The eventB package*

Thai Son Hoang
ETH-Zurich
<htson at inf dot ethz dot ch>

July 3, 2012

Abstract

This class provides a template for typesetting Event-B models. It was developed at the Swiss Federal Institute of Technology Zurich (ETH-Zurich).

Contents

1 Introduction

This package was developed in order to ease the typesetting of Event-B models in LaTeX.

2 Usage

See `sample-eventB.tex` for an example of how to use the package.

3 Implementation

```
%%%%% BEGIN Package loading %%%%%
\RequirePackage{xspace}
\RequirePackage{xcolor}
%%%%% END Package loading %%%%%

%%%%% Keywords can be coloured by package option "color"
\newcommand{\Bkeyword}[1]{\ensuremath{\B@keyword{#1}}\xspace}
\newcommand{\Bidentifier}[1]{\B@identifier{#1}\xspace}
\newcommand{\Blabel}[2][]{\ensuremath{\B@label[#1]{#2}}\xspace}
\newcommand{\Bpo}[1]{\ensuremath{\B@po{#1}}\xspace}
```

*This document corresponds to eventB ?, dated ?.

```
\newcommand{\Bvspace}[1][2ex]{\\[#1]}
\newcommand{\Bhspace}[1][2em]{\hspace{#1}}
\newcommand{\Bsep}{\quad}
\DeclareOption{color}{
  \colorlet{Bkeywordcolor}{blue}
  \colorlet{Blabelcolor}{green!50!black}
  \colorlet{Bpocolor}{red}
  %%%%% Keywords are coloured.
  \renewcommand{\Bkeyword}[1]{\ensuremath{\textcolor{Bkeywordcolor}{\B@keyword{#1}}}\xspace}
  \renewcommand{\Blabel}[2][]{\ensuremath{\textcolor{Blabelcolor}{\B@label[#1]{#2}}}\xspace}
  \renewcommand{\Bpo}[1]{\ensuremath{\textcolor{Bpocolor}{\B@po{#1}}}\xspace}
}
\DeclareOption{colour}{
  \colorlet{Bkeywordcolor}{blue}
  \colorlet{Blabelcolor}{green!50!black}
  \colorlet{Bpocolor}{red}
  %%%%% Keywords are coloured.
  \renewcommand{\Bkeyword}[1]{\ensuremath{\textcolor{Bkeywordcolor}{\B@keyword{#1}}}\xspace}
  \renewcommand{\Blabel}[2][]{\ensuremath{\textcolor{Blabelcolor}{\B@label[#1]{#2}}}\xspace}
  \renewcommand{\Bpo}[1]{\ensuremath{\textcolor{Bpocolor}{\B@po{#1}}}\xspace}
}

\newcommand{\B@keyword}[1]{\ensuremath{\mathbf{#1}}\xspace}
\newcommand{\B@identifier}[1]{\ensuremath{\mathit{#1}}\xspace}
\newcommand{\B@label}[2][]{
  \def\is@thm{#1}
  \ifx\is@thm\@empty
  \ensuremath{\mathsf{#2}}\xspace
  \else
  \ensuremath{\mathit{#2}}\xspace
  \fi
}

%%%%% Bcode environment %%%%%
%%%%% the same as ``center'' %%%%%
\newenvironment{Bcode}[1][\normalsize]{\begin{center}#1}{\end{center}}
\newcommand{\Bdeclaration}[2]{
  \fbox{
    \ensuremath{
      \B@declaration{#1}{#2}
    }
  }
}

\newcommand{\Bsection}[3][]{
  \setlength{\B@oldfboxsep}{\fboxsep}
  \setlength{\fboxsep}{2ex}
  \fbox{
    \ensuremath{
      \B@section[#1]{#2}{#3}
    }
  }
  \setlength{\fboxsep}{\B@oldfboxsep}
}
```

```
\newcommand{\event}[7][]{
  \setlength{\B@oldfboxsep}{\fboxsep}
  \setlength{\fboxsep}{2ex}
  \fbox{
    \ensuremath{
      \B@event[#1]{#2}{#3}{#4}{#5}{#6}{#7}
