http://cnmat.berkeley.edu/publications/xml-based-sdif-stream-relationships-language

An XML-based SDIF Stream Relationships Language

Matthew Wright, Amar Chaudhary, Adrian Freed, Sami Khoury, Ali Momeni, Diemo Schwarz*, David Wessel
CNMAT, UC Berkeley, 1750 Arch St., Berkeley, CA 94709, www.cnmat.berkeley.edu
*IRCAM,1 Place Igor Stravinsky, 75004 Paris, France, www.ircam.fr
{matt,amar,adrian,khoury,ali,wessel}@cnmat.berkeley.edu, *schwarz@ircam.fr,

Abstract

We introduce the SDIF Stream Relationships Language ("SDIF-SRL"), a formal language for describing the relationships among streams in an SDIF file. SDIF-SRL is based on XML, the "Extensible Markup Language," an emerging standard for data modeling and representation. We describe the structure of SRL and its use in several applications.

1. Introduction

SDIF, the "Sound Description Interchange Format," (Wright, et al., 1999, Wright, et al., 1998) is becoming the computer-music community's standard representation for many kinds of sound descriptions. SDIF represents all sound descriptions as "streams" of "frames" over time, each frame consisting of matrices of numerical or text data. An SDIF *entity* (either an SDIF file or SDIF data streamed over a network) may be an aggregate of 2 or more streams.

While this format is adequate to represent any particular sound description, it does not include any mechanism to represent the relationships among multiple sound descriptions in an entity. An SDIF entity may contain time-domain samples and spectra in two streams, but what does it mean for the two streams to be in the same entity?

We introduce the SDIF Stream Relationships Language ("SDIF-SRL"), a formal language for describing the relationships among streams in an SDIF entity. We designed SDIF-SRL to be powerful enough to express all of the stream relationships we could think of, i.e., all of the reasons to put two or more streams together in the same entity. An SDIF entity can be self-describing by including an SDIF-SRL description of the relationships among its streams.

SDIF-SRL is defined using XML, the Extensible Markup Language, an important new standard for creating structured document types and describing them formally (DuCharme, 1999).

We chose XML for many reasons. XML is human-readable, so without any XML tools you can still read an SDIF-SRL document. XML is receiving widespread industry support, so we expect to take advantage of a plethora of freely available XML parsers, editors, and other tools.

2. What is XML?

XML, the "Extensible Markup Language," is a document format standardized by the World Wide Web Consortium (www.w3c.org). XML can be understood in relation to HTML, the "HyperText Markup Language" that is the basis of the World-Wide Web. The "markup" in both XML and HTML consist of *elements* delimited by *tags*, e.g.,

```
<P align="center">This is a paragraph.</P>
```

In this example of a paragraph element, the tags are `<P align="center">`, the start tag, and the matching end tag `</P>`. The start tag also contains an *attribute* called "align" with a value "center." XML has the same syntax for tags, with less-than and greater-than characters for delimiters, a slash indicating the end tag, and the same syntax for attributes. Unlike HTML, XML requires strictly proper nesting of elements; no implicit end tags are allowed.

XML and HTML differ greatly in what the tags may be and what they mean. HTML has only one purpose, hypertext documents to be displayed in a web browser, and therefore has a fixed set of tags. In contrast, each XML document has a "document type" that determines what elements may appear in the document, and the structure of how they must nest.

XML's extensibility comes from the ability to define new document types formally. The definition of an XML document type unambiguously determines whether any particular document is valid, guaranteeing that a parser will accept only syntactically correct documents. There are currently two mechanisms for defining XML document types: "Document Type Definitions" ("DTDs") and the newer, more powerful "XML Schemas" (Thompson, et al., 2000).

