Supporting Deaf Sign Languages in Written Form on the Web

Antônio Carlos da Rocha Costa, Graçaliz Pereira Dimuro

Universidade Católica de Pelotas, Brazil

{rocha, liz}@atlas.ucpel.tche.br

Abstract

The SignWriting system (developed at the Center for Sutton Movement Writing, Ca., USA), is a writing system for deaf sign languages. In SignWriting, signs are represented as figures made up of graphic symbols, each symbol schematically representing an important aspect of the sign gestures: hand configurations, hand and finger movements, contacts, facial expressions, etc. SWML (SignWriting Markup Language) is an XML-based format that we are developing for the storage and processing of SignWriting texts and dictionaries, allowing the interoperability of SignWriting savvy applications and promoting web accessibility to deaf people in their own natural languages. SSS.svg is a standard representation of the whole SignWriting symbol sequence as SVG symbols, allowing the flexible rendering of signs written in SignWriting as graphical elements in web pages. This paper initially presents the SignWriting system and its importance as a deaf sign language writing system and considers why supporting written sign languages on the web should be considered both as an accessibility issue and as a problem of multiculturalism and multi-linguality in the web. Then, the paper reports on the state of the work being done on the development of SWML (current status: version 1.0 - draft 2) and states preliminary ideas for the organization and searching of sign language text database. Finally, the paper shows the prospects for and the current status of the development of a SignWriting-based web application (SW-WebMail) using SWML and SSS.svg to render sign languages.

Keywords: Deaf sign languages, SignWriting, SignWriting Markup Language, SWML, Accessibility, Multilingual support

1. Introduction

The SignWriting system was developed by Valerie Sutton, of the Center for Sutton Movement Writing (Ca., USA), as a writing system for deaf sign languages. As is widely know, deaf people have not yet established a standard way for putting their native sign languages in written form, and the SignWriting system is one of the systems that are currently being proposed for such end. Alternative systems for writing sign languages exist, and they have already been compared (see papers in the SignWriting linguistic forum).

As deaf people don't get tired to explain, the implication of deafness is a linguistic and cultural difference between deaf and hearing persons: deafness implies a particular way of people developing their communication functions during their development in early childhood, and with this language difference comes other differences (cognitive, etc.) that impact both the way deaf children develop and the way they should be educated (see papers in the SignWriting education forum).

Moreover, during adulthood, communication in sign languages implies particular ways of dealing with social interactions (at home, among friends, in the work, etc.) so that deaf people are naturally led to organize deaf communities within the hearing societies in which they are being hosted. Such deaf communities have their own social structures (values, habits, jokes, traditions, history, etc.), developed in such a degree that they constitute in fact a culture of their own, the so-called Deaf Culture [1, 2].

This paper is organized as follows. Section 2 gives a brief introduction to the SignWriting system, giving examples of its symbol set and of the way symbols are put together to represent signs. It also explains why we have chosen it as the preferred means for writing deaf sign languages.

Section 3 presents SWML (SignWriting Markup Language), an XML-based format that we are defining for the interoperable storage and processing of sign language documents. Section 4 gives examples of SignWriting symbols represented as SVG symbols, showing how the full SignWriting Symbol Sequence will be represented in a standard way in the single SSS.svg file, which is can be used as a symbol file (analogous to a font file) when rendering stand-alone SWML documents, as well as when inserting SWML islands in HTML documents.

Section 5 tackles the problem of searching sign language texts. We present a simple means that we have devised for searching such texts. Given the graphical nature of SignWriting, a graphical pattern matching method is needed, which can deal in controlled ways with the personal variations people can imprint in the way they write signs. The solution we present introduces a concept of graphical coherence between symbols, allowing for a sort of fuzzy graphical pattern matching procedure for signs.

Finally, Section 6 presents the prospects for an SWML based web application that we started to develop, namely, a sign language webmail system (SW-Webmail). The Conclusion, indicating short-term and long-term future works, as well as including a call for cooperation, is in Section 7.

2. The SignWriting System

2.1 History and background

The SignWriting System was created and by Valerie Sutton, for the Center of Sutton Movement Writing, in 1974. It reached it present, stable version, around 1985 (see the SignWriting history). Since 1995, the system is present in the Web, in the site http://www.signwriting.org/.

Various systems, other than SignWriting, have been proposed for writing deaf sign languages (e.g., HamNoSys - the Hamburg Notation System; Stoke system - derived from the work of the linguist that pioneered sign language linguistics). Such systems usually employ alphanumeric characters, with specially assigned meanings, for representing linguistic aspects of signs, usually aiming at the linguist developing specialized work in sign language linguistics. As such, they are more of a notation system for writing *about* sign languages, then a notation system for writing *in* sign languages.

The SignWriting system, on the other hand, was conceived within the general program of the Sutton Movement Writing initiative, which is that of representing movements as such, as they are visually perceived, and not for the eventual meaning that such movements can be transmitting. For instance, in the realm of dance, a DanceWriting system was devised, where dance movements are represented (choreographed) without concern to the meanings the dancer may attempting to transmit to the audience.

With respect to sign languages, the SignWriting system was invented within the same general approach. It provides means for representing signs, as they are visually perceived, as organized sequences of gestures, not for the eventual linguistic content they may be conveying.