    }
  }
  \setlength{\fboxsep}{\B@oldfboxsep}
}

\newcommand{\eventinline}[7][]{
  \setlength{\B@oldfboxsep}{\fboxsep}
  \setlength{\fboxsep}{2ex}
  \fbox{
    \ensuremath{
      \B@eventinline[#1]{#2}{#3}{#4}{#5}{#6}{#7}
    }
  }
  \setlength{\fboxsep}{\B@oldfboxsep}
}

\DeclareOption{compact}{
  \renewenvironment{Bcode}[1][\footnotesize]{\begin{center}#1}{\end{center}}
  \renewcommand{\Bdeclaration}[2]{
    \B@declaration{#1}{#2}
  }

  \renewcommand{\Bsection}[3][]{
    \B@section[#1]{#2}{#3}
  }

  \renewcommand{\event}[7][]{
    \B@event[#1]{#2}{#3}{#4}{#5}{#6}{#7}
  }

  \renewcommand{\eventinline}[7][]{
    \B@eventinline[#1]{#2}{#3}{#4}{#5}{#6}{#7}
  }

  \renewcommand{\Bvspace}[1][0ex]{\\[#1]}
  \renewcommand{\Bhspace}[1][0.5em]{\hspace{#1}}
  \renewcommand{\Bsep}{\ }
}
\DeclareOption{small}{
  \renewenvironment{Bcode}[1][\footnotesize]{\begin{center}#1}{\end{center}}
  \renewcommand{\Bvspace}[1][1ex]{\\[#1]}
  \renewcommand{\Bhspace}[1][1em]{\hspace{#1}}
  \renewcommand{\Bdeclaration}[2]{
    \B@declaration{#1}{#2}
  }

  \renewcommand{\Bsection}[3][]{
```

```
    \B@section[#1]{#2}{#3}
  }

  \renewcommand{\event}[7][]{
    \B@event[#1]{#2}{#3}{#4}{#5}{#6}{#7}
  }
  \renewcommand{\eventinline}[7][]{
    \B@eventinline[#1]{#2}{#3}{#4}{#5}{#6}{#7}
  }

  \renewcommand{\Bsep}{\ }

}
\DeclareOption{tiny}{
  \renewenvironment{Bcode}[1][\scriptsize]{\begin{center}#1}{\end{center}}
  \renewcommand{\Bvspace}[1][-0.5ex]{\\[#1]}
  \renewcommand{\Bhspace}[1][0.5em]{\hspace{#1}}
  \renewcommand{\Bdeclaration}[2]{
    \B@declaration{#1}{#2}
  }

  \renewcommand{\Bsection}[3][]{
    \B@section[#1]{#2}{#3}
  }

  \renewcommand{\event}[7][]{
    \B@event[#1]{#2}{#3}{#4}{#5}{#6}{#7}
  }
  \renewcommand{\eventinline}[7][]{
    \B@eventinline[#1]{#2}{#3}{#4}{#5}{#6}{#7}
  }

  \renewcommand{\Bsep}{\ }

}

\newcommand{\B@declaration}[2]{
  \begin{array}{l@{\Bsep}l}
    \Bkeyword{#1:} &  #2
  \end{array}
}
\newlength{\B@oldfboxsep}
\newcommand{\B@section}[3][]{
  \def\no@title{#1}
  \ifx\no@title\@empty
  \begin{array}{l}
    \Bkeyword{#2:} \\
    \begin{array}{l@{\Bsep}l}
      #3
    \end{array}
  \end{array}
  \else
  \begin{array}{l@{\Bsep}l}
    #3
```

```
  \end{array}
  \fi
}

\newcommand{\B@po}[1]{\ensuremath{\mathsf{#1}}\xspace}

%%%%% BEGIN Execution of options %%%%%
\ProcessOptions
%%%%% END Execution of options %%%%%

%%%%% (BEGIN) Macros for Pretty-Print Event-B Components %%%
\newcommand{\eventB}{Event-B\xspace}
\newcommand{\SKIP}{\textsc{skip}}

%%%%% Event-B Keywords %%%%%
\newcommand{\Bany}{\Bkeyword{any}}
\newcommand{\Bbegin}{\Bkeyword{begin}}
\newcommand{\Bend}{\Bkeyword{end}}
\newcommand{\Brefines}{\Bkeyword{refines}}
\newcommand{\Bstatus}{\Bkeyword{status}}
\newcommand{\Bthen}{\Bkeyword{then}}
