ISO TC 184/SC4/ EC ___ N 47b **Date:** 2002/09/04

Supersedes ISO TC 184/SC4/EC ___ N 47a

ISO/WD 10303-3456.2
Product data representation and exchange: documentation methods: LaTeX package files for ISO 10303: User manual

ABSTRACT:

This document describes and illustrates the LaTeX macros for typesetting ISO 10303. The International Organisation for Standardisation (ISO) has specified editorial directives for all international standards published by them. The LaTeX macros described here were developed to meet additional editorial directives for ISO 10303.

KEYWORDS: LaTeX, document preparation, typesetting ISO standards

COMMENTS TO READER:

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation. This document serves two purposes. Firstly, it provides a description of the current LaTeX macros for ISO 10303. Secondly, the source can be used as an example of using the LaTeX commands. Although the document is written as though it were a standard, it is not, and is not intended to become, a standard.

Project Leader: Peter R. Wilson
Address: Boeing Commercial Airplane
PO Box 3701
MS 2R-97
Seattle, WA 98124-2207
USA

Telephone: +1 (206) 544-0589
Facsimile: +1 (206) 544-5889
E-mail: peter.r.wilson@boeing.com

Project Editor: Peter R. Wilson
Address: Boeing Commercial Airplane
PO Box 3701
MS 2R-97
Seattle, WA 98124-2207
USA

Telephone: +1 (206) 544-0589
Facsimile: +1 (206) 544-5889
E-mail: peter.r.wilson@boeing.com

revision 8, 1/02 (PRW)

Contents

Tables

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75% of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 10303–3456 was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration*, Subcommittee SC4, *Industrial data*.

This International Standard is organized as a series of parts, each published separately. The structure of this International Standard is described in ISO 10303–1.

Each part of this International Standard is a member of one of the following series: description methods, implementation methods, conformance testing methodology and framework, integrated generic resources, integrated application resources, application protocols, abstract test suites, application interpreted constructs, and application modules. This part is a member of the documentation methods series.

A complete list of parts of ISO 10303 is available from the Internet:

`<http://www.nist.gov/sc4/editing/step/titles/>`

Annexes A, B and C are a normative part of this part of ISO 10303. Annexes D, E and F are for information only.

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation of product information and for the exchange of product data. The objective is to provide a neutral mechanism capable of describing products throughout their life cycle. This mechanism is suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases, and as a basis for archiving.

This part of ISO 10303 specifies the LaTeX facilities specifically designed for use in preparing the various parts of this standard.

Major subdivisions of this part of ISO 10303 are:

— the step package facility;

— the ir package facility;

— the ap package facility;

— the aic package facility;

— the ats package facility.

This part of ISO 10303 is intended to be used in conjunction with *LaTeX for ISO standards: User manual* which is based in part upon material in the ISO/IEC Directives, Part 2 (*Rules for the structure and drafting of International Standards, Fourth edition*). The LaTeX facilities described here are based as well upon the specifications given in ISO TC184/SC4 N1217n (*SC4 Supplementary directives — Rules for the structure and drafting of SC4 standards for industrial data*).

Overview

This document describes a set of LaTeX macro files for use within ISO 10303, commonly called STEP (STandard for the Exchange of Product model data). The electronic source of this document also provides an example of the use of these files.

The current set of macro files have been developed by Peter Wilson (`peter.r.wilson@boeing.com`) from a macro file developed by Kent Reed (NIST) for LaTeX v2.09. In turn, this was a revision of files originally created by Phil Spiby (CADDETC), based on earlier work by Phil Kennicott (GE).[1)]

NOTE It is important to remember that these macro files are only compatible with LaTeX2e.

Documents produced with the LaTeX files have been twice reviewed by the ISO Editorial Board in Geneva for conformance to their typographical requirements. The first review was of a set of Draft International Standard documents. This review resulted in some changes to the style files. The second review was of a set of twelve International Standard documents (ISO 10303:1994). Likewise, this review led to changes in the style files to bring the documents into conformance.

With the issuance of the first STEP release, the opportunity was taken to provide a new baseline release of the package files. In particular, one STEP specific package file is available for all STEP

[1)]In mid 1994 LaTeX was upgraded from version 2.09 to what is called LaTeX2e. The files described in this document are only applicable to LaTeX2e (support for LaTeX v2.09 was dropped in September 1997).

parts, while others contain only commands relevant to the documentation of particular series of parts. The range of package files may be extended in the future to cater for documentation specific to all STEP parts.

The 1997 baseline release was also designed to cater for the fact that a major update of LaTeX to LaTeX2e took place during 1994. LaTeX2e is the only officially supported version of LaTeX.

Because ISO standard documents have a very structured layout, the isov2 class and the package files described here have been designed to reflect the logical document structure to a much greater extent than the 'standard' LaTeX files.

With ISO's move toward accepting documents in PDF and HTML, the advent of second editions of some of the STEP parts, and a new edition of the STEP Supplementary Directives, a 2002 baseline release has been developed and is documented here.