This brings a lot of advantages, from the viewpoint of a writing system for sign languages. First, the system can be used by linguists to write *about* sign languages, because it provides a means for the representation of the syntactical and so-called phonetic and phonologic aspects of signs, in a way that is clearly neutral with respect to meanings.

On the other hand, as the system is based on a set of graphical and schematic symbols that are highly intuitive, and as it uses simple rules for combining symbols into signs, it provides a simple and effective way for common deaf (and hearing) people, that have no special technical training in sign language linguistics, to write *in* sign languages.

This intuitiveness seems to be the main driving force for the systems increasing acceptance among people interested in written forms for sign languages. Another main driving force for such increasing acceptance is the SignWriter program, a very easy to use graphical editor for sign language texts, developed by Richard Gleaves, around 1995. The program is available as shareware from the SignWriting website.

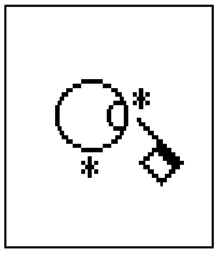
Also, written signs have been use extensively as icons in pages of sign language websites, a feature that has contributed to the spreading of SignWriting.

In summary, due to the system's emphasis on the purely visual aspect of the signs, it is looks as the most promising current approach to a true writing system for sign languages. And that is the reason for which we have chosen it as the writing system employed in our sign language processing research work, which at the present emphasizes the deployment and processing of sign language documents on the web.

2.2 Writing signs in SignWriting

In the SignWriting system, a sign is represented by a set of symbols. The whole set of symbols is called the SignWriting Symbol Sequence (see the SignBank site).

A typical sign written in the SignWriting system is the following:



The sign for "deaf" (the same in ASL and in LIBRAS)

The main classes of symbols are: heads and hands (obvious in the sign for "deaf"), movements (usually arrows, not illustrated in the sign for "deaf") and contacts (illustrated by the asterisk).

Symbols can be submitted to many kinds of transformations. Typical are the various positions in which one single hand configuration can be used (palm facing the signer, back facing the signer, etc.). Also, the relative positioning of the various symbols, and the rotations in which they can be placed, are important transformations.

The set of information concerning a symbol being used in a sign comprises, at the end, seven items:

- shape number (indicating the symbol outline)
- fill (a code for the way the outline is filled, generally indicating its facing to the signer)
- variation (indicating complementary transformations)
- rotation (in steps of 45 degrees, which is a discretization with enough linguistic relevance)
- flop (indicating if the symbol is mirrored or not)
- x and y (the symbol coordinates within the sign-box containing the sign to which it belongs)

2.3 Sample writings in SignWriting

To give the reader a feel for signs written in SignWriting, we reproduce here some signs and their approximate glosses in English. The signs are taken both from the American Sign Language (ASL) and the Brazilian Sign Language (LIBRAS).

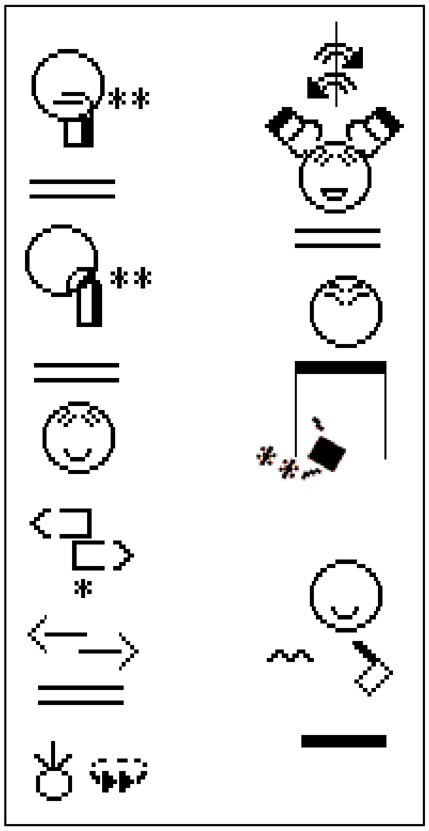
On the Web, such signs are usually shown in the GIF format, so that their rendering is immediate in any browser. The SVG rendering of signs is analyzed in section 3.

