structview.sty: Structures and Views in \$\LaTeX\$

Michael Kohlhase
FAU Erlangen-Nürnberg
FAU Erlangen-Nürnberg
http://kwarc.info/kohlhase

March 20, 2019

Abstract
The structview package is part of the \$\LaTeX\$ collection, a version of \TeX/\LaTeX\ that allows to markup \TeX/\LaTeX\ documents semantically without leaving the document format, essentially turning \TeX/\LaTeX\ into a document format for mathematical knowledge management (MKM).

This package supplies infrastructure for OMDoc structures and views: complex semantic relations between modules/theories.

Contents
1 Introduction 2
2 The User Interface 2
   2.1 Package Options 2
   2.2 Theory Morphisms 2
   2.3 Structures 3
   2.4 Views 3
3 Limitations & Extensions 5
4 The Implementation 5
   4.1 Package Options 5
   4.2 Theory Morphisms by Assignments 5
   4.3 Structures 5
   4.4 Views 6

*Version v1.4 (last revised 2019/03/20)
1 Introduction

Structures and views constitute ways of defining and relating theories in a theory graph that considerably extend the “object-oriented inheritance” constituted by the imports relation given by the \texttt{S*EX module} package.

Structures are like imports, only that they allow to define new theories via inheritance with renaming. Views relate pre-existing theories and model conceptual refinements, framing, and implementation relations, again via a mapping between the languages defined by the source and target theories; we call these mappings theory morphisms.

For details about theory morphisms we refer to \cite{RK13}, but hope to make the underlying concepts clear with examples.

![Figure 1: A Theory Graph with Structures and Views](image)

2 The User Interface

The main contributions of the \texttt{modules} package are the \texttt{module} environment, which allows for lexical scoping of semantic macros with inheritance and the \texttt{\symdef} macro for declaration of semantic macros that underly the \texttt{module} scoping.

2.1 Package Options

\texttt{mh} The \texttt{mh} option turns on MathHub support.

2.2 Theory Morphisms

A theory morphism is a mapping between the languages of its source and target theory. This can be described mathematically using all the structures in the

---

\footnote{\textsc{EdNote:} explain the contribution of structures and views to theory graphs and synchronize with Figure[1]}

---
\(\text{\LaTeX} \) distribution. However, in many situations, the language transformation of a morphism can be given in form of \textit{assignments} that map symbols of the source theory to expressions of the target theory.

There are three kinds assignments:\(^2\)

\begin{itemize}
  \item \texttt{symbol assignments} via \texttt{\vassign{\langle sym \rangle}{\langle exp \rangle}}, which maps a symbol \(\langle sym \rangle\) from source theory an expression \(\langle exp \rangle\) in the target theory.
  \item \texttt{function assignments} via \texttt{\fassign{\langle bvars \rangle}{\langle pat \rangle}{\langle exp \rangle}}, is a variant which maps a function symbol \(\langle sym \rangle\) by mapping a pattern expression \(\langle pat \rangle(\langle sym \rangle\text{applied to }\langle bvars \rangle)\) to an expression \(\langle exp \rangle\) in the target theory on bound variables \(\langle bvars \rangle\).
  \item \texttt{term assignments} via \texttt{\tassign{\langle sym \rangle}{\langle tname \rangle}}, another special case, where the value is the symbol with name \(\langle tname \rangle\) in the target theory.
\end{itemize}

Figure 1 shows a concrete example\(^3\)

The assignments above can be seen as abbreviations for a simple, formal definitions, which define a symbol of the source theory by an expression in the target theory.

### 2.3 Structures

\textbf{structure} Structures are specified by the \texttt{sstructure} environment:

\begin{verbatim}
\begin{sstructure}[(keys)]{\langle name \rangle}{\langle sthy \rangle}{\langle morph \rangle}\end{sstructure}
\end{verbatim}

gives the structure the name \(\langle name \rangle\), specifies the “source theory” via its identifier \(\langle sthy \rangle\), and the morphism \(\langle morph \rangle\). The \texttt{sstructure} environment takes the same keys as the \texttt{importmodule} macro, which it generalizes. The morphism \(\langle morph \rangle\) in the body of the \texttt{sstructure} environment specifies the morphism (see 2.2 above). In a structure, we take the target theory to be the current theory.

### 2.4 Views

A view is a mapping between modules, such that all model assumptions (axioms) of the source module are satisfied in the target module. For marking up views the \texttt{structview} package supplies the \texttt{view} environment; see Figure 2 for the \texttt{\LaTeX} markup of view \(v_1\) from Figure 1. The \texttt{view} environment takes one optional key/value argument followed by two mandatory ones: the names of the source and target modules. The \texttt{view} environment takes the following keys: \texttt{id} for a name, \texttt{title} and \texttt{display} for visual presentation, \texttt{loadfrom}, \texttt{loadto}, and \texttt{ext}\(^4\) for specifying the source files that supply the source and target modules, \texttt{creators}, \texttt{contributors}, \texttt{srccite} for document metadata, and \texttt{type}\(^5\).