Many XML document types (also called "XML applications") have already been defined (www.oasis-open.org, www.xml.org), including real estate listings, mathematical expressions, bibliographies, customer profiles, chemical structures, and western musical notation. Each of these document types has its own list of elements and grammatical rules. For example, RELML, the "Real Estate Markup Language" (defined at openmls.com), has an element called `RESIDENTIAL-LISTING` that must include an element `GENERAL` that must include the elements `PRICE`, `WHEN-BUILT`, and `LOCATION`. This document type guarantees that every `RESIDENTIAL-LISTING` will have a `PRICE`.

3. Representing SDIF Stream Relationships in XML

We have defined SDIF's Stream Relationships Language as an XML document type.

Defining an XML document type mainly consists of defining elements and the rules for how they may nest. Therefore, to leverage the most expressive power from our use of XML, we represent each stream and each stream relationship with an XML element.

Each SDIF stream is identified by its "Stream ID," an arbitrary 32-bit integer that appears in the header of all frames in the stream. In SDIF-SRL, a stream is represented by an empty `stream` element (i.e., a stream element with nothing between the start tag and the end tag) with an attribute `id` giving the Stream ID as a decimal numeral. For example, the stream with

ID 544321 would be represented in SDIF-SRL as

```
<stream id="544321"></stream>
```

XML allows this shorthand for an empty element:

```
<stream id="544321"/>
```

Sometimes it is necessary to change a stream's stream ID, e.g., if duplicate stream IDs are found when merging two SDIF entities. Since stream IDs appear in SDIF-SRL only in the `id` attribute of a `stream` element, they are easy for a program to find and change.

Each kind of relationship in SDIF-SRL is also represented as an element. For example, the *spatializes* relationship indicates that the first stream gives spatial coordinates (over time) specifying how the sound described by the second stream is to be spatialized. For example, here's how to say that stream 1 spatializes stream 2:

```
<spatializes>
  <stream id="1"/>
  <stream id="2"/>
</spatializes>
```

In the definition of the SDIF-SRL document type, we specify that each `spatializes` element must contain exactly two `stream` elements. We document the semantics separately, to say that the first of the two streams is the spatialization data and the second stream is the sound description to be spatialized.

Both methods for defining an XML document type include a mechanism for adding new syntactic constructs to an existing document type. This allows creators of not-yet-standard SDIF-SRL relationship types to append the definitions of these types to the standard SDIF-SRL definition, and also provides a mechanism for updating the standard SDIF-SRL definition in the future.

4. What SDIF-SRL is Not

SDIF-SRL was designed to provide functionality orthogonal to what exists already in SDIF. SDIF-SRL is therefore designed not to represent sound descriptions, which should be represented with SDIF's existing frame/stream mechanism. SDIF-SRL is also designed not to overlap with SDIF's Name/Value table matrix type, which provides a database of textual name/value associations.

5. Applications of SDIF-SRL

5.1. SDIF as a sound file database

SDIF's 1TDS frame type represents time-domain samples. Putting several 1TDS streams together in an SDIF file makes a flat sound-file archive, equivalent to a large directory of sound files. By using SDIF-SRL to represent relationships among these sounds, the SDIF file becomes a structured database.

5.2. Grouping Analysis Results into a Single SDIF File

The desire to have a single SDIF file containing all of the results of an analysis/synthesis process was the original motivation for allowing an SDIF file to contain multiple streams. For example, the sinusoidal analysis/resynthesis package from IRCAM in use at CNMAT takes in time-domain samples and outputs the sound descriptions listed in Table 1.

In the days before SDIF, each of these resulting sound descriptions would be stored in a separate file, grouped together by their being in the same directory and by having a common filename prefix. For example, input sound foo.aiff would produce files such as foo.f0, foo.pics, foo.synt.aiff, and foo.noise.aiff.

We can represent these sound descriptions with SDIF and merge them all into a single file. Each stream can include a Name/Value Table indicating which program created it, what arguments were used with that program, when the program was run, etc.