2.3.1 Sample icons in American Sign Language (ASL)

Home
SignWriting

About
SignWriting

2.3.2 Sample text in Brazilian Sign Language (LIBRAS)

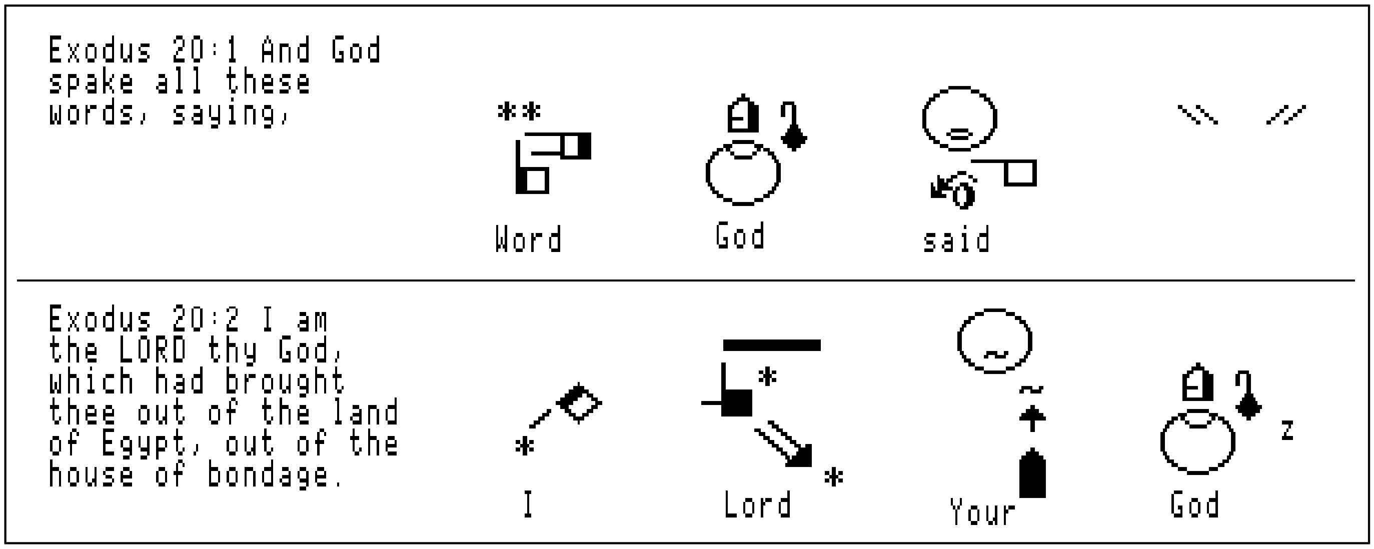


The Story of Goldilocks

"Papa, and Mama, and Baby, three bears, they hungry, they want to eat."

(First lines of the Goldilocks story, written in LIBRAS, in the vertical mode)

2.2.3 Sample text in American Sign Language (ASL)


Exodus 20:1 And God spake all these words, saying,
Word
God
said
Exodus 20:2 I am the LORD thy God, which had brought thee out of the land of Egypt, out of the house of bondage.
I
Lord
Your
God

[The Ten Commandments]

(The first lines of the Ten Commandments, The Signed Bible, by Pastor Ron Dettloff)

3. SWML - SignWriting Markup Language

SWML, the SignWriting Markup Language, is an XML-based format for the storage and processing of sign language documents written in the SignWriting system, and for insertion of sign language texts in HTML documents.

The current version of the SWML DTD (version 1.0, draft 2) is the following:

```
<!ELEMENT swml  (generator? ,  (sw_table | sw_text ) )>
<!ATTLIST swml  version   CDATA  #REQUIRED
                symbolset CDATA  #FIXED 'SSS-1995' >
<!ELEMENT generator  (name , version )>
<!ELEMENT name (#PCDATA )>
<!ELEMENT version (#PCDATA )>
<!ELEMENT sw_table (sw_table_defaults? ,  table_length , table_entry* )>
<!ELEMENT sw_text (sw_text_defaults? ,  (sign_box | text_box | new_line )* )>
<!ELEMENT sw_table_defaults (sign_boxes? , glosses? )>
<!ELEMENT sw_text_defaults (sign_boxes? , text_boxes? )>
<!ELEMENT sign_boxes (unit , height? , width? )>
<!ELEMENT text_boxes (boxtype , unit , height? , width? )>
<!ELEMENT boxtype (#PCDATA )>
<!ELEMENT unit (#PCDATA )>
<!ELEMENT height (#PCDATA )>
<!ELEMENT width (#PCDATA )>
<!ELEMENT table_length (#PCDATA )>
<!ELEMENT table_entry (sign_box , gloss )>
<!ELEMENT new_line EMPTY>
<!ELEMENT gloss (#PCDATA )>
<!ATTLIST gloss separator CDATA #IMPLIED
                field_names CDATA #IMPLIED >
<!ELEMENT text_box (chr* )>
<!ELEMENT sign_box (symbol* )>
<!ELEMENT chr (#PCDATA )>
<!ATTLIST chr x CDATA #REQUIRED
              y CDATA #REQUIRED >
<!ELEMENT symbol (shape , transformation )>
```

```
<!ATTLIST symbol x CDATA #REQUIRED
                 y CDATA #REQUIRED >
<!ELEMENT shape EMPTY>
<!ATTLIST shape number CDATA #REQUIRED
                variation CDATA #REQUIRED
                fill CDATA #IMPLIED >
<!ELEMENT transformation EMPTY>
<!ATTLIST transformation rotation CDATA #REQUIRED
                         flop CDATA #REQUIRED >
<!ELEMENT glosses (separator , field_name* )>
<!ELEMENT separator (#PCDATA )>
<!ELEMENT field_name (#PCDATA )>
```

