— sTeX Blue Note[*] —
Rethinking Modules and Semantic Macros in sTeX

Michael Kohlhase
Computer Science, Jacobs University.de

April 11, 2014

Abstract

In this note, we document the state of rethinking the sTeX infrastructure in terms of the SMGloM.

1 Introduction

We have been using sTeX as the encoding for the Semantic Multilingual Glossary of Mathematics (SMGloM; see [Gin+14]). The SMGloM data model has been taxing the representational capabilities of sTeX with respect to multilingual support and verbalization definitions; see [Koh14a], which we assume as background reading for this note.

2 Mixed Presentation/Content Markup

Currently, sTeX produces content markup in the OpenMath encoding. But often sTeX formulae often contain bits of presentational LaTeX, which LaTeXML has to convert into OpenMath heuristically, which often leads to non-optimal results. Therefore we want to rethink the representation of formulae, instead of insisting on homogeneous content markup in OpenMath, we switch to MathML allow mixed presentation/content MathML, which conforms much more closely to user input (preserving presentational bits) and postpones full semantification to later stages of processing. Let us make an example: consider the formula $(a + b)^n$ encoded das `\exp{a+b}n`, where we have a semantic macro `\exp` defined by `\symdef{exp}[2]{#1^{#2}}` in module `arith` Then we should create

```
<math>
  <apply>
    <csymbol cd="arith">exp</csymbol>
    <mrow><ci>a</ci><mo>+</mo><ci>b</ci></mrow>
    <ci>n</ci>
  </apply>
</math>
```

Note that MathML does indeed allow to freely mix content and presentation MathML, here we have an application produced by the semantic macro `\exp` applied to the presentational $a + b$, where a and b are "content identifiers".

A side effect of the switch to MathML is that complex variable names are much nicer in MathML: x_5 is just

```
<ci name="x5"><msub><mi>x</mi><mn>5</mn></msub></ci>
```

[*]Inspired by the "blue book" in Alan Bundy's group at the University of Edinburgh, sTeX blue notes, are documents used for fixing and discussing ϵ-baked ideas in projects by the sTeX group (see `http://github.com/KWARC/sTeX`). Unless specified otherwise, they are for project-internal discussions only. Please only distribute outside the sTeX group after consultation with the author.

3 sTeX Module Signatures

(monolingual) sTeX had the intuition that the symbol definitions (`\symdef` and `\symvariant`) are interspersed with the text and we generate sTeX module signatures (SMS `*.sms` files) from the sTeX files. The SMS duplicate "formal" information from the "narrative" sTeX files. In the SMGloM, we extend this idea by making the the SMS primary objects[1] that contain the language-independent part of the formal structure conveyed by the sTeX documents and there may be multiple narrative "language bindings" that are translations of each other – and as we do not want to duplicate the formal parts, those are inherited from the SMS rather than written down in the language binding itself. So instead of

Listing 1: Old-Style sTeX

```
\begin{module}[id=foo]
\symdef{bar}{BAR}
\begin{definition}[for=bar]
  A \defiii{big}{array}{raster} ($\bar$) is a\ldots
\end{definition}
\end{module}
```

we now advocate the divided style in Listing 2[1]. There the `modsig` environment works exactly like the old `module` environment, only that the `id` attribute has moved into the required argument – anonymous module signatures do not make sense. The `modnl` environment takes two arguments the first is the name of the module signature it provides language bindings for and the second the ISO 639 language specifier of the content language. We add the `primary` key to the optional argument of `modnl`, which can specify the primary language binding (the one the others translate from; and which serves as the reference in case of translation conflicts). EdN:1

Listing 2: New-Style sTeX

```
\begin{modsig}{foo}
\symdef{bar}{BAR}
\end{modsig}

\begin{modnl}[creators=miko,primary]{foo}{en}
\begin{definition}
  A \defiii[bar]{big}{array}{raster} ($\bar$) is a\ldots
\end{definition}
\end{modnl}
```

We retain the old `module` environment as an intermediate stage. It is still useful for monolingual texts. Note that for files with a module, we still have to extract `*.sms` files. It is not completely clear yet, how to adapt the workflows. We clearly need a lmh or editor command that transfers an old-style module into a new-style signature/binding combo to prepare it for multilingual treatment.

4 Verbalization Definitions

Currently, sTeX only supports notation definitions for symbols, but we also need verbalization definitions for flexiformal mathematics; see [Koh14a] for a description of the concept and background on their use and [Koh14b, section 5] for first ideas towards an sTeX encoding. We will extend the latter here.

The first thing to understand is that `\symdef` does two things in the LaTeXML workflow. It creates a `symbol` element and a `notation` element. In our new infrastructure, both go into the module signature. For verbalization definitions the situation is different. We want the `symbol` element in the module signature and the verbalization definition in the language bindings.