Stream ID	Frame Type	Description
1	1TDS	Original input sound
2	1STF	STFT results for the STFT used by the fundamental frequency tracker
3	1FQ0	Fundamental frequency ("F0") estimate over time
4	1STF	More STFT results, this time with a window size based on the F0
5	1PIC	Spectral Peaks found in the stream above
6	1HRM	Pseudo-harmonic Sinusoidal Tracks based on the F0 estimate and the spectral peaks
7	1TDS	Phase-accurate resynthesis of the sinusoidal tracks
8	1TDS	Residual "noise" containing nonsinusoidal portion of original signal

Table 1: Sound descriptions produced by analysis

```
<sdif-srl>
  <analysis>
    <sources><stream id="1"/></sources>
    <results><stream id="2"/></results>
  </analysis>
  <analysis>
    <sources><stream id="2"/></sources>
    <results><stream id="3"/></results>
  </analysis>
  <analysis>
    <sources><stream id="1"/>
             <stream id="3"/></sources>
    <results><stream id="4"/></results>
  </analysis>
  <analysis>
    <sources><stream id="4"/></sources>
    <results><stream id="5"/></results>
  </analysis>
  <analysis>
    <sources><stream id="5"/>
            <stream id="3"/></sources>
    <results><stream id="6"/></results>
  </analysis>
  <analysis>
    <sources><stream id="6"/></sources>
    <results><stream id="7"/></results>
  </analysis>
  <residual>
    <stream id="7"/> <stream id="1"/>
    <stream id="8"/>
  </residual>
</sdif-srl>
```

Example 1: SDIF-SRL document describing the relationships among the streams in Table 1

We use the SDIF-SRL relationship `analysis` to indicate that a given set of streams came from an analysis of another set of streams. Each `analysis` element must contain the two elements `sources` and `results`, each of which must contain one or more `stream` elements. An `analysis` element may also contain an optional `provenance` element specifying a stream with a Name/Value table of the analysis program and parameters used. The `residual` relationship is a special case with exactly three `stream` elements, indicating that the third stream is the remainder of subtracting the second from the first. (A-B=C).

Example 1 is an SDIF-SRL document indicating the `analysis` and `residual` relationships among the streams in Table 1, assuming the stream IDs listed in the leftmost column.

This allows us to store the results of many analyses of the same input sound in a single SDIF file, either using different analysis parameters or entirely different analysis techniques.

What kind of programs would parse this SDIF-SRL document, and what would they do with it? An analysis-process browser graphically displays the dependency graph of the streams in an SDIF file and synthesizes each resulting sound description on demand.

5.3. Loudness Matching

Although precise control of loudness is crucial to working with sound, subjective loudness is a slippery psychoacoustic percept. One method for dealing with loudness is "loudness matching," where a given listener adjusts the gains of a set of sounds to make them all sound equally loud.

The *same-loudness* relationship indicates that a set of streams all have the same subjective loudness to a particular listener. SDIF-SRL's `same-loudness` element must begin with a `listener` subelement, to indicate who perceived the streams as equally loud. Each `stream` subelement may be followed by a `gain` element indicating the amplitude that the stream was adjusted by to make it match the common loudness level. (The default gain, of course, is 1.)

```
<sdif-srl><same-loudness>
  <listener>Les N. Ear</listener>
  <stream id="23894"/><gain>2.3432</gain>
  <stream id="435534"/>
  <stream id="8734"/><gain>0.696</gain>
</same-loudness></sdif-srl>
```

5.4. Marking Features for Timbral Interpolation

Timbral Interpolation or "sound morphing" is a desirable musical effect. The CAST ("CNMAT Additive Synthesis Tools") synthesizer can interpolate among sinusoidal track models, providing an effective form of timbral interpolation (Freed and Wright, 1998).

One of the issues in performing high-quality timbral interpolation is the need to mark equivalent sonic features in the sound models being interpolated. By analogy, visual morphing requires marking of visual features, e.g., "nose," for the interpolation to preserve meaningful structure. By indicating the nose in the two images being morphed, it's possible to ensure that all of the intermediate images will have something that looks like a nose.