A sample SWML file is the following:

```
<?xml version="1.0"?>
<swml version="1.0-d2" symbolset="SSS-1995">
   <generator>
      <name>Sign Writer</name>
      <version>4.3</version>
   </generator>
   <sw_text>
      <sw_text_defaults>
         <sign_boxes>
            <unit> pt </unit>
            <height> 60 </height>
         </sign_boxes>
         <text_boxes>
            <box_type> graphic_box </box_type>
            <unit> pt </unit>
            <height> 60 </height>
         </text_boxes>
      </sw_text_defaults>
      <new_line/>
      <sign_box>
         <symbol x="20" y="9">
            <shape number="215" fill="1" variation="0"/>
            <transformation rotation="3" flop="0" />
         </symbol>
         <symbol x="15" y="33">
            <shape number="114" fill="1" variation="1"/>
            <transformation rotation="7" flop="0" />
         </symbol>
         <symbol x="15" y="27">
            <shape number="87" fill="1" variation="0"/>
            <transformation rotation="0" flop="0" />
         </symbol>
         <symbol x="23" y="28">
            <shape number="0" fill="1" variation="1"/>
            <transformation rotation="1" flop="0" />
         </symbol>
      </sign_box>
   </sw_text>
</swml>
```

Essentially, the DTD says that an SWML document is either an sw_text (a text generated by an SWML aware SignWriting editor) or an sw_table (a sign language database or dictionary, generated by an SWML aware SignWriting application).

An sw_text is made up of sign_boxes and text_boxes, where each sign_box contains a sign (set of symbols) and each text_box contain an alphanumeric string (oral language text incorporated in a sign language text).

An sw_table is made up of table_entries, where each table_entry contains a sign_box (the sign in the table) and a gloss (typically, in a dictionary, a gloss in an oral language of the sign contained in the sign_box). Glosses are allowed to have optional internal structure (defined as a sequence of fields).

Both sign_boxes and text_boxes may have graphical features that vary from box to box. Default values for such features may be optionally stated for the whole document.

A note should be added: the SWML encoding of SignWriting texts considers only the graphical features of such texts. This is no dismissing of the importance of semantic issues in sign language processing.

The point is that SignWriting is not a system for writing the *meanings* of signs, but just the *gestures* that constitute them, in the same way that the latin alphabet is not used to write the *meanings* of words of oral languages, but is used just to write the *sounds* that comprise those words.

That is the reason why no intrinsically linguistic concept was used to struture the representation of signs in SWML. Only concepts related to the graphical building of SignWriting texts were necessary, when defining SWML.

Going beyond the graphical level would mean to start *interpreting* sign texts, that is, going much further then merely providing for the interchangeability of SignWriting files, just as going beyond the character-set level would mean the same for files containing oral language texts.

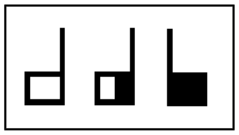
4. Rendering signs with SVG

The symbols of the SignWriting system are usually rendered in the Web as GIF files, obtained by screen capture from the SignWriter program (the sole SignWriting aware program existing up to now !).

Given an SWML representation of a SignWriting file, using vector graphic formats for rendering the file may be much more interesting, due the greater flexibility allowed by such format.

SSS.svg is a reference file that assigns an SVG symbol for each SignWriting symbol, thus allowing the rendering of signs in vector format. Signs may be rendered from SWML files by simply grouping SVG symbols together and rendering them, under scripting control, either at the server or at the client side.

A sample of SVG symbols representing some SignWriting symbols is the following (abbreviated):



```
<svg width="8" height="14">
   <rect style="stroke:#000000;
       stroke-width:1;
       stroke-opacity:1;
       fill:#000000;
       fill-opacity:0"
       x="7" y="0"
       width="1" height="8" />
   <rect style="stroke:#000000;
       stroke-width:1;
       stroke-opacity:1;
       fill:#000000;
       fill-opacity:0"
       x="0" y="8"
       width="8" height="6" />
</svg>
<svg width="8" height="14">
    <rect style="stroke:#000000; .../>
    <rect style="stroke:#000000; .../>
    <rect style="stroke:#000000; .../>
</svg>
<svg width="8" height="14">
    ...
</svg>
```

5. Matching procedure for searching in sign language texts

Having document servers able to serve sign language texts stored in SWML would be almost useless if one couldn't define searching procedures for such texts.

However, there is a major problem in searching sign languages texts: dealing with the small graphical variations people can imprint in the way they write the same signs. The SignWriting system distinguishes explicitly some graphical properties of the symbols, like rotation and flop, for example, but does not distinguish (neither identify) tiny variations due to vertical and/or horizontal displacements of symbols within signs, because such values aren't discretized in the system (as opposed to, e.g., rotation, which can only assume a few set of possible discretevalues).

The solution we've found to allow the user to control the criteria to be used for judging on the similarity of two signs is to define a kind of degree of similarity between the component symbols of a sign, assuring that two corresponding symbols should have the same symbol type, rotation and flop, but allowing them to have some variation on their relative positions within the respective signs.. This kind of similarity is formalized here as a parameterized, reflexive and symmetric relation, that we call **sign similarity relation**.