Industrial automation systems and integration — Product data representation and exchange — Part 3456 : Documentation methods: LaTeX package files for ISO 10303: User manual

1 Scope

This part of ISO 10303 describes a set of LaTeX facilities for typesetting documents according to the ISO/IEC Directives Part 2, together with the Supplementary Directives for drafting and presentation of ISO 10303.

The following are within the scope of this part of ISO 10303:

— use of LaTeX for preparing ISO 10303 documents.

The following are outside the scope of this part of ISO 10303:

— use of LaTeX for preparing ISO standard documents in general;

— use of LaTeX in general;

— use of other document preparation systems.

IMPORTANT: The preparation of this document has been partly funded by the US Government and is not subject to copyright. Any copyright notices within the document are for illustrative purposes only.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO 10303. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO 10303 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards, Fourth edition.*

ISO TC 184/SC4 N1217:2001(E), *SC4 Supplementary directives — Rules for the structure and drafting of SC4 standards for industrial data.*

ISO 10303-1:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles.*

ISO 10303-11:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual.*

ISO/TR 10303-12:1997, *Industrial automation systems and integration — Product data representation and exchange — Part 12: Description method: The EXPRESS-I language reference manual.*

ISO/IEC 8824-1:1998, *Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation.*

P. R. WILSON:—[2], *LaTeX for ISO standards: User manual.*

3 Terms, definitions, and abbreviations

3.1 Terms defined in ISO 10303-1

This part of ISO 10303 makes use of the following terms defined in ISO 10303-1:

— application protocol (AP)

— integrated resource

3.2 Other definitions

For the purposes of this part of ISO 10303, the following definitions apply.

3.2.1
boilerplate
Text whose wording is fixed and which has been agreed to be present in a specific type of document.

3.2.2
style file
A set of LaTeX macros assembled into a single file with an extension .sty.

3.2.3
package file
A style file for use with LaTeX2e.

3.2.4
facility
A generic term for a set of LaTeX macros assembled for a common purpose. The macros may be defined in either a style file or a package file.

3.3 Abbreviations

For the purposes of this part of ISO 10303, the following abbreviations apply.

[2]To be published.

AIC Application Interpreted Construct

AM Application Module

AP Application Protocol

DIS Draft International Standard

IS International Standard

ISOD ISO/IEC Directives, Part 2

SD Supplementary Directives — *SC4 Supplementary directives — Rules for the structure and drafting of SC4 standards for industrial data*

IS-REVIEW The ISO Editorial Board review (September 1994) of twelve IS documents for conformance to ISO typographical and layout requirements.

4 Conformance requirements

The facility files shall not be modified in any manner.

If there is a need to modify any of the macro definitions then this shall be done using the LaTeX `\renewcommand` and/or the `\renewenvironment` commands. These shall be placed in a new .sty file (or files) which shall be called in within the preamble of the document being typeset.

There shall be no author specified `\label{...}` commands where the first two characters of the label are `;s` (semicolon and 's'); the creation of labels starting with these characters is reserved to the maintainer of the facility files.

NOTE For conformance to the isov2 class, author specified labels starting with the characters `;i` (semicolon and 'i') are prohibited.

5 Fundamental concepts and assumptions

It is assumed that the reader of this document is familiar with the LaTeX document preparation system and in particular with the isov2 class and associated facilities described in *LaTeX for ISO standards: User manual*.

NOTE 1 Reference [1] describes the LaTeX system.

The reader is also assumed to be familiar with the ISO/IEC Directives Part 2 (ISOD) and the SC4 Supplementary directives for the structure and drafting of SC4 standards (SD).

If there are any discrepancies between the layout and wording of this document and the requirements of the ISOD or the SD, then the requirements in those documents shall be followed for ISO 10303 standard documents.

The packages described herein have been designed to be used with the isov2 document class. It is highly unlikely that the packages will perform at all with any other LaTeX document class.

Table 1 – File versions current at publication time

Facility	File	Version
step	stepv13.sty	v1.3.2
ir	irv12.sty	v1.2
ap	apv12.sty	v1.2
aic	aicv1.sty	v1.0
ats	atsv11.sty	v1.1

Because of many revisions over the years to the packages described herein, a naming convention has been adopted for the package files. The naming convention is that the primary name of the file is suffixed by v#, where # is the primary version number of the file in question. All file primary names have been limited to a maximum of eight characters.

NOTE 2 Table 1 shows the versions of the files that were current at the time of publication.

NOTE 3 This document is not, and is never intended to become, a standard, although it has been laid out in a similar, but not necessarily identical, manner.

6 The step package facility

The step package facility provides commands and environments applicable to all the ISO 10303 series of documents.

6.1 Preamble commands

Certain commands shall be put in the preamble of any document.

The command `\partno{`<*number*>`}` is used to specify the Part number of the ISO 10303 standard (e.g., `\partno{3456}`).

The command `\series{`<*series title*>`}` is used to specify the name of the ISO 10303 series of which the Part is a member (e.g., `\series{application modules}`).