[1] Thanks to Deyan Ginev for realizing this.

[1] EdNote: MK: the names of the environments are still very much in the air. "modsig" I rather like, but "modnl" is terrible

For verbalization definitions in OMDoc we want to reuse the `notation` element, thus it seems normal to use the `\symdef` macro as well. In the situation of Listing 2, a verbalization definition for `bar` as the English phrase "*big array raster*" could be encoded as something like

```
\symvariant{bar}{lang:en}{\text{big array raster}}
```

Note that we already have a symbol `bar` generated by the `\symdef`, so we have to use the —— macro for this, if there were no prior `\symdef`, we would have to use a `\symdef`. To hide this choice from the user we hould probably have a wrapper macro

```
\verbdef[name=bar]{en}{big array raster}
```

But in most situations, an explicit language binding is unnecessary, since we have the definiendum markup. In the situation of Listing 2, we have a symbol `bar` generated by the `\symdef` and a definiendum for the symbol `bar` marked up by the `\defiii` macro – see [Koh13] for details on `\def*`. Note that the optional argument of `\defiii` is used to specify the symbol name, here `bar` here. We could let LaTeXML let generate the equivalent of a `verbdef` as above implicitly, freeing the user from writing down specifications twice.

But let us also look at a more interesting symbol: the "special linear group" already discussed in [Koh14b]. Here the sTeX verbalization definition would be

```
\verbdef[name=slgroup]{SLGroup}[2]{special linear group of order #1 over #2}
```

Here we have a problem with retrieving this from the definition without additional markup. A normal definition would have the form

```
\begin{definition}
  The \defiii[slgroup]{special}{linear}{group} \notatiendum{$\SLgroup{n}{F}$}
  of degree $n$ over a \trefi[field]{field} $F$ is ...
\end{definition}
```

In particular, the definiendum is discontiguous and usually only the "head" is explicitly emphasized by boldface font. In this situation, a "continuation markup might help – just exploring the syntax here:

```
\begin{definition}
  The \defiii[slgroup]{special}{linear}{group} \notatiendum{$\SLgroup{n}{F}$}
  \defc[slgroup]{of degree \defarg[1]{$n$}}
  \defc[slgroup]{over \defarg[2]{a \trefi[field]{field} $F$}} is ...
\end{definition}
```

Here the `\defc` macro continues the definiendum started with the `\defiii` – we specify which one with the symbol name in the optional argument and the embedded `\defarg` macro excapes out of that and marks its argument as an argument specifier. I am not sure that this is better than just adding the explicit verbalization definition above. But maybe the inline markup gives us more structure.

An alternative would be to have a long definiendum markup and use `\notatiendum` to escape out of it. Something like

```
\begin{definition}
  The \definiendum[slgroup]{special linear group \notatiendum{$\SLgroup{n}{F}$}
  of degree \defarg[1]{$n$} over \defarg[2]{a \trefi[field]{field} $F$}} is ...
\end{definition}
```

This implies less markup work. But do we lose structure here? If we have optional arguments (and here both are), we would like to associate "*of order*" with the first argument and "*over*" with the second. So maybe something like

```
\begin{definition}
  The \definiendum[slgroup]{\defhead{special linear group}
    \notatiendum{$\SLgroup{n}{F}$}
    \defarg[1,opt]{of degree \arg{$n$}}
    \defarg[2,opt]{over \arg{a \trefi[field]{field} $F$}}} is ...
\end{definition}
```

is more realistic. That would allow us to account for all the elision forms.

5 Conclusion

We have described a set of new functionalities for sTeX and specified some aspects of them. Now, they need to be implemented and tested.

References

[Gin+14] Deyan Ginev et al. "The SMGLoM Project and System". submitted to CICM 2014. 2014. URL: `http://kwarc.info/kohlhase/submit/cicm14-smglom-system.pdf`.

[Koh13] Michael Kohlhase. *`statements.sty`: Structural Markup for Mathematical Statements*. Tech. rep. 2013. URL: `https://github.com/KWARC/sTeX/raw/master/sty/statements/statements.pdf`.

[Koh14a] Michael Kohlhase. "A Data Model and Encoding for a Semantic, Multilingual Glossary of Mathematics". submitted to CICM 2014. 2014. URL: `http://kwarc.info/kohlhase/submit/cicm14-smglom-datamdl.pdf`.

[Koh14b] Michael Kohlhase. "A Data Model and Encoding for SMGloM". SMGloM Blue Note. 2014. URL: `http://gl.mathhub.info/smglom/smglom-doc/raw/master/source/blue/datamdl/note.pdf`.