\newcommand{\Bwhen}{\Bkeyword{when}}
\newcommand{\Bwhere}{\Bkeyword{where}}
\newcommand{\Bwith}{\Bkeyword{with}}

%%%%% Event-B internal elements %%%%%
\newcommand{\Bctx}[1]{\ensuremath{\mathbf{#1}}\xspace}
\newcommand{\Bset}[1]{\Bidentifier{#1}}
\newcommand{\Bcst}[1]{\Bidentifier{#1}}
\newcommand{\Baxm}[1]{\Blabel{#1}}
\newcommand{\Bthm}[1]{\Blabel[thm]{#1}}

\newcommand{\Bmch}[1]{\ensuremath{\mathbf{#1}}\xspace}
\newcommand{\Bvrb}[1]{\Bidentifier{#1}}
\newcommand{\Binv}[1]{\Blabel{#1}}
\newcommand{\Bevt}[1]{\Blabel{#1}}
\newcommand{\Bpar}[1]{\Bidentifier{#1}}
\newcommand{\Bact}[1]{\Blabel{#1}}
\newcommand{\Bgrd}[1]{\Blabel{#1}}
\newcommand{\Bbap}[1]{\hbox{\sl\bfseries #1}}
%%%%
%%%%% Creating Event-B elements macros %%%%%

%%%%% Create a new B macro
%%%%% Arguments:
%%%%% 1. The macro string, (OPTIONAL) if empty then the expanded string will be used.
%%%%% 2. The expanded string
%%%%% 3. The mark-up macros, e.g. \Bvrb
%%%%% Usage:
%%%%% - \B@newmacro[aaa]{a\_a\_a}{\Bvrb} will create a new macro \aaa
%%%%% which will be expanded to be \Bvrb{a\_a\_a}
%%%%% - \B@newmacro{aaa}{\Bvrb} will create a new macro \aaa
%%%%% which will be expanded to be \Bvrb{aaa}
```

```
%%%%%
\newcommand{\B@newmacro}[3][]{
  \def\input@macro{#1}
  \ifx\input@macro\@empty
  \expandafter\def\csname #2\endcsname{#3{#2}}
  \else
  \expandafter\def\csname #1\endcsname{#3{#2}}
  \fi
}

%%%%% Create a new context macro
%%%%% Arguments:
%%%%% 1. The macro string (OPTIONAL)
%%%%% 2. The expanded string
%%%%% Usage:
%%%%% - \newBctx[aaa]{a\_a\_a} will create a new macro \aaa
%%%%% which will be expanded to be \Bctx{a\_a\_a}.
%%%%% - \newBctx{aaa} will create a new macro \aaa which will be
%%%%% expanded to be \Bctx{aaa}.
%%%%%
\newcommand{\newBctx}[2][]{%
  \B@newmacro[#1]{#2}{\Bctx}
}

%%%%% Create a new carrier set macro
%%%%% Arguments:
%%%%% 1. The macro string (OPTIONAL)
%%%%% 2. The expanded string
%%%%% Usage:
%%%%% - \newBset[aaa]{a\_a\_a} will create a new macro \aaa
%%%%% which will be expanded to be \Bset{a\_a\_a}.
%%%%% - \newBset{aaa} will create a new macro \aaa which will be
%%%%% expanded to be \Bset{aaa}.
%%%%%
\newcommand{\newBset}[2][]{%
  \B@newmacro[#1]{#2}{\Bset}
}

%%%%% Create a new constant macro
%%%%% Arguments:
%%%%% 1. The macro string (OPTIONAL)
%%%%% 2. The expanded string
%%%%% Usage:
%%%%% - \newBcst[aaa]{a\_a\_a} will create a new macro \aaa
%%%%% which will be expanded to be \Bcst{a\_a\_a}.
%%%%% - \newBcst{aaa} will create a new macro \aaa which will be
%%%%% expanded to be \Bcst{aaa}.
%%%%%
\newcommand{\newBcst}[2][]{%
  \B@newmacro[#1]{#2}{\Bcst}
}

%%%%% Create a new axiom macro
```

```
%%%%% Arguments:
%%%%% 1. The macro string (OPTIONAL)
%%%%% 2. The expanded string
%%%%% Usage:
%%%%% - \newBaxm[aaa]{a\_a\_a} will create a new macro \aaa
%%%%% which will be expanded to be \Baxm{a\_a\_a}.
%%%%% - \newBaxm{aaa} will create a new macro \aaa which will be
%%%%% expanded to be \Baxm{aaa}.
%%%%%
\newcommand{\newBaxm}[2][]{%
  \B@newmacro[#1]{#2}{\Baxm}
}


%%%%% Create a new theorem macro
%%%%% Arguments:
%%%%% 1. The macro string (OPTIONAL)
%%%%% 2. The expanded string
%%%%% Usage:
%%%%% - \newBthm[aaa]{a\_a\_a} will create a new macro \aaa
%%%%% which will be expanded to be \Bthm{a\_a\_a}.
%%%%% - \newBthm{aaa} will create a new macro \aaa which will be
%%%%% expanded to be \Bthm{aaa}.
%%%%%
\newcommand{\newBthm}[2][]{%
  \B@newmacro[#1]{#2}{\Bthm}
}


%%%%% Create a new machine macro
%%%%% Arguments:
%%%%% 1. The macro string (OPTIONAL)
%%%%% 2. The expanded string
%%%%% Usage:
%%%%% - \newBmch[aaa]{a\_a\_a} will create a new macro \aaa
%%%%% which will be expanded to be \Bmch{a\_a\_a}.
%%%%% - \newBmch{aaa} will create a new macro \aaa which will be
%%%%% expanded to be \Bmch{aaa}.
%%%%%
\newcommand{\newBmch}[2][]{%
  \B@newmacro[#1]{#2}{\Bmch}
}


%%%%% Create a new variable macro
%%%%% Arguments:
%%%%% 1. The macro string (OPTIONAL)
%%%%% 2. The expanded string
%%%%% Usage:
%%%%% - \newBvrb[aaa]{a\_a\_a} will create a new macro \aaa
%%%%% which will be expanded to be \Bvrb{a\_a\_a}.
%%%%% - \newBvrb{aaa} will create a new macro \aaa which will be
%%%%% expanded to be \Bvrb{aaa}.
%%%%%
\newcommand{\newBvrb}[2][]{%
  \B@newmacro[#1]{#2}{\Bvrb}
```

```
}

%%%% Create a new invariant macro
%%%% Arguments:
%%%% 1. The macro string (OPTIONAL)
%%%% 2. The expanded string
%%%% Usage:
%%%% - \newBinv[aaa]{a\_a\_a} will create a new macro \aaa
%%%% which will be expanded to be \Binv{a\_a\_a}.
%%%% - \newBinv{aaa} will create a new macro \aaa which will be
%%%% expanded to be \Binv{aaa}.
%%%%
\newcommand{\newBinv}[2][]{%
  \B@newmacro[#1]{#2}{\Binv}
}

%%%% Create a new event macro
%%%% Arguments:
%%%% 1. The macro string (OPTIONAL)
%%%% 2. The expanded string
%%%% Usage:
%%%% - \newBevt[aaa]{a\_a\_a} will create a new macro \aaa
%%%% which will be expanded to be \Bevt{a\_a\_a}.
%%%% - \newBevt{aaa} will create a new macro \aaa which will be
%%%% expanded to be \Bevt{aaa}.
%%%%
\newcommand{\newBevt}[2][]{%
  \B@newmacro[#1]{#2}{\Bevt}
}

%%%% Create a new parameter macro
%%%% Arguments:
%%%% 1. The macro string (OPTIONAL)
%%%% 2. The expanded string
%%%% Usage:
%%%% - \newBpar[aaa]{a\_a\_a} will create a new macro \aaa
%%%% which will be expanded to be \Bpar{a\_a\_a}.
%%%% - \newBpar{aaa} will create a new macro \aaa which will be
%%%% expanded to be \Bpar{aaa}.
%%%%
\newcommand{\newBpar}[2][]{%
  \B@newmacro[#1]{#2}{\Bpar}
}

%%%% Create a new guard macro
%%%% Arguments:
%%%% 1. The macro string (OPTIONAL)
%%%% 2. The expanded string
%%%% Usage:
%%%% - \newBgrd[aaa]{a\_a\_a} will create a new macro \aaa
%%%% which will be expanded to be \Bgrd{a\_a\_a}.
%%%% - \newBgrd{aaa} will create a new macro \aaa which will be
%%%% expanded to be \Bgrd{aaa}.
```

```
%%%%%
\newcommand{\newBgrd}[2][]{%
  \B@newmacro[#1]{#2}{\Bgrd}
}

%%%%% Create a new action macro
%%%%% Arguments:
%%%%% 1. The macro string (OPTIONAL)
%%%%% 2. The expanded string
%%%%% Usage:
%%%%% - \newBact[aaa]{a\_a\_a} will create a new macro \aaa
%%%%% which will be expanded to be \Bact{a\_a\_a}.
%%%%% - \newBact{aaa} will create a new macro \aaa which will be
%%%%% expanded to be \Bact{aaa}.
%%%%%
\newcommand{\newBact}[2][]{%
  \B@newmacro[#1]{#2}{\Bact}
}

%%%%% Pretty print carrier sets
%%%%% Arguments:
%%%%% 1. (Comma-separated) list of carrier sets.
%%%%%
%%%%% Usage: \carriersets{S, T}
\newcommand{\carriersets}[1]{
  \Bdeclaration{sets}{#1}
}

%%%%% Pretty print constants
%%%%% Arguments:
%%%%% 1. (Comma-separated) list of constants.
%%%%%
%%%%% Usage: \constants{m, n}
\newcommand{\constants}[1]{
  \Bdeclaration{constants}{#1}
}

%%%%% Pretty print axioms
%%%%% Arguments:
%%%%% 1. (Newline(\\)-separated) list of axioms.
%%%%%
%%%%% Usage: \axioms{\Baxm{axm0\_1}: & x \in \nat \\
%%%%%                \Baxm{axm0\_2}: & y \in \nat \\[2ex]}
\newcommand{\axioms}[2][]{
  \Bsection[#1]{axioms}{#2}
}

%%%%% Pretty print variables
%%%%% Arguments:
%%%%% 1. (Comma-separated) list of variables.
%%%%%
%%%%% Usage: \variables{x, y}
\newcommand{\variables}[1]{
```

```
  \Bdeclaration{variables}{#1}
}

%%%% Pretty print invariants
%%%% Arguments:
%%%% 1. (Newline(\\)-separated) list of invariants.
%%%%
%%%% Usage: \invariants{\Binv{inv0\_1:} & x \in \nat \\
%%%%                    \Binv{inv0\_2:} & y \in \nat \\[2ex]}
\newcommand{\invariants}[2][]{
  \Bsection[#1]{invariants}{#2}
}

%%%% Pretty print variant
%%%% Arguments:
%%%% 1. The variant
%%%%
%%%% Usage: \variant{V}
\newcommand{\variant}[1]{
  \Bdeclaration{variant}{#1}
}

%%%% Pretty print an general Event-B event
%%%% Arguments:
%%%% 1. (Optional) convergence status.
%%%% 2. Name of the event.
%%%% 3. Name of the abstract event.
%%%% 4. (Comma-separated) list of parameters.
%%%% 5. (Newline(\\)-separated) list of guards.
%%%% 6. (Newline(\\)-separated) list of witness predicates.
%%%% 7. (Newline(\\)-separated) list of assignments.
%%%%
%%%% Usage: \B@event[conv]{conc}{abs}{x,y}{G1(x,y)\\G2(x,y)}{W1\\W2}{S1(v,x,y)\\S2(w,x,y)}
%%%%       will produce the following
%%%%
%%%%       conc
%%%%       refines abs
%%%%       status conv
%%%%       any x, y where
%%%%         G1(x, y)
%%%%         G2(x, y)
%%%%       with
%%%%         W1
%%%%         W2
%%%%       then