As a first example, suppose we are interpolating between a 1-second stream and a 2-second stream. Naïvely, for each time t in the output, we look up data at time t in the first stream and interpolate it with the data at time t in the second stream.

$$Output(t) = weight \times Input_1(t) + (1 - weight) \times Input_2(t)$$

Clearly, this will produce a problem after one second, because there will be no data in the first stream after that time. Instead, it might be better to time-scale the two streams to be the same duration, so that data at time .75 in the first stream will be interpolated with the data at time 1.5 in the second stream.

$$Output(t / Duration(Output)) =$$
$$weight \times Input_1(t / Duration(Input_1)) +$$
$$(1 - weight) \times Input_2(t / Duration(Input_2))$$

Now consider a more difficult example. Suppose we have two somewhat percussive sound descriptions of the same duration, but that stream A has a rapid 10 ms attack while stream B has a slow 100 ms attack. Again, for each time t in the output, we would be tempted to interpolate between the data from time t in each of our input sound descriptions. At time 10ms, stream A will be making its loudest contribution to the result, while stream B will still be making relatively little. Just after time 10, stream A's contribution will diminish while stream B's continues to increase. Eventually, at time 100, stream B will peak, and then afterwards the result will start to decay. The interpolated result will have a double attack or a smeared attack, unlike either of the inputs.

The solution is to mark the temporal feature "attack" in each stream, and for the interpolator to be aware of temporal features. SDIF's time marker data type can be used to mark these features, and SDIF-SRL's `time-marking` relationship can associate the streams containing the markers with the sound descriptions they annotate.

The interpolator's job is much easier when interpolating between sound descriptions with the same number and kinds of temporal features. If one sound is a ten-note phrase and the other is a slow glissando, even marking the times of the ten notes will not help the interpolator. But a user could mark 10 regions of the glissando with the idea of mapping those to the ten notes. In this case the two time marker streams would have an `equivalent-feature-marks` relationship.

5.5. Timbre Spaces

Timbre spaces are a powerful and intuitive musical control structure for timbral interpolation (Wessel, 1979). A timbre space consists of multiple sound descriptions arranged geometrically. Typically, perceptual similarity among the sounds corresponds to proximity in the space. During synthesis the performer moves through this space; at each instant the synthesized timbre is an interpolation of the sound descriptions in the space, weighting the geometrically nearest sound descriptions most heavily.

Naturally, a performer would also like to control pitch, loudness, and duration of synthesized sounds, features that by definition are not "timbre." There are many ways to do this, including trivial frequency and amplitude scaling, more perceptually plausible models of pitch and loudness, and interpolation techniques similar to those used in timbre spaces. In any case, complex structures of multiple sound descriptions are required to provide satisfactory musical control (Wessel, et al., 1987). For example, the "timbres" in a timbre space might be parametric models capable of producing a range of pitches and loud-

nesses. Alternatively, there may be multiple timbre spaces each composed of sound descriptions with constant pitch and loudness, with a higher-level interpolation model to produce desired pitches and amplitudes from the outputs of each timbre space.

We use SDIF to represent a timbre space in a form usable for synthesis. The sound descriptions in the timbre space are of course represented with SDIF streams. SDIF-SRL groups sound descriptions into the complex structures required by different interpolation models.

One set of relationship types groups together sound descriptions according to what they have in common; this includes the *same-loudness* relationship introduced above as well as *same-pitch* and *comparable-timbre*[*].

SDIF-SRL's *timbre-space* relationship groups a set of timbres, each with its own timbre space coordinates, into a timbre space. The "timbres" in a timbre space may be streams or more complex structures.

We allow the coordinates of each timbre to change over time, not necessarily in synchrony with the frames of a sound description; therefore we represent the coordinates in a separate stream. (This allows an SDIF entity efficiently to contain multiple timbre spaces that are geometric rearrangements of a common set of sound descriptions.) All of the timbre-space coordinate streams in a *timbre-space* relationship must have the same number of dimensions and the same scale for the axes.

5.6. Sonic Rendering of Multichannel Sound

Typical distribution of multichannel sound is based on making assumptions at the production stage about the environment in which the sound will be diffused, e.g., number and position of loudspeakers, room size, ambient noise conditions, etc.