\[^2\]\textit{EdNote}: \textit{MK}: we need better macros here.
\[^3\]\textit{EdNote}: adapt when we fully understand this, and the implementation works.
\[^1\]The old \texttt{importmodule} via environment is now deprecated.
\[^4\]\textit{EdNote}: \textit{MK}: we probably need toext and fromext here, but this never came up yet.
\[^5\]\textit{EdNote}: ????
Example 1: A Module for Rings with inheritance from monoids and commutative groups

```
\begin{module}[id=ring]
  \symdef{rbase}{R}
  \symdef{rtimes}[2]{\infix\cdot{#1}{#2}}
  \symdef{rone}{1}
  \begin{sstructure}{mul}{monoid}
    \tassign{magbase}{rbase}
    \fassign{a,b}{\magmaop{a}b}{\rtimes{a}b}
    \tassign{monunit}{rone}
  \end{sstructure}
  \symdef{rplus}[2]{\infix+{#1}{#2}}
  \symdef{rminus}[1]{\infix-{#1}{#2}}
  \begin{sstructure}{add}{cgroup}
    \fassign{a,b}{\magmaop{a}b}{\rplus{a}b}
    \tassign{monunit}{rzero}
    \tassign{cginvOp}{\rminus}
  \end{sstructure}
... 
\end{module}
```

Example 2: A view from monoids to integers

```
\begin{view}{monoid}{integers}
  \vassign{magbase}{base}
  \fassign{a,b}{\magmaop{a}b}{\inttimes{a,b}}
  \tassign{monunit}{\intzero}
  \begin{assertion}
    The Integers with addition form a monoid in the obvious way.
  \end{assertion}
\end{view}
```

Example 2: A view from monoids to integers
3 Limitations & Extensions

In this section we will discuss limitations and possible extensions of the modules package. Any contributions and extension ideas are welcome; please discuss ideas, requests, fixes, etc on the \TeX{} TRAC \TeX{}.

4 The Implementation

4.1 Package Options

We declare some switches which will modify the behavior according to the package options. Generally, an option \texttt{xxx} will just set the appropriate switches to true (otherwise they stay false). The options we are not using, we pass on to the \texttt{sref} package we require next.

\begin{verbatim}
\newif\if@structview@mh\@structview@mh@false
\DeclareOption{mh}{\@structview@mh@true}
\PassOptionsToPackage{\CurrentOption}{modules}
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{modules}}
\ProcessOptions
\end{verbatim}

The next measure is to ensure that the \texttt{sref} and \texttt{xcomment} packages are loaded (in the right version). For \LaTeX{}XML, we also initialize the package inclusions.

\begin{verbatim}
\if@structview@mh\RequirePackage{structview-mh}\fi
\RequirePackage{modules}
\end{verbatim}

4.2 Theory Morphisms by Assignments

\begin{verbatim}
\newrobustcmd\vassign[3][3]{\ifmod@show\ensuremath{#2\mapsto #3}, \fi}
\newrobustcmd\fassign[4][4]{\ifmod@show\ensuremath{#3(#2)\mapsto #4}, \fi}
\newrobustcmd\tassign[3][3]{\ifmod@show#2\mapsto\ #3, \fi}
\end{verbatim}

4.3 Structures

The \texttt{structure} environment just calls \texttt{\importmodule}, but to get around the group, we first define a local macro \texttt{\@@doit}, which does that and can be called with an \texttt{\aftergroup} to escape the environment grouping introduced by \texttt{structure}.

\begin{verbatim}
\newenvironment{\texttt{sstructure}}[3][3]{%\%
\gdef\@@doit{\importmodule[#1]{#3}}%
\ifmod@show\par\noindent importing module #3 via \@@doit\fi%}
\{\%\%
\aftergroup\@@doit\ifmod@show end import\fi%
\}%
\end{verbatim}

\textit{EdN:} probably get rid of the optional argument
4.4 Views

We first prepare the ground by defining the keys for the view environment.

\srefaddidkey{view}  
\addmetakey*{view}{title}  
\addmetakey{view}{display}  
\addmetakey{view}{loadfrom}  
\addmetakey{view}{loadto}  
\addmetakey{view}{creators}  
\addmetakey{view}{contributors}  
\addmetakey{view}{srccite}  
\addmetakey{view}{type}  
\addmetakey{view}{ext}  

Then we make a convenience macro for the view heading. This can be customized.

\ifdef{\thesection}{\newcounter{view}[section]}{\newcounter{view}}  
\newrobustcmd\view@heading[4]{%  
  \if@importing%  
  \else%  
    \stepcounter{view}  
    \edef\@display{#3}\edef\@title{#4}%  
    \noindent%  
    \ifx\@display\st@flow%  
    \else%  
    {\textbf{View} \thesection.\theview from \textsf{#1} to \textsf{#2}}%  
    \sref@label@id{View \thesection.\theview}%  
    \ifx\@title\@empty%  
    \else%  
    ($\@title$)%  
    \fi%  
  \fi% ifmod@show  
  \ignorespaces%  
}\view@heading  

The view environment relies on the @view environment (used also in the S\TeX\ module signatures) for module bookkeeping and adds presentation (a heading and a box) if the showmods option is set.

\newenvironment{view}[3][]% keys, from, to  
\metasetkeys{view}{#1}%

importmodulevia This is now deprecated, we give an error, but punt to structure.
The \texttt{@view} does the actual bookkeeping at the module level.

\begin{verbatim}
&view@heading[#2]{#3}{\view@display}{\view@title}{%}
\end{verbatim}

The \texttt{viewsketch} environment is deprecated, we give an error

\begin{verbatim}
&view@loadfrom}{#1}{\view@ext}{
&view@loadto}{#2}{\view@ext}{}
\end{verbatim}

The \texttt{obligation} element does not do anything yet on the latexml side.\footnote{EdNote: document above}

\begin{verbatim}
\if@importing%
\else Axiom #2 is proven by \sref{#3}\
\fi%
\end{verbatim}