%%%%         S1(v, x, y)
%%%%         S2(w, x, y)
%%%%       end
%%%%
%%%% Special case:
%%%% - Empty abstract event --> refines clause is omitted.
%%%% - Empty convergence status --> status clause is omitted.
%%%% - Empty witness --> with clause is omitted.
```

```
%%%%% - Empty parameters, empty guards --> begin ... end
%%%%% - Empty parameters --> when ... then ... end
%%%%% - Empty actions --> \SKIP
\newcommand{\B@event}[7][]{
  \def\evt@sts{#1}
  \def\evt@name{#2}
  \def\evt@absevts{#3}
  \def\evt@pars{#4}
  \def\evt@grds{#5}
  \def\evt@wits{#6}
  \def\evt@acts{#7}
  %% Pretty-print convergence status
  \ifx\evt@sts\@empty
  \def\pretty@sts{}
  \else
  \def\pretty@sts{\Bsep\Bstatus \Bsep \evt@sts \\}
  \fi
  % Pretty-print abstract events
  \ifx\evt@absevts\@empty
  \def\pretty@absevts{}
  \else
  \def\pretty@absevts{\Bsep\Brefines \Bsep \evt@absevts \\}
  \fi
  % Pretty-print parameters
  \ifx\evt@pars\@empty
  \def\pretty@pars{}
  \else
  \def\pretty@pars{\Bsep\Bany \Bsep \evt@pars \Bsep \Bwhere \\}
  \fi
  % Pretty-print guards
  \ifx\evt@grds\@empty
  \def\pretty@grds{}
  \else
  \def\evt@grds@tmp{
    \begin{array}{@{\Bsep\Bsep}l@{\Bsep}l}
      \evt@grds
    \end{array}\\
  }
  \ifx\evt@pars\@empty
  \def\pretty@grds{
    \Bsep \Bwhen \\
    \evt@grds@tmp
  }
  \else
  \def\pretty@grds{\evt@grds@tmp}
  \fi
  \fi
  % Pretty-print witnesses
  \ifx\evt@wits\@empty
  \def\pretty@wits{}
  \else
  \def\pretty@wits{
    \Bsep\Bwith\\
    \begin{array}{@{\Bsep\Bsep}ll}
```

```
      \evt@wits
    \end{array}\\
  }
  \fi
  % Pretty-print actions
  \ifx\evt@acts\@empty
  \def\evt@acts{\SKIP}
  \else
  \fi
  \def\evt@acts@tmp{
    \begin{array}{@{\Bsep\Bsep}l@{\Bsep}l}
      \evt@acts
    \end{array}\\
  }
  \def\evt@acts@keyword{\Bsep\Bthen \\}
  \ifx\evt@pars\@empty
  \ifx\evt@grds\@empty
  \def\evt@acts@keyword{\Bsep\Bbegin \\}
  \else
  \fi
  \else
  \fi
  \def\pretty@acts{
    \evt@acts@keyword
    \evt@acts@tmp
  }
  % Really do it now
  \begin{array}{l}
    \Bevt{\evt@name} \\
    \pretty@sts
    \pretty@absevts
    \pretty@pars
    \pretty@grds
    \pretty@wits
    \pretty@acts
    \Bsep\Bend
  \end{array}
}


%%%% Pretty print an general Event-B event
%%%% Arguments:
%%%% 1. (Optional) convergence status.
%%%% 2. Name of the event.
%%%% 3. Name of the abstract event.
%%%% 4. (Comma-separated) list of parameters.
%%%% 5. (Newline(\\)-separated) list of guards.
%%%% 6. (Newline(\\)-separated) list of witness predicates.
%%%% 7. (Newline(\\)-separated) list of assignments.
%%%%
%%%% Usage: \B@event[conv]{conc}{abs}{x,y}{G1(x,y)\\G2(x,y)}{W1\\W2}{S1(v,x,y)\\S2(w,x,y)}
%%%%       will produce the following
%%%%
%%%%       conc
%%%%       refines abs
```

```
%%%%       status conv
%%%%       any x, y where