The command `\doctitle{`<*informal title*>`}` is used to specify the title to be used on the cover sheet. For example:
`\doctitle{LaTeX package files for ISO 10303: User manual}`

The command `\ballotcycle{`<*number*>`}` is used to specify the ballot cycle number for the document (e.g., `\ballotcycle{2}`).

The command `\haspatentstrue` shall be put in the preamble when the document includes identified patented material; otherwise the command `\haspatentsfalse` may, but need not, be used instead.

The `\extrahead` macro, from the isov2 class, shall be defined to be the document number (e.g., `\renewcommand{\extrahead}{47a}`).

NOTE The commands `\standard`, `\yearofedition` and `\languageofedition` from the isov2 class must also be put in the preamble.

6.2 Cover page

The command `\STEPcover{`<*commands*>`}` produces a cover page for a STEP document. The complete list of commands is shown below.

— `\wg{`<*working group*>`}` the working group or other committee producing the document e.g., WG 5

— `\docnumber{`<*number*>`}` the number of the document e.g., 156

— `\docdate{`<*date*>`}` date of publication e.g., 1993/07/03

— `\oldwg{`<*working group*>`}` superseded working group e.g., WG 1

— `\olddocnumber{`<*number*>`}` number of previous document e.g., 107

— `\abstract{`<*text*>`}` an abstract of the document

— `\keywords{`<*text*>`}` for listing relevant keywords

— `\owner{`<*text*>`}` name of the project leader

— `\address{`<*text*>`}` address of the project leader

— `\telephone{`<*number*>`}` the project leader's telephone number

— `\fax{`<*number*>`}` the project leader's fax number

— `\email{`<*text*>`}` Email address of the project leader

— `\altowner{`<*text*>`}` name of the editor of the document

— `\altaddress{`<*text*>`}` the editor's address

— `\alttelephone{`<*number*>`}` the editor's telephone number

— `\altfax{`<*number*>`}` the editor's fax number

— `\altemail{`<*text*>`}` the editor's Email address

— `\comread{`<*text*>`}` comments to the reader

Use only those commands within `\STEPcover` that are relevant to the purposes at hand. The order of the commands within `\STEPcover` is immaterial.

EXAMPLE 1 The commands used to produce the cover sheet for one version of this document were:

```
\STEPcover{
\wg{EC}
\docnumber{41}
\oldwg{EC}
\olddocnumber{35}
\docdate{1994/08/19}
\abstract{This document describes the \latex{} style files created for ISO~10303.
          It also describes the program GenIndex which provides some
          capabilities to assist in the creation of indexes for \latex{}
          documents in general.}
\keywords{\latex, Style file, GenIndex, Index}
\owner{Peter R Wilson}
\address{NIST\newline
         Bldg. 220, Room A127 \newline
         Gaithersburg, MD 20899 \newline
         USA }
\telephone{+1 (301) 975-2976}
\email{\texttt{pwilson@cme.nist.gov}}
\altowner{Tony Day}
\altaddress{Sikorsky Aircraft}
\comread{This document serves two purposes. Firstly, it provides a description
         of the current \latex{} style file for ISO 10303. Secondly, the source
         can be used as an example of using the \latex{} commands.} % end comread
} % end of STEPcover
```

Note the use of the `\newline` command instead of `\\` in the argument of the `\address` command to indicate a new line. The `\newline` is needed to ensure satisfactory conversion to HTML.

The macro `\draftctr` generates boilerplate that may be used in the 'Comments to Reader' section of a cover page.

EXAMPLE 2 The LaTeX source `\draftctr This document \ldots` prints:

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation. This document ...

6.3 Heading commands

The commands described in this subclause specify various 'standard' clause headings.

6.3.1 The Foreword commands

The `\Foreword` command specifies that a table of contents, list of figures and a list of tables be produced. Page numbering is roman style and the table of contents starts on page iii. A new unnumbered clause entitled Foreword is started containing both ISO required boilerplate and boilerplate text specific to ISO 10303.

Any text may be written after the `\Foreword` command. The Foreword clause is ended by the `\endForeword{`<*norm annexes*>`}{`<*inf annexes*>`}` command. This command takes two parameters.

a) <*norm annexes*> A phrase that starts the sentence '<*norm annexes*> a normative part of this part ...'. If there are no normative annexes, then use an empty argument (i.e., `{}` with no spaces between the braces).

b) <*inf annexes*> A phrase that starts the sentence '<*inf annexes*> for information only.'. If there are no informative annexes, then use an empty argument.

The `\endForeword` command produces some additional boilerplate text specifically for ISO 10303.

EXAMPLE 1 The LaTeX source for the Foreword for this document is:

```
\Foreword
\fwdshortlist
\endForeword
{Annexes A, B and C are}  % normative annexes
{Annexes D, E and F are} % informative annexes
```

The `\fwdshortlist` command produces boilerplate text for inclusion in the foreword referencing the STEP parts and series.

EXAMPLE 2 In this document, the command `\fwdshortlist` prints:

This International Standard is organized as a series of parts, each published separately. The structure of this International Standard is described in ISO 10303–1.