5.1. Basic Definitions and Geometric Considerations

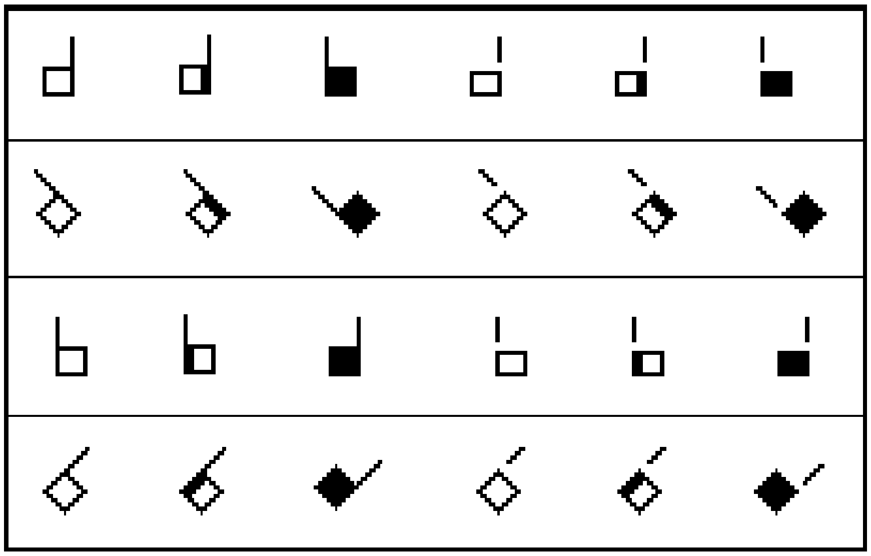
Each SignWriting **symbol** can be characterized, in principle, by three basic features: its shape number, its filling information, and its so-called variation.

Let *s* be a SignWriting symbol. Let *t* be its symbol shape, let *f = 0,1,2,3,4,5* be its filling information (which, e.g., codes both the way hands are facing the signer and if they are in a horizontal or in vertical plane), and let *v*, the variation, be complementary information about the symbol. The **identification tuple** for symbol *s* can thus be defined as the tuple *ids=(t,f,v)*, allowing us to identify the symbol with its identification tuple.

A set of symbols that represent the same essential linguistic information, having the same symbol shape and differing only in their filling or variation information, is called a **symbol group**. Each symbol group is a class of equivalence represented by the **basic symbol**, whose identified tuple is *ids = (t,0,0).*

For any symbol *s* (with its shape, fill and variation information) of any symbol group, *s* can - in principle - be given two additional spatial information: **rotation**, denoted by *r*, and **flop**, denoted by *fl*, where *r = 0,1,2,3,4,5,6,7* indicates a counter clockwise rotation applied to *s* (given in intervals of 45 degrees), and *fl* is a Boolean value indicating if the symbol is vertically mirrored or not, relative to the basic symbol. A symbol with such additional information is called an **oriented symbol**, and is identified by the tuple (*ids*, *r*, *fl*), where *ids* is the symbol identification tuple.

Example: The symbol group called **index**, whose symbols represent hands with index finger straight up and closed fist, is shown in the figure below. Each symbol *s* in the group is identified by a tuple *ids* = (*0,f,0*): shape number 0, *f=0,1,...,5* (from left to right in the figure), and variation 0. The symbols in the first line have the basic orientation (no rotations, no flop) and are identified by codes of the form (*ids,0,0*). In the second line, a rotation of 45 degrees was applied to each symbol, and the symbols in that line are thus identified by (*ids,1,0*). In the third and fourth lines, there are flopped symbols, identified by (*ids,0,1*) (with no rotations) and (*ids,7,1*) (with rotations of 315 degrees).

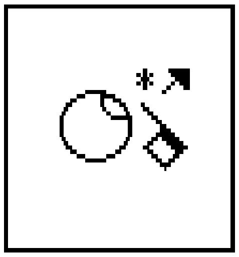


The group of symbols called *index*, which represents hands with index finger straight up and closed fist, and some of its rotated and flopped symbols.

A **symbol-box** is the least box that contains a symbol. It is identified by *sb* = (*x,y,w,h*), where *x* and *y* are, respectively, the horizontal and vertical coordinates of the upper left corner of the symbol_box (relative to the upper left corner of the sign-box containing the symbol_box – see below), *w* is its width and *h* is its height. Symbol_boxes serve the purpose of indicating the placement of symbol instances within signs. A **symbol instance**, that is, an occurence of an oriented symbol within a sign, is defined as a pair *si* = (*s;sb*), where *s* = (*ids;r;fl*) is an oriented symbol and *sb* is a symbol box.

A **sign**, denoted by *Sg*, is a finite set of symbol instances. A **sign_box** is the least box that contains a sign. It is identified by *Sgb* = (*w,h*), where *w* is the box's width and *h* is its height. A **sign instance** is a sign *Sg* together with its sign_box *Sb* and an index *j* indicating the position of the sign within a sign sequence (a phrase in sign language).. It is represented by the tuple *Sgi* = (*Sg;Sgb;j*).

Remark: All the definitions presented above are reflected in the SWML format. Note, in particular, that as defined above, sign_boxes (and consequently, sign instances) have no coordinate information. This is so because, in the current version of SWML, sign language texts are treated as simple strings of signs, with no formatting information. The format in which texts are rendered is left completely to the discretion of the application.



A sign instance for the sign "idea" in LIBRAS.