%%%%         G1(x, y)
%%%%         G2(x, y)
%%%%       with
%%%%         W1
%%%%         W2
%%%%       then
%%%%         S1(v, x, y)
%%%%         S2(w, x, y)
%%%%       end
%%%%
%%%% Special case:
%%%% - Empty abstract event --> refines clause is omitted.
%%%% - Empty convergence status --> status clause is omitted.
%%%% - Empty witness --> with clause is omitted.
%%%% - Empty parameters, empty guards --> begin ... end
%%%% - Empty parameters --> when ... then ... end
%%%% - Empty actions --> \SKIP
\newcommand{\B@eventinline}[7][]{
  \def\evt@sts{#1}
  \def\evt@name{#2}
  \def\evt@absevts{#3}
  \def\evt@pars{#4}
  \def\evt@grds{#5}
  \def\evt@wits{#6}
  \def\evt@acts{#7}
  %% Ignore convergence status
  \def\pretty@sts{}
  % Pretty-print abstract events
  \ifx\evt@absevts\@empty
  \def\pretty@absevts{}
  \else
  \def\pretty@absevts{\Brefines~\evt@absevts~}
  \fi
  % Pretty-print parameters
  \ifx\evt@pars\@empty
  \def\pretty@pars{}
  \else
  \def\pretty@pars{\Bany~\evt@pars~\Bwhere~}
  \fi
  % Pretty-print guards
  \ifx\evt@grds\@empty
  \def\pretty@grds{}
  \else
  \def\evt@grds@tmp{
      \evt@grds
  }
  \ifx\evt@pars\@empty
  \def\pretty@grds{
    \Bwhen~
    \evt@grds@tmp~
  }
  \else
```

```
  \def\pretty@grds{\evt@grds@tmp~}
  \fi
  \fi
  % Pretty-print witnesses
  \ifx\evt@wits\@empty
  \def\pretty@wits{}
  \else
  \def\pretty@wits{
    \Bwith~
    \evt@wits~
  }
  \fi
  % Pretty-print actions
  \ifx\evt@acts\@empty
  \def\evt@acts{\SKIP}
  \else
  \fi
  \def\evt@acts@tmp{
    \evt@acts
  }
  \def\evt@acts@keyword{\Bthen}
  \ifx\evt@pars\@empty
  \ifx\evt@grds\@empty
  \def\evt@acts@keyword{\Bbegin}
  \else
  \fi
  \else
  \fi
  \def\pretty@acts{
    \evt@acts@keyword~
    \evt@acts@tmp~
  }
  % Really do it now
  \begin{array}{l}
    \Bevt{\evt@name}~\widehat{=}~
    \pretty@sts
    \pretty@absevts
    \pretty@pars
    \pretty@grds
    \pretty@wits
    \pretty@acts
    \Bend
  \end{array}
}


%%%%% INITIALISATION label
\newBevt{init}

%%%%% Pretty print the initialisation: no ``refines'' clause. no parameters, no
%%%%% guards
%%%%% Arguments:
%%%%% 1. (Newline(\\)-separated) list of assignments.
%%%%%
%%%%% Usage: \init{S1(v,x,y)\\S2(w,x,y)}
```

```
%%%%%       will produce the following
%%%%%
%%%%%       init
%%%%%       begin
%%%%%         S1(v, x, y)
%%%%%         S2(w, x, y)
%%%%%       end
%%%%%
\newcommand{\initialisation}[1]{
  \event{\init}{}{}{}{}{#1}
}


%%%% Theorem Proof Obligation
%%%% Print the theorem proof obligation, given the theorem label.
%%%% Arguments:
%%%% 1. Theorem label
%%%%
%%%% Usage:
%%%% - \thmpo{thm} will produce "thm/THM"
\newcommand{\thmpo}[1]{\Bthm{#1}/\Bpo{THM}}


%%%% Axiom Well-definedness Proof Obligation
%%%% Print the axiom well-definedness proof obligation, given the