Each part of this International Standard is a member of one of the following series: description methods, implementation methods, conformance testing methodology and framework, integrated generic resources, integrated application resources, application protocols, abstract test suites, application interpreted constructs, and application modules. This part is a member of the documentation methods series.

The `\steptrid` command produces boilerplate text for inclusion in the foreword describing the creators of a STEP Technical Report.

EXAMPLE 3 The LaTeX command `\steptrid` in this document prints:

ISO/TR 10303–3456, which is a Technical Report of type 2, was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration,* Subcommittee SC4, *Industrial data.*

6.3.2 The Introduction environment

The `\begin{Introduction}` environment starts a new unnumbered clause entitled Introduction and adds some boilerplate text specifically for ISO 10303.

EXAMPLE 1 The following LaTeX source was used to specify the Introduction to this document.

```
\begin{Introduction}

    This part of ISO 10303 specifies the \latex{} facilities
specifically designed for use in preparing the various parts of
this standard.

\begin{majorsublist}
\item the \file{step} package facility;
\item the \file{ir} package facility;
\item the \file{ap} package facility;
\item the \file{aic} package facility;
\item the \file{atc} package facility.
\end{majorsublist}

    This part of ISO 10303 is intended to be used ...

\sclause*{Overview}

    This document describes a set of \latex{} files for use
within ISO~10303 ...

\end{Introduction}
```

6.3.3 The stepparttitle command

The `\stepparttitle{`<*part title*>`}` command produces the title for an ISO 10303 part, where <*part title*> is the title of the part.

EXAMPLE The title for this document was produced using:

```
\stepparttitle{Documentation methods:
               LaTeX package files for ISO 10303: User manual}
```

6.3.4 Other headings

Most of these commands take no parameters. They start document clauses with particular titles. The commands that take no parameters are listed in Table 2. Some of these headings commands have predefined labels, which are also listed in the table.

NOTE In the tables, C = clause, SC = subclause, SSC = subsubclause, NA = normative annex, IA = informative annex.

The commands listed in Table 3 are equivalent to the general sectioning commands, but are intended to indicate the start of a particular documentation element. These commands take

Table 2 – STEP package parameterless heading commands

Command	Clause	Default text	Label
`\partidefhead`	SC	Terms defined in ISO 10303-1	
`\otherdefhead`	SC	Other definitions	
`\introsubhead`	SC	Introduction	
`\fcandasubhead`	SC	Fundamental concepts and assumptions	
`\shortnamehead`	NA	Short names of entities	`;ssne`
`\picshead`	NA	Protocol Implementation Conformance Statement (PICS) proforma	`;spics`
`\objreghead`	NA	Information object registration	`;sior`
`\docidhead`	SC	Document identification	
`\schemaidhead`	SC	Schema identification	
`\expresshead`	IA	EXPRESS listing	
`\listingshead`	IA	Computer interpretable listings	`;scil`
`\expressghead`	IA	EXPRESS-G diagrams	`;seg`
`\techdischead`	IA	Technical discussions	`;std`
`\exampleshead`	IA	Examples	`;sex`

Table 3 – STEP package parameterized heading commands

Command	Clause	Parameterized title
`\refdefhead`	SC	Terms defined in <*ISO ref*>
`\schemahead`	C	<*schema name*>
`\singletypehead`	SC	<*schema name*> type definition: <*type name*>
`\typehead`	SC	<*schema name*> type definitions
`\atypehead`	SSC	<*type name*>
`\singleentityhead`	SC	<*schema name*> entity definition: <*entity name*>
`\entityhead`	SC	<*schema name*> entity definitions <*group*>
`\anentityhead`	SSC	<*entity name*>
`\singlerulehead`	SC	<*schema name*> rule definition: <*rule name*>
`\rulehead`	SC	<*schema name*> rule definitions
`\arulehead`	SSC	<*rule name*>
`\singlefunctionhead`	SC	<*schema name*> function definition: <*function name*>
`\functionhead`	SC	<*schema name*> function definitions
`\afunctionhead`	SSC	<*function name*>
`\aschemaidhead`	SSC	<*schema name*> identification

either one or two parameters. The parameters are denoted in the column headed ‘Parameterized title’.

6.4 Miscellaneous commands

The following commands provide some printing options for commonly occurring situations. The `\nexp{}` command is intended to be used for printing EXPRESS entity names etc.

— The command `\B{abc}` prints **abc**

— The command `\E{abc}` prints *abc*

— The command `\Express` prints EXPRESS

- The command \ExpressG prints EXPRESS-G
- The command \ExpressI prints EXPRESS-I
- The command \ExpressX prints EXPRESS-X
- The command \BG{<*mathsymbol*>} prints <*mathsymbol*> in bold font.
- The command \HASH prints #
- The command \LT prints <
- The command \LE prints <=
- The command \NE prints <>
- The command \INE prints :<>:
- The command \GE prints >=
- The command \GT prints >
- The command \CAT prints ||
- The command \QUES prints ?
- The command \IEQ prints :=:
- The command \INEQ prints :<>:
- The command \nexp{an_entity} prints **an_entity**
- The command \xword{ExpResS_KeyworD} prints EXPRESS_KEYWORD

The command \ix{<*word or phrase*>} both prints its parameter and also makes an index entry out of it.

