Mario Wolczko
Dept. of Computer Science
The University
Manchester M13 9PL
U.K.
mario@cs.man.ac.uk, ...!uknet!man.cs!mario
21 May 1992
Version 626836.626836

Contents

1 Overview

This document describes a style option, `vdm`, for use with LaTeX. The purpose of `vdm` is to make the typesetting of VDM specifications easy. Other goals are:

To enable users of `vdm` to communicate their specifications to others, possibly in a variety of concrete syntaxes, without having to change their source files

To enable authors of `vdm` to concentrate on the abstract specifications, and ignore the detailed layout as much as possible. A side effect of this is that the effort required to improve layout is concentrated in one place, within the `vdm` macros.

This version of the `vdm` style option uses the BSI concrete syntax. Any document prepared using earlier versions is still accepted, but the way it is typeset will match more closely the BSI standard concrete syntax. There are also a few additional commands (summarised at the end). Note that this is *not* a complete style file for all of BSI VDM.)

But enough evangelising. Letjs get to the the real meat. This document is broken up into the following sections:

General points about using `vdm`

Typesetting formulas

How to typeset data types

How to typeset functions

How to typeset operations

How to typeset proofs

How to tailor/extend the system for your own application.

You should definitely read the first two sectionsnthen youjll know roughly what youjre in for, and whether you want to continue. The remaining sections can be read as and when you need them.

In keeping with the best traditions of TEX documentation, paragraphs that contain material that is not essential for novices, but vital if you want to parameterise or extend the system, are in smaller type, like this one.

Just to give a preliminary example, here is some output from `vdm`, and the corresponding input:

[Sorry. Ignored `\beginvdm ... \endvdm`]

```
\beginvdm
\beginfndecptrs,om \\
\signature
 \setofOop \x \mapofOopObject \to \mapofOopObject

\If ptrs = \emptyset
\Then om
\Else   \Let gone = \setp \in ptrs | RC(om(p)) = 1 \In
   \Let om' = gone \dsub om \In
   \Let om'' = om' \owr
 \mapp \mapsto \chgom'(p)RCRC\minus 1
    | p \in ptrs \diff gone \In
   dec(\Union\set\elemsBODY(om(p)) | p \in gone, om'')
\Fi
\endfn
```

```
\beginop[DESTROYPTR]
\args Obj, Ptr : Oop
\ext \Wr OM : \mapofOopBasic_Object
\pre ptr \in  \elemsBODY(om(obj))
\post om = ~om \owr \map obj \mapsto
 \chgom(obj)BODYBODY \diff \setptr
\endop
\endvdm
```

2 Using vdmnGeneral Points

To get at vdm, include vdm as a document style option, e.g.:

```
\documentstyle[12pt,vdm]report
```

To the best of my knowledge, the use of vdm does not conflict with any of the other document styles, except when something has been redefined. An attempt will be made to document all such redefinitions.

Once vdm has been included, you can then use the vdm environment. For example,

```
  \beginvdm
....
  \endvdm
```

All specification material should be placed within the vdm environment. The use of vdm only affects text within the vdm environment, except for the following global changes (which are only relevant when in math or display math mode):

1. The mathcodes of a?z and A?Z have been changed. In plain English, this means that when you type letters in math mode the inter-letter spacing is different This is would be useful if you are using LaTeX. This is because LaTeX math mode is usually tuned for single letter identifiers, as used by mathematicians for millenia. However, you and I both know that most meaningful identifiers have more than one letter in them, so vdm provides better spacing for them. As an example, if you type $identifier$, LaTeX would normally print *identifier*, whereas the use of vdm will yield *identifier*.

   you really want to use the inormalj inter-letter spacing, say .

2. Underscore gives you an underscore, and not a subscript. If you want a subscript use @, e.g., *x*@0 is typed x@0, or use TEXjs macro. An @ is still an @ when not in math mode. Occasionally you may find that an @ in math mode *doesnjt* give you a subscript (particularly when used with moving arguments). Should this happen, you are advised to use TEXjs macro, e.g., $x\sb0$.

   If you donjt use underscores much, and you want to use _ for subscripts, you can say (and to make it revert to its usual meaning in vdm).

3. - typesets a hyphen, and not a minus sign. VDM specifications usually contain a lot more

[Sorry. Ignored \beginvdm ... \endvdm]

than subtractions, so on the whole this alteration should save effort. If you really want to do a single subtraction sign, use `*`. If you find the default is inappropriate, you can revert to the original behaviour using ; `*` is the inverse. Example: `a-b \mathminus a-b` gives

[Sorry. Ignored `\beginvdm ... \endvdm`]

.

4. `|` gives you a

   [Sorry. Ignored `\beginvdm ... \endvdm`]

   and not a |. Do you see the difference? No? The former goes between things, e.g.,

   [Sorry. Ignored `\beginvdm ... \endvdm`]

   while the latter is a delimiter, e.g., |*x*|. In VDM, most people use the former more than the latter, so again this seems reasonable. If you really want a | (the second kind), say `|`.

5. In TEX and LATEX `~` has always been a tie (a space between words at which the line is never broken). Well in `vdm` it isnjt. `~x` will give you a

   [Sorry. Ignored `\beginvdm ... \endvdm`]

   . For long identifiers, such as

   [Sorry. Ignored `\beginvdm ... \endvdm`]

   say `~long`. *Note that this only applies in math mode; elsewhere a* `~` *is still a tie.*

6. In math mode, the double quote character `"` is actually a macro. Placing text between pairs of double quotes causes that text to be set in the normal text font. For example, `$x="a variable"$` gives you *x*="a variable".

   If you want to change the font used for text placed between quotes, redefine the command . By default it is defined to be (for the New Font Selection Scheme).

7. The following macros have been altered in a non-trivial way: *, * (see later).

When you typeset some VDM within the `vdm` environment, by default it is set in from the left margin by an amount equal to , the indentation at the beginning of each paragraph. If you want to change this, change the value of , e.g.:

```
\setlength\VDMindent0cm
```

will make your specs come out flush left. This document has been typeset with equal to 3*. Similarly, the right hand margin is controlled by a parameter called . By default it is also set to .

You can have a particular line spacing in force within the `vdm` environment. The spacing within a `vdm` environment is dictated by the command. Note that this is *not* a length, but a command. By default it expands to so that the line spacing is that of the surrounding text, whatever size that may be. To make it smaller, you may want to say

```
\renewcommand\VDMbaselineskip0.8\baselineskip
```

for example.