions which are consistent with the notation's basic principles. These minimal descriptions are combinations of the terms pitch, noise, sustained, iterative, glissando, pulse, short and medium. They supplement the aural learning process of sound-object classes as they provide an alternative way of memorizing a general meaning of the graphic symbol. A key goal is to help the performer develop a more intuitive association with the symbol that emerges from listening to the sound examples.

Alpha-numeric	Graphic Symbol	Description
N		Pitch-sustained, medium duration (3 sec)
X		Noise-sustained, medium duration (3 sec)
Y		Glissando-pitch-sustained, medium duration (3 sec)
N'		Pitch-pulse, short duration (0.5 sec)
X'		Noise-pulse, short duration (0.5 sec)
Y'		Glissando-pitch-pulse, short duration (0.5 sec)
N"		Pitch-iterative, medium duration (3 sec)
X"		Noise-iterative, medium duration (3 sec)
Y"		Glissando-pitch-iterative, medium duration (3 sec)

Figure 8. Notation key of sound objects as presented to the SIG~ performers.

6. PERFORMANCE PRACTICES

The incorporation of the TARTYP classification and generative grammars in an improvisation-base performance calls for the development of a new set of performance practices. Additional performance practices are being developed in the SIG~ ensemble to manage the technical operation of the IPIs, the IIP and the group performance system. The members of SIG~ are introduced to the TARTYP language with a training package that includes a notation key and Schaeffer's sound examples. The goal of this training package is to allow the performers to become familiar with the TARTYP sound-object classes as they were portrayed by Schaeffer's sound examples. The intent is to create a framework for long-term collaborative exploration of these sound-object classes, rather then to learn a static notation system. The performers are instructed to use the training package as a tool for creating their own personal interpretation of sound-object classes, i.e., to create sounds with similar characteristics in regard to elements such as amplitude envelope, frequency content (pitch vs. noise), duration and variability (changes in frequency content throughout the sound's duration). Similar characteristics do not, however, imply direct reproduction or duplication of Schaeffer's sound examples.

Real-time interpretation and interactive presentation of sound objects presents a significant performance challenge. For this reason, in the SIG~ ensemble the creation of new sound objects to fit a specified TARTYP grammar is separated from the interactive playback of these sound objects in performance; this separation sidesteps what is likely to be a significant human-computer interaction challenge. A performer first uses the IPI to record his or her personal interpretation of the sound-object classes from a specific TARTYP generative grammar. The number of sound objects that could be stored in an IPI by each performer matches the number of sound-object classes included in the TARTYP sub-collection associated with the IPI's grammar. For example, an IPI in the Balanced grammar can store nine sound objects. A performer may choose to record new sound objects (e.g., in the beginning of a new composition) yet that would entail erasing previously recorded sound object.

Following the recording of sound objects, the performer switches the IPI to playing mode. In this mode, the grammar engine constructs sequences of sound objects using the recorded sounds. The grammar engine constructs these sequences in two phases: it first converts a path extracted by the grammar object into a string of sound-object classes and then it plays the recorded sounds that match these sound-object classes. Similarly, the entire sequence appears at first in the animated score as a string of graphic symbols and then a red cursor moves from symbol to symbol indicating the time progression (see Figure 9). The performer can interact with these sequences in different ways. For example, using a USB foot pedal the performer may command the grammar engine to construct a new sequence at any time. In addition, the performer may choose between looping a sound-object sequence and playing it a single time by changing IPI parameters. The performer may also choose to musically relate to the sequence in different ways including following the notated sequence, using it as an accompaniment (looping), or creating a counterpoint line.

Sound objects recorded by performers in SIG~ rehearsals are being archived. These sound files as well as excerpts from the ensemble's recordings form the base for the creation of pre-composed material. This pre-composed material includes two main elements: new sound objects and larger-scale pre-composed sound events. The new sound objects are incorporated in the application running by the IIP that includes the Sub-Table grammar engine. These new sound objects are the group's interpretation of the sound-object classes from the Excentric, Sample and Accumulation sub-collections. The sound events are used to further structure the musical material and trigger new ideas for improvisation.