The command \mnote{<*Marginal note text*>} prints its parameter as a marginal note. Remember, though, that marginal notes are only printed when the isov2 class draft option is used. Marginal notes are not allowed by ISO.

6.4.1 Standard reference commands

Many parts of STEP use the same normative or informative references. The most common of these are provided via commands. The currently available commands are listed in Table 4.

The naming convention used for references to parts of ISO 10303 is to end the command name with the number of the part expressed in lower case Roman numerals. Should further references to parts of ISO 10303 be added later, the same naming convention will be used.

EXAMPLE 1 The normative references in this document were input as:

Table 4 – Commands for common references to standards

Standard	Command
ISO/IEC 8824-1	`\nrefasni`
ISO 10303-1	`\nrefparti`
ISO 10303-11	`\nrefpartxi`
ISO 10303-12	`\nrefpartxii`
ISO 10303-21	`\nrefpartxxi`
ISO 10303-22	`\nrefpartxxii`
ISO 10303-31	`\nrefpartxxxi`
ISO 10303-32	`\nrefpartxxxii`
ISO 10303-41	`\nrefpartxli`
ISO 10303-42	`\nrefpartxlii`
ISO 10303-43	`\nrefpartxliii`

```
\begin{nreferences}
\isref{ISO/IEC Directives, Part 2}{Rules for the structure and drafting
       International Standards, Fourth edition.}
\isref{...}
\nrefparti
\nrefpartxi
\nrefpartxii
\nrefasni
\disref{P. R. WILSON:---}{LaTeX for ISO standards: User manual.}
\end{nreferences}
```

NOTE For the commands providing references to STEP parts, the part number is denoted by lowercase Roman numerals. Should further reference commands be provided for other STEP parts, then the same naming scheme will be used.

Some informative bibliographic reference commands are also provided.

The command `\bibidefo` produces the reference entry to the IDEF0 document and `\brefidefo` can be used for citing the reference in the body of the document.

The commands `\bibidefix` and `\bibieeedefix` produce the reference entry to the original FIPS version of IDEF1X and the IEEE version of IDEF1X respectively. The command `\brefidefix` can be used for citing an IDEF1X reference in the body of the document.

IDEF0 and IDEF1X are references [11] and [12] in the bibliography.

EXAMPLE 2 Part of the bibliography for this document looks like:

```
\begin{references}
...
\reference{BRYAN, M.,}{SGML --- An Author's Guide to the Standard Generalized
           Markup Language,}{Addison-Wesley Publishing Co., 1988. }\label{bryan}
\bibidefo
\bibieeeidefix
\reference{RESSLER, S.,}{The National PDES Testbed Mail Server User's Guide,}
           {NSTIR 4508, National Institute of Standards and Technology,
           Gaithersburg, MD 20899. January 1991.} \label{ressler}
...
\end{references}
```


EXAMPLE 3 The source for one of the sentences above was:

```
IDEF0 and IDEF1X are references \brefidefo{} and \brefidefix{} in the bibliography.
```

6.5 Commands for documenting EXPRESS code

The Supplementary Directives specify the layout of the documentation of EXPRESS code. The following commands are intended to serve two purposes:

a) To provide environments for the documentation of entity attributes, etc.;

b) To provide begin and end tags around all the EXPRESS code documentation.

This latter purpose is to provide an enabling capability for the automatic extraction of portions of the documentation of an EXPRESS model so that they could be placed into another document. For example, tools could be developed that would automatically extract pieces of resource model documentation and place them into an AP document.

NOTE This document uses the `hyphenat` package which enables automatic hyphenation of 'words' containing the underscore character command (`\_`). Such words would normally have to be coded as `long\_\-word` to ensure potential hyphenation at the position of the underscore. When using the `hyphenat` package it is an error to put the `\-` discretionary hyphen command after the underscore command as this then stops further hyphenation.

6.5.1 Environments ecode, eicode and excode

The `ecode` environment is for tagging EXPRESS code. It prints the appropriate title and sets up the relevant fonts.

EXAMPLE The following LaTeX source code:

```
\begin{ecode}\ixent{an\_entity}
\begin{verbatm}  % read verbatm as verbatim
*)
ENTITY an_entity;
  attr : REAL;
END_ENTITY;
(*
\end{verbatm}   % read verbatm as verbatim
\end{ecode}
```

produces:

EXPRESS specification:

```
*)
ENTITY an_entity;
  attr : REAL;
END_ENTITY;
(*
```

Similarly, the `eicode` and `excode` environments are for tagging EXPRESS-I and EXPRESS-X code and setting up the relevant titles and fonts.

6.5.2 Environment attrlist

The `attrlist` environment produces the heading for attribute definitions and sets up a `description` list.

EXAMPLE The following LaTeX source code:

```
\begin{attrlist}
\item[attr\_1] The \ldots
\item[attr\_2] This \ldots
\end{attrlist}
```

produces:

Attribute definitions:

attr_1: The ...

attr_2: This ...