We propose a model in which multichannel sound is rendered locally and adaptively from SDIF entities that contain the sound descriptions to be played and high-level instructions about where they should be spatialized, how loud they should be, etc.

We store spatialization data in a separate stream from the sound description being spatialized, because the desired spatial location of a sound can change over time, not necessarily in synchrony with the frames of the sound description. This requires the `spatializes` SDIF-SRL relationship described above to relate the two streams.

Likewise, the relative gains indicating how the sound descriptions should be mixed are stored in a separate stream and related to the sound descriptions with the `mix` relationship.

This system could be extended with a more sophisticated control of loudness. In a pristine listening environment the mix would be as specified above, but to account for ambient noise, e.g., in a night club or airplane, the most important streams could be processed with multi-band compression to ensure their audibility.

[*] We resist using the notion of *same* when timbres are compared across different pitches and dynamics. Unlike pitch and loudness, each of which can remain invariant with respect to timbre, we know of no evidence from perceptual experiments that timbre can be invariant with respect to pitch or loudness. We use *comparable* to imply that the timbre is produced by the same model or physical device or is related to other timbres in the space in an analogous manner at the new pitch and/or loudness level.

6. Future Work

We have not yet formally defined the semantics of relationship types, in particular the SDIF frame types that are legal in each role of each relationship. We are working on an XML-based formal language for defining these relationship type semantics.

Our ongoing timbre space work will no doubt discover new mechanisms for control of pitch, loudness, and timbre, resulting in new SDIF-SRL *timbre-space* structures.

There is often no reason to keep intermediate analysis results. With the sources for a given analysis step and a complete *provenance* name/value table, we can recompute the results as necessary. Work is underway to implement such a system.

Perhaps SDIF-SRL could be extended to express relationships among parts of an SDIF file other than streams, for example, documenting why extra matrices appear in a frame or addressing individual channels (i.e., columns) of 1TDS streams.

XML includes a facility called Xlink (World Wide Web Consortium, 1998), which "allows elements to be inserted into XML documents in order to create and describe links between resources." This could allow relationships to be expressed among SDIF streams residing in different entities.

SDIF-SRL could support real-time synthesis of groups of sound descriptions by declaring computational limits such as maximum number of sinusoids, amplitude limits, etc. We could define a relationship type to mean "all of these streams can be synthesized together in real-time in this given environment."

7. Acknowledgments

Ahm Lee, Xavier Rodet.

8. References

B. DuCharme (1999), *XML: The Annotated Specification*. Saddle River, NJ: Prentice-Hall. http://www.w3.org/TR/REC-xml

A. Freed and M. Wright (1998), "CAST: CNMAT's Additive Synthesis Tools," http://www.cnmat.berkeley.edu/CAST

H. S. Thompson, D. Beech, M. Maloney, and N. Mendelsohn (2000), "XML Schema Part 1: Structures" http://www.w3.org/TR/xmlschema-1

D. Wessel, D. Bristow, and Z. Settel (1987), "Control of Phrasing and Articulation in Synthesis," proceedings of the International Computer Music Conference, Champaign, Urbana, USA.

D. L. Wessel (1979), "Timbre space as a musical control structure," *Computer Music Journal*, vol. 3, num. 2, pp. 45-52.

World Wide Web Consortium (1998), "XML Linking Language (XLink): Working Draft" http://www.w3.org/TR/WD-xlink

M. Wright, A. Chaudhary, A. Freed, S. Khoury, and D. Wessel (1999), "Audio Applications of the Sound Description Interchange Format Standard," proceedings of the Audio Engineering Society 107th Convention.

M. Wright, A. Chaudhary, A. Freed, D. Wessel, X. Rodet, D. Virolle, R. Woehrmann, and X. Serra (1998), "New Applications of the Sound Description Interchange Format," proceedings of the International Computer Music Conference, Ann, Arbor, Michigan.