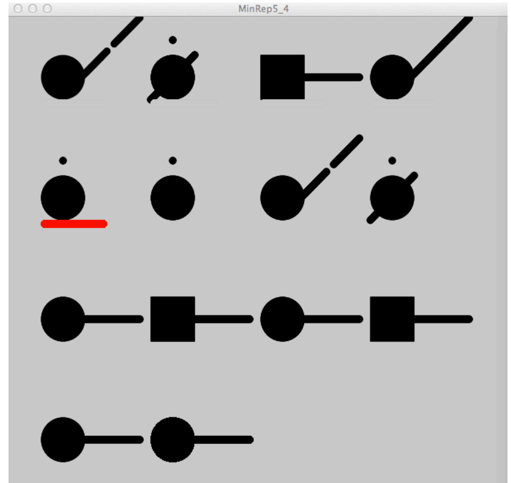

Figure 9. The graphic notation of a sequence of sound object displayed in the animated score. The red cursor highlights the currently playing sound object.

7. CONCLUSIONS

The premise of this paper is the idea that improvisation and aural learning could free Schaefferian theory in general and the TARTYP in particular from a sort of semantic impasse that prevents them from becoming practical analytical and compositional tools. In this paper, I presented interfaces that support performer-computer interaction in the context of the TARTYP sound-object taxonomy. This software was developed as part of the performance system used by the ensemble SIG~, a Schaefferian improvisation group that explores the practical applications of Schaefferian theory to real-time composition and computer improvisation. I also presented animated scores that use a graphic notation of sound-object classes based on Thoresen [1]. Finally I discussed performance practices that emerged from the intersection of the TARTYP and human-computer improvisation.

8. REFERENCES

[1] L. Thoresen, "Spectromorphological analysis of sound objects: an adaptation of Pierre Schaeffer's typomorphology," *Organised Sound* 12 no.2 (2007): 129-141.

[2] I. Neuman, "Generative Grammars for Interactive Composition Based on Schaeffer's TARTYP," *Proceedings of the International Computer Music Conference*, Perth, 2013.

[3] I. Neuman, "Generative Tools for Interactive Composition: Real-Time Musical Structures Based on Schaeffer's TARTYP and on Klumpenhouwer Networks," *Computer Music Journal* 38 no. 2 (2014):63-77.

[4] M. Chion, *Guide to Sound Objects: Pierre Scaeffer and Musical Research*, J. Dack and C. North trans. Paris: Editions Buchet/Chastel, 2009. Available online at www.ears.dmu.ac.uk/IMG/pdf/Chion-guide (accessed January 9, 2013). (originally published 1983).

[5] J. Dack, "At the limits of Schaeffer's TARTYP," *Proceedings of Music Without Walls? Music Without Instruments? Conference*, 2001. Available online at www.dmu.ac.uk/documents/art-design-and-humanities-documents/research/mtirc/nowalls/mww-dack.pdf (accessed October 23, 2013).

[6] R. Normandeau, "A revision of the TARTYP published by Pierre Schaeffer," *Proceedings of the Seventh Electroacoustic Music Studies Network Conference,* 2010. Available online at www.ems-network.Org/ IMG/pdf_EMS10_Normandeau.pdf (accessed July 10, 2012).

[7] J. Dack and C. North, "Translating Pierre Schaeffer: Symbolism, Literature and Music," *EMS: Electroacoustic Music Studies Network*, Beijing, 2006.

[8] P. Schaeffer and G. Reibel, *Solfe`ge de l'objet sonore*. New editions: 1988, 2005 (1966b).

[9] P. Schaeffer, "Schaeffer's Typology of Sound Objects," in *Cinema for the Ear: A History and Aesthetics of Electroacoustic Music*, 2012. http://www.dmu.uem.br/ aulas/tecnologia/SolObjSon/HTMLs/Schaeffer.html (accessed July 10, 2012).

[10] P. Schaeffer, *Traité des objets musicaux*. Paris: Éditions du Seuil 1966.

[11] W. Brent, "A Timbre Analysis And Classification Toolkit For Pure Data," *Center for Research in Computing and the Arts University of California,* San Diego, 2009. http://williambrent.Conflations.com/ papers /timbreID.pdf (accessed January 9, 2015)