%%%% axiom label.
%%%% Arguments:
%%%% 1. Axiom label
%%%%
%%%% Usage:
%%%% - \axmwdpo{axm} will produce "axm/WD"
\newcommand{\axmwdpo}[1]{\Baxm{#1}/\Bpo{WD}}


%%%% Invariant Proof Obligation
%%%% Print the invariant proof obligation, given the event name and
%%%% invariant label
%%%% Arguments:
%%%% 1. Event name
%%%% 2. Invariant label
%%%%
%%%% Usage:
%%%% - \invpo{evt}{inv} will produce "evt/inv/INV"
\newcommand{\invpo}[2]{\Bevt{#1}/\Binv{#2}/\Bpo{INV}}


%%%% Theorem (in guard) Proof Obligation
%%%% Print the simulation proof obligation, given the event name and
%%%% the theorem (in guard) label.
%%%% Arguments:
%%%% 1. Event name
%%%% 2. Theorem (in guard) label
%%%%
%%%% Usage:
%%%% - \grdthmpo{evt}{thm} will produce "evt/thm/THM"
\newcommand{\grdthmpo}[2]{\Bevt{#1}/\Bthm{#2}/\Bpo{THM}}
```

```
%%%% Feasibility Proof Obligation
%%%% Print the feasibility proof obligation, given the event name and
%%%% the action label
%%%% Arguments:
%%%% 1. Event name
%%%% 2. Action label
%%%%
%%%% Usage:
%%%% - \fispo{evt}{act} will produce "evt/act/FIS"
\newcommand{\fispo}[2]{\Bevt{#1}/\Bact{#2}/\Bpo{FIS}}


%%%% Variant finiteness Proof Obligation
%%%% Print the Variant finiteness proof obligation
%%%% Arguments: No arguments
%%%%
%%%% Usage:
%%%% - \finpo will produce "FIN"
\newcommand{\finpo}{\Bpo{FIN}}


%%%% Variant Proof Obligation
%%%% Print the guard strengthen proof obligation, given the event name
%%%% Arguments:
%%%% 1. Event name
%%%%
%%%% Usage:
%%%% - \grdpo{evt} will produce "evt/VAR"
\newcommand{\varpo}[1]{\Bevt{#1}/\Bpo{VAR}}


%%%% Simulation Proof Obligation
%%%% Print the simulation proof obligation, given the event name and
%%%% the action label.
%%%% Arguments:
%%%% 1. Event name
%%%% 2. Action label
%%%%
%%%% Usage:
%%%% - \simpo{evt}{act} will produce "evt/act/SIM"
\newcommand{\simpo}[2]{\Bevt{#1}/\Bact{#2}/\Bpo{SIM}}


%%%% Guard Strengthen Proof Obligation
%%%% Print the guard strengthen proof obligation, given the event
%%%% name and the guard label
%%%% Arguments:
%%%% 1. (Abstract) Event name
%%%% 2. (Abstract) Guard label
%%%%
%%%% Usage:
%%%% - \grdpo{evt}{grd} will produce "evt/grd/GRD"
\newcommand{\grdpo}[2]{\Bevt{#1}/\Bgrd{#2}/\Bpo{GRD}}


%%%% Variant Natural Number Proof Obligation
%%%% Print the Variant Natural Number proof obligation, given the event name
```

```
%%%% Arguments:
%%%% 1. Event name
%%%%
%%%% Usage:
%%%% - \natpo{evt} will produce "evt/NAT"
\newcommand{\natpo}[1]{\Bevt{#1}/\Bpo{NAT}}
```

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the definition; numbers in roman refer to the pages where the entry is used.

Change History

v1.0

General: Initial version 1

v1.0.1

General: Ensure that the keywords, labels are in math-mode 1