Example: The representation in SWML of the sign for "idea" in LIBRAS (see figure above) is:

```
<sign_box>
    <!-- sign "idea" in LIBRAS -->
    <symbol x="20" y="9">
         <!-- the head -->
         <shape number="215" fill="1" variation="0"/>
         <transformation    rotation="3" flop="0" />
 </symbol>
 <symbol x="15" y="33">
         <!-- the arrow -->
         <shape number="114" fill="1" variation="1"/>
         <transformation rotation="7" flop="0" />
    </symbol>
    <symbol x="15" y="27">
         <!-- the asterisk -->
         <shape number="87" fill="1" variation="0"/>
         <transformation rotation="0" flop="0" />
    </symbol>
    <symbol x="23" y="28">
         <!-- the hand -->
         <shape number="0" fill="1" variation="1"/>
         <transformation rotation="1" flop="0" />
    </symbol>
</sign_box>
```

5.2. Sign Similarity Relation

The **sign similarity relation** is a parameterized, reflexive, symmetric and not transitive relation, that we introduce here to analyze the relative similarity between two sign instances, providing for the construction of matching procedures for signs and sign language expressions.

The sign similarity relation has to embody an admissible variation between the relative positions of the symbol instances within the two corresponding sign instances, taking into account a degree of significance for this variation (relatively to the symbol-boxes dimensions), as determined by the user.

The admissible variation in symbol instances positions is expressed in percentages by a so-called **minimum degree of similarity**, denoted by *e*.

Consider two **symbol instances** $si_1 = (s_1; sb_1)$ and $si_2 = (s_2; sb_2)$, with $sb_1 = (x_1, y_1, w_1, h_1)$ and $sb_2 = (x_2, y_2, w_2, h_2)$, respectively. si_1 and si_2 are said to be ***similar with at least degree e***, denoted by $si_1 \approx_e si_2$, if and only if the following conditions hold:

- Structural similarity:

$$s_1 = s_2$$

which implies that

$$w_1 = w_2$$

and

$$h_1 = h_2$$

- Admissible horizontal variation:

$$|x_2 - x_1| \le kw$$

- Admissible vertical variation:

$$|y_2 - y_1| \le kh$$

where

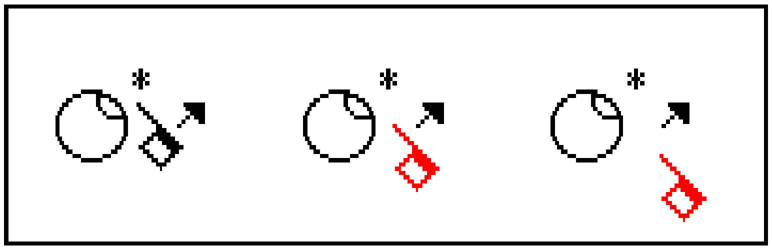
$$k = \frac{\varepsilon}{100} \ge 0$$

Two **sign instances** $Sgi_1 = (Sg_1; Sgb_1)$ and $Sgi_2 = (Sg_2; Sgb_2)$ are said to be **similar with at least degree e** if and only there exists a bijection

$$f : Sg_1 \to Sg_2$$

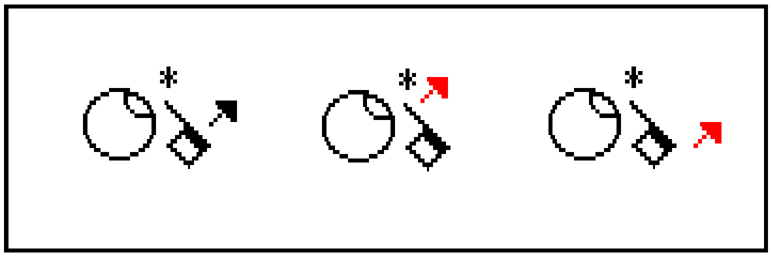
such that for each $s \in Sg_1$

$$s \approx_{\varepsilon} f(s) \in Sg_2 .$$



Three instances of the sign "idea" in LIBRAS, which may be similar or not, depending on the minimum degree of similarity that is required by the user.

Example: Consider the three sign instances for the sign "idea" in LIBRAS (see figure above). Observe that they contain three different instances for the symbol "index", each one with a different *y* coordinate (all other symbol instances are exactly equal). Consider a situation where a user is searching for that sign in a text. Suppose he writes the first sign instance as the sign to be searched and that the other two instances are present in the text. The later two instances have some degree of similarity with respect to the first sign instance. In spite of the fact, in a strict sense, they are graphically different from the first instance. They may be considered to represent the same sign, depending on the minimum degree of similarity required by the user from the results of the matching process. With an intermediate degree of similarity, the second instance would match the first, while the third instance would not (the hand is too low in comparison with its position in the first sign instance). With a low degree of similarity, all instances would match. If the user required total similarity, no instance would match. The total degree of similarity ($k_h, k_v = 0$) requires that no difference be admitted between the two sign instances being compared.



Three sign instances for the sign of "idea" in LIBRAS, which must always be considered similar, independently of the degree of similarity required by the user.