6.5.3 Environment fproplist

The `fproplist` environment is similar to `attrlist` except that it is for formal propositions.

EXAMPLE The following LaTeX source code:

```
\begin{fproplist}
\item[un\_1] The value of \ldots\ shall be unique.
\item[gt\_0] The value of \ldots\ shall be greater than zero.
\end{fproplist}
```

produces:

Formal propositions:

un_1: The value of ... shall be unique.

gt_0: The value of ... shall be greater than zero.

6.5.4 Other listing environments

The environments `iproplist`, `enumlist`, and `arglist` are similar to `attrlist`. Respectively they are environments for informal propositions, enumerated items, and argument definitions.

6.5.5 Indexing

The command `\ixent{`<*entity*>`}` generates an index entry for the entity <*entity*>.

There are similar macros, each of which takes the name of the declaration as its argument, for indexing the other EXPRESS declarations: `\ixenum` for enumeration, `\ixfun` for function, `\ixproc` for procedure, `\ixrule` for rule, `\ixsc` for subtype_constraint, `\ixschema` for schema, `\ixselect` for select, and `\ixtype` for type.

6.5.6 Documentation tagging

Several environments are defined to tag the general documentation of EXPRESS code.

The environment `\begin{espec}{<name>}` may be used to enclose, and give a name to, a complete specification block for an EXPRESS entity. There are analogous environments — `fspec`, `rspec`, `sspec`, and `tspec` — for functions, rules, schemas and types respectively.

The `dtext` environment may be used to anonymously enclose descriptive text.

EXAMPLE 1 Here is the suggested tagged documentation style for part of an EXPRESS model.

```
%\ssclause{committee\_def}
\begin{espec}{committee_def}
\begin{dtext}
    A committee is composed of an odd number of people.
Each committee also has a name.
    The ideal size of a committee is less than three.

\begin{anote} Figures and tables may also be placed here. \end{anote} % end note
\end{dtext}
\begin{ecode}\ixent{committee\_def}
\begin{verbatm} % read verbatm as verbatim
*)
ENTITY committee_def;
  title   : name;
  members : SET [1:?] OF person;
DERIVE
  ideal : BOOLEAN := SIZEOF(members) = 1;
UNIQUE
  un1 : title;
WHERE
  odd_members : ODD(SIZEOF(members));
END_ENTITY;
(*
\end{verbatm}   % read verbatm as verbatim
\end{ecode}
\begin{attrlist}
\item[title] The name of the committee.
\item[members] The people who form the committee.
\item[ideal] TRUE if there is only one person
             on the committee.
             That is, if the committee is the ideal size.
\end{attrlist}
\begin{fproplist}
\item[un1] The \nexp{title} of the committee shall be unique.
\item[odd\_members] There shall be an odd number of people
                    on the committee.
\end{fproplist}
\begin{iproplist}
\item[chair] The members of a committee shall appoint one of
             their number as
             chair of the committee.
\end{iproplist}
\end{espec}
```

EXAMPLE 2 The code in example 1 produces the following result:

A committee is composed of an odd number of people. Each committee also has a name. The ideal size of a committee is less than three.

NOTE Figures and tables may also be placed here.

EXPRESS specification:

```
*)
ENTITY committee_def;
  title   : name;
  members : SET [1:?] OF person;
DERIVE
  ideal : BOOLEAN := SIZEOF(members) = 1;
UNIQUE
  un1 : title;
WHERE
  odd_members : ODD(SIZEOF(members));
END_ENTITY;
(*
```

Attribute definitions:

title: The name of the committee.

members: The people who form the committee.

ideal: TRUE if there is only one person on the committee. That is, if the committee is the ideal size.

Formal propositions:

un1: The **title** of the committee shall be unique.

odd_members: There shall be an odd number of people on the committee.

Informal propositions:

chair: The members of a committee shall appoint one of their number as chair of the committee.

6.6 Commands producing boilerplate text

The following commands produce boilerplate text as specified by the Supplementary Directives.

NOTE In the examples, the parameters of those commands that take them have been specified in *this font style* so their effects can be seen in the resulting printed text.

6.6.1 Definition of EXPRESS-G

The `\expressgdef` prints the boilerplate for where the definition of EXPRESS-G can be found.

EXAMPLE The command `\expressgdef` prints:

EXPRESS-G is defined in annex D of ISO 10303-11

6.6.2 Major subdivision listing

The majorsublist environment prints the boilerplate for the heading of a listing of major subdivisions of the standard and starts an itemized list. An illustration of its use is given in example 1 on page 8.

The heading text is produced by the \majorsubname command.

EXAMPLE The command \majorsubname command prints:

Major subdivisions of this part of ISO 10303 are:

6.6.3 Schema introduction

The command \schemahead{<*schema name*>} prints the heading for a schema clause.

The command \schemaintro{<*schema name*>} produces the boilerplate for the introduction to an EXPRESS schema clause.

EXAMPLE The command \schemaintro{\nexp{this_schema}} prints:

The following EXPRESS declaration begins the **this_schema** and identifies the necessary external references.

6.6.4 Short names of entities

The command \shortnamehead prints the heading for the short names annex.

The command \shortnames produces the boilerplate for the introduction to the annex listing short names.

EXAMPLE The command \shortnames prints:

Table A.1 provides the short names of entities specified in this part of ISO 10303. Requirements on the use of short names are found in the implementation methods included in ISO 10303.

6.6.5 Registration commands

The command \objreghead prints the heading for the information object registration annex.

The command \docidhead prints the heading for the document identification subclause.

The command \docreg{<*version no*>} produces the boilerplate for document registration. The command takes one parameter: <*version no*> which is the version number.[3)]

EXAMPLE 1 The command \docreg{1} prints:

To provide for unambiguous identification of an information object in an open system, the object identifier

{ iso standard 10303 part(3456) version(*1*) }

[3)] The SD say that the version number should be 1 for a first edition IS. The version number is incremented by one for each corrigenda, amendment or new edition.

is assigned to this part of ISO 10303. The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

The command \schemaidhead prints the heading for the schema identification subclause. The command \aschemaidhead{<*schema name*>} prints the heading for a particular schema identification subsubclause.

The command \schemareg{<*version no*>}{<*p2*>}{<*p3*>}{<*p4*>}{<*p5*>}{<*p6*>} produces the boilerplate concerning schema registration. The command takes six parameters.

a) <*version no*> The version number;

b) <*p2*> The name of an EXPRESS schema (with underscores);

c) <*p3*> The number of the schema object (typically 1);

d) <*p4*> The name of the schema, with hyphens replacing any underscores in the name;

e) <*p5*> The number identifying the schema;

f) <*p6*> The clause or annex in which the schema is defined.

EXAMPLE 2 The command
\schemareg{1}{a_schema}{3}{a-schema}{5}{clause 6} prints:

To provide for unambiguous identification of the schema-name in an open information system, the object identifier

{ iso standard 10303 part(3456) version(*1*) schema(*3*) *a-schema*(*5*) }

is assigned to the **a_schema** schema (see *clause 6*). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

6.6.6 Computer interpretable listings

The command \listingshead prints the heading for the computer interpretable listings annex.

The command \expurls{<*short*>}{<*express*>} produces the boilerplate for the introduction to the annex listing short names and EXPRESS, where <*short*> is the URL for the short names and <*express*> is the URL for the EXPRESS.

EXAMPLE The command \expurls{http:/www.short/}{http://www.express/} prints:

This annex references a listing of the EXPRESS entity data type names and corresponding short names as specified in this part of ISO 10303. It also references a listing of each EXPRESS schema specified in this part of ISO 10303, without comments or other explanatory text. These listings are available in computer-interpretable form and can be found at the following URLs:

Short names: <http://www.short/>
EXPRESS: <http://www.express/>

If there is difficulty accessing these sites contact ISO Central Secretariat or contact the ISO TC 184/SC4 Secretariat directly at: sc4sec@cme.nist.gov.

NOTE The information provided in computer-interpretable form at the above URLs is informative. The information that is contained in the body of this part of ISO 10303 is normative.

7 The ir package facility

The ir package provides commands and environments specifically for the ISO 10303 Integrated Resources series of documents.

Use of this package requires the use of the step package.

7.1 Boilerplate commands

The ir package modifies the `\fwdshortlist` command to produce extra IR-specific boilerplate.

The following commands produce boilerplate text as specified by the SD.

7.1.1 Integrated resource EXPRESS-G

The command `\expressghead` prints the heading for the EXPRESS-G diagrams annex.

The command `\irexpressg` produces the boilerplate for the introduction to the integrated resource EXPRESS-G annex.

EXAMPLE The command `\irexpressg` prints:

The diagrams in this annex correspond to the EXPRESS schemas specified in this part of ISO 10303. The diagrams use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined in annex D of ISO 10303-11.

8 The ap package facility

The `ap` package provides commands and environments specifically for the ISO 10303 Application Protocol series of documents.

Use of this package requires the use of the `step` package.

8.1 Preamble commands

Certain commands shall be put in the preamble of an AP document.

The command `\aptitle{`<*title of AP*>`}` shall be put into the preamble. The parameter shall be of such a form that it will read naturally in a sentence of the form: '... for the <*title of AP*> application protocol.'.

EXAMPLE For the purposes of later examples, the command `\aptitle{implicit drawing}` has been put in the preamble of this document.

If the AP makes use of one or more AICs, then the command `\aicinaptrue` shall be put in the document preamble.

If a mapping specification is used instead of a mapping table, the command `\mapspectrue` shall be put in the preamble. If mapping templates are used then `\maptemplatetrue` shall also be put in the preamble.

If IDEF1X is used instead of EXPRESS-G as the graphical form for the ARM, then `\idefixtrue` shall be put in the preamble.

8.2 Heading commands

These commands start document clauses with particular titles. The commands that take no parameters are listed in Table 5. Some of these commands have predefined labels, which are also listed in the table.

The commands listed in Table 6 take parameters.