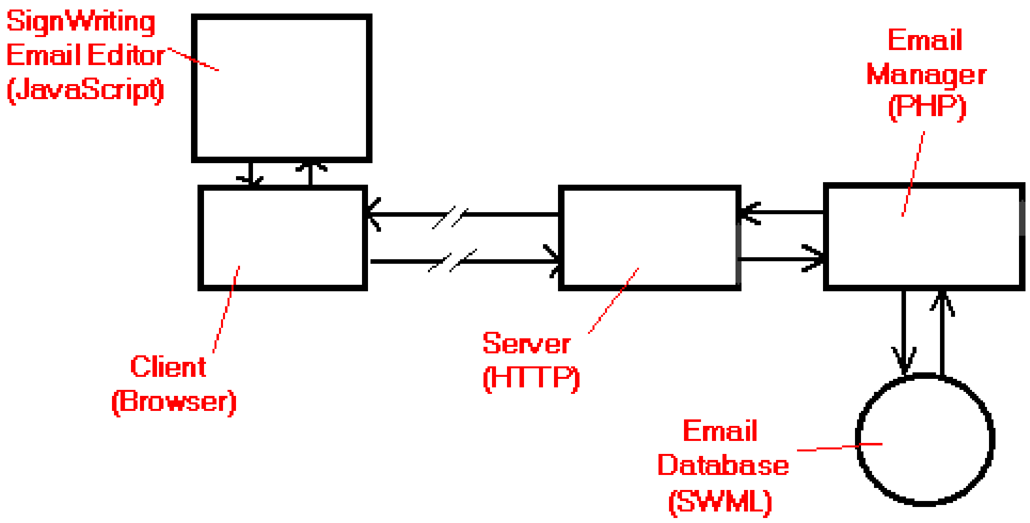
Example: The similarity relation defined above does not take into account some important (and frequent) exceptions. Such exceptions are mainly related to symbols like the "arrow" symbol encountered in the sign "idea" in LIBRAS (see figure above), whose position within the sign is absolutely not critical. Such symbols have most of their meaning completely encoded in their shapes and transformations, and the place where they are put in the sign-boxes is essentially irrelevant. For instance, the "arrow" symbol in the sign for "idea" means that the right hand moves in the horizontal plane, in the indicated direction, and this information is the same, wherever the "arrow" is placed in the sign-box. In such cases, the relative position of the symbol-box within the sign-box is not important. In the examples of the figure above, even if a rigorous or a total degree of similarity is required, the match process should find that those three sign instances are similar. On the other hand, for symbols like the "asterisk", almost no variation of the its position should be allowed, since it indicates a position where two components of the sign (head, hands, etc.) touch each other, when the sign is performed, and even low degrees of variations may imply linguistically relevant differences between the signs.

Remark: SWML, as currently defined, has all information needed to allow the matching procedure sketched here to be fully performed. The treatment of exceptions is to be embedded in the matching process, requiring from SWML only that it identifies symbol instances completely, which it already does.

6. Prospects for an SWML-based application: SW-WebMail

To illustrate the use of SWML in the storage and processing of sign language texts, we are building SW-WebMail, an SWML-based webmail service.

The following figure shows the overall architecture of the system.

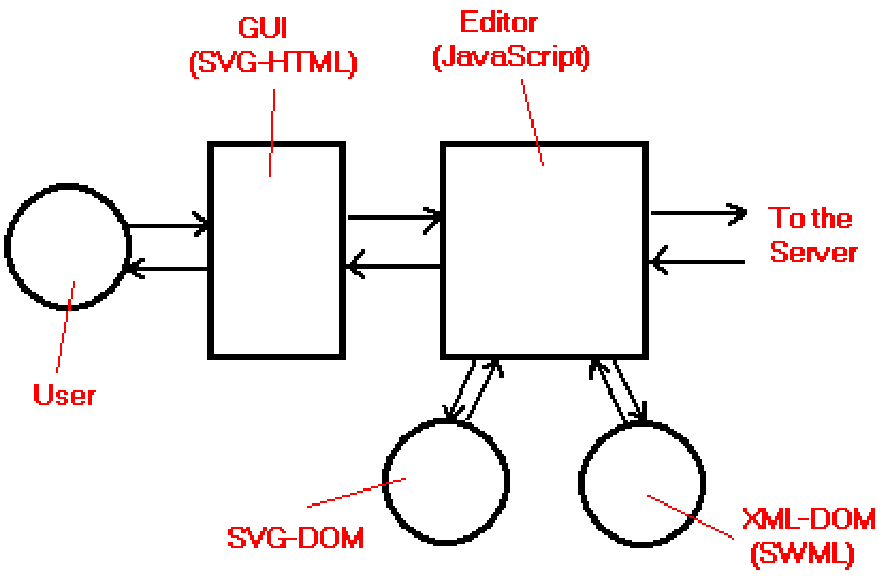


SW-WebMail: overview.

The critical component of the SW-WebMail system is, certainly, the SignWriting Email Editor. Its functionality should provide, at least, means for building full SignWriting signs (all symbols should be allowed).

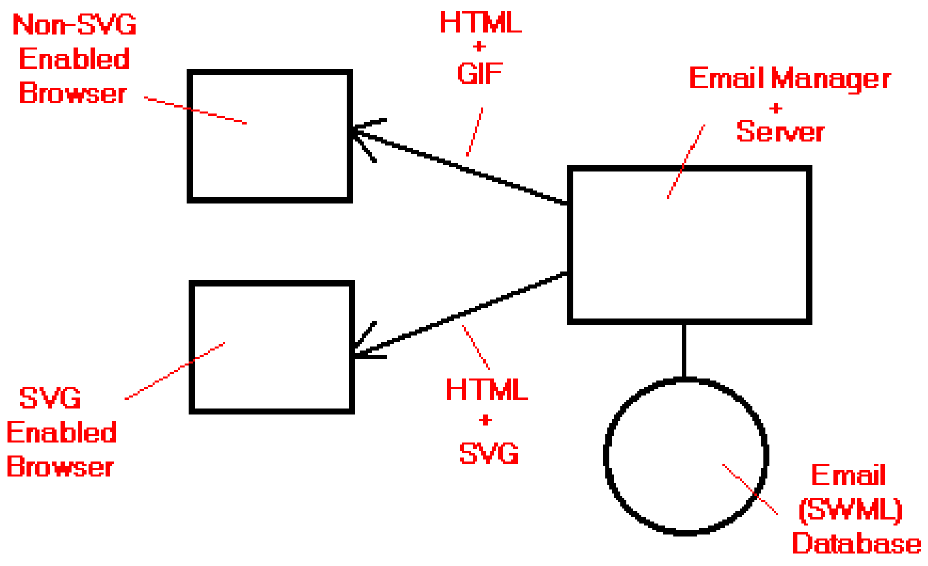
Important editing functions, such as copy, cut, paste and search of signs in the email body, and of symbols within signs, will be included in the editor on the tradeoff of the size the resulting script, since the idea is that the editing process will happen within a web page, downloaded from the server whenever the user wants to edit a new message.

Signs will be built graphically, by being represented in SVG in the browser. The editing process will operate, thus, on the SVG-DOM tree of the sign being built, as shown in the figure below, along with on the XML-DOM tree representing the SWML file that corresponds to the message.



The editor module of SW-WebMail.

On sending the message to the server, however, only the SWML file will be sent and stored in the server. This will allow the users to choose the way they want to render received messages: either by symbols represented by small GIF files, or by symbols represented in SVG - a choice to be made when reading the messages, in accordance with the capabilities of the browsers their are using (as shown in the figure below).



Selective retrieval of messages in SW-WebMail.

7. Conclusion, further work and call for cooperation

This paper presented work that is going on toward promoting the use of deaf sign languages, in written form, on the Web. The two main tools for that are the SignWriting system (a writing system for sign languages that has an intuitive appeal, due to its graphic-schematic principles) and SWML - SignWriting Markup Language (an XML-based format that is being proposed in the work for the representation of sign language documents and databases written in SignWriting).

To give practical evidence of the feasibility of using SWML as a representation format, an SWML-based web application is being developed: SW-WebMail, a webmail system for email messages written in SignWriting. SVG is being proposed as the preferred way of representing SignWriting symbols, and a standard rendering of SignWriting symbols as SVG symbols is being defined.

The SWML initiative - and the SignNet Project that supports it - encompass more than the work presented here, concerning the definition of SWML, its rendering in the SVG format, the research of suitable searching mechanisms for sign language texts, and the application of SWML to the SW-WebMail system.

It concerns also several other problems, such as the:

- development of new SignWriting software (new text editors, sign language databases systems and thesaurus, sign language dictionaries, sign language chat systems, etc.)
- development of means to help the interoperability of such SignWriting software
- development of techniques for the automatic spell cheking of sign language texts
- development of techniques for the automatic translation between sign languages and oral languages (and between sign languages and sign languages),
- the creation of methods for the animation of written signs.

Thus, due to the wideness of its scope, the SWML initiative and the SignNet Project are always willing to cooperate with every person or group interested in the subject of sign language processing. To make contact, look at the sites indicated in the paper.

Acknowledgments

First of all, the authors should say that they are grateful to Valerie Sutton. Without her long-standing friendship, continuous support, and illuminating views about the problems of sign language writing, this work would not have been possible. And, to Richard Gleaves, for his permanent will of helping, when help is needed. Second, the authors would like to thank CNPq, the Brazilian national research funding agency that, by giving financial support for the SignNet Project, made possible the initial development of SWML. FAPERGS, our regional research funding agency, gave complementary financial support for the SignNet Project. Last, but not least, the authors would like to thank all the deaf friends they have found during this work: their eagerness in learning how to write their sign language, and in disseminating this knowledge in the Deaf community, has been a main source of inspiration.

References

1. **Cleve, J. and Crouch, B.** *A Place of Their Own - creating the Deaf Community in America*. Gallaudet University Press, 1989.

2. **Kyle, J. and Woll, B**. *Sign Language - the study of deaf people and their language*. Cambridge University Press, 1995.

3. **Sutton, V.** *Lessons in SignWriting.* Reachable at http://www.signwriting.org/lessons/lessons.html

4. **Costa, A. C. R.** *The (current version of) SWML DTD.* Reachable at http://swml.ucpel.tche.br/