8.3 Boilerplate commands

The following commands produce boilerplate text as specified by the SD.

NOTE In the examples, the parameters of those commands that take them have been specified in *this font style* so their effects can be seen in the resulting printed text.

8.3.1 AP introduction

The command `\apextraintro` produces extra boilerplate for the Introduction to an AP.

EXAMPLE The command `\apextraintro` prints:

Application protocols provide the basis for developing implementations of ISO 10303 and abstract test suites for the conformance testing of AP implementations.

Clause 1 defines the scope of the application protocol and summarizes the functionality and data covered by the AP. Clause 3 lists the words defined in this part of ISO 10303 and gives pointers to words defined

Table 5 – AP package parameterless heading commands

Command	Clause	Default text	Label
\inforeqhead	C	Information requirements	;sireq
\uofhead	SC	Units of functionality	;suof
\applobjhead	SC	Application objects	;sao
\applasserthead	SC	Application assertions	;saa
\aimhead	C	Application interpreted model	;saim
\mappinghead	SC	Mapping table, or	;smap
		Mapping specification	;smap
\templateshead	SSC	Mapping templates	
\aimshortexphead	SC	AIM EXPRESS short listing	;saesl
\confreqheadhead	C	Conformance requirements	;scr
\aimlongexphead	NA	AIM EXPRESS expanded listing	;saeel
\aimshortnameshead	NA	AIM short names	;sasn
\impreqhead	NA	Implementation method specific requirements	;simreq
\aamhead	IA	Application activity model	;saam
\aamdefhead	SC	Application activity model definitions and abbreviations	
\aamfighead	SC	Application activity model diagrams	
\armhead	IA	Application reference model	;sarm
\aimexpressghead	IA	AIM EXPRESS-G	;saeg
\aimexpresshead	IA	AIM EXPRESS listing	
\apusagehead	IA	Application protocol usage guide	;sapug

Table 6 – AP package parameterized heading commands

Command	Clause	Parameterized title
\auofhead	SSC	*<UoF>*
\mapuofhead	SSC	*<UoF>*
\mapobjecthead	SSSC	*<application object>*
\mapattribhead	SSSSC	*<attribute>*

elsewhere. An application activity model that is the basis for the definition of the scope is provided in annex F. The information requirements of the application are specified in clause 4 using terminology appropriate to the application. A graphical representation of the information requirements, referred to as the application reference model, is given in annex G.

Resource constructs are interpreted to meet the information requirements. This interpretation produces the application interpreted model (AIM). This interpretation, given in 5.1, shows the correspondence between the information requirements and the AIM. The short listing of the AIM specifies the interface to the integrated resources and is given in 5.2. Note that the definitions and EXPRESS provided in the integrated resources for constructs used in the AIM may include select list items and subtypes which are not imported into the AIM. The expanded listing given in annex A contains the complete EXPRESS for the AIM without annotation. A graphical representation of the AIM is given in annex H. Additional requirements for specific implementation methods are given in annex C.

8.3.2 AP scope

The command `\apscope{`*<application purpose and context>*`}` produces the boilerplate for the start of an AP scope clause.

EXAMPLE The command `\apscope{application purpose and context.}` prints:

This part of ISO 10303 specifies the use of the integrated resources necessary for the scope and information requirements for *application purpose and context.*

NOTE The application activity model in annex F provides a graphical representation of the processes and information flows that are the basis for the definition of the scope of this part of ISO 10303.

8.3.3 AP information requirements

The command `\inforeqhead` prints the heading for the information requirements clause.

The command `\apinforeq{`<*AP purpose*>`}` produces the boilerplate for the clause.

EXAMPLE The command `\apinforeq{AP purpose.}` prints:

This clause specifies the information required for *AP purpose.*

The information requirements are specified as a set of units of functionality, application objects, and application assertions. These assertions pertain to individual application objects and to relationships between application objects. The information requirements are defined using the terminology of the subject area of this application protocol.

NOTE 1 A graphical representation of the information requirements is given in annex G.

NOTE 2 The information requirements correspond to those of the activities identified as being within the scope of this application protocol in annex F.

NOTE 3 The mapping specification specified in 5.1 shows how the integrated resources and application interpreted constructs are used to meet the information requirements of this application protocol.

8.3.4 AP UoF

The command `\uofhead` prints the heading for the UoF subclause.

The environment `\begin{apuof}`<*item list*>`\end{apuof}` produces the boilerplate for the introduction to the clause.

EXAMPLE Remembering that `\aptitle` was set to `implicit drawing` in the preamble, the commands

```
\begin{apuof}
\item Name of UoF1;
\item Name of UoF2;
\item Name of UoFn.
\end{apuof}
```

prints:

This subclause specifies the units of functionality for the implicit drawing application protocol. This part of ISO 10303 specifies the following units of functionality:

— Name of UoF1;

— Name of UoF2;

— Name of UoFn.

The units of functionality and a description of the functions that each UoF supports are given below. The application objects included in the UoFs are defined in 4.2.

8.3.5 AP application objects

The command `\applobjhead` prints the heading for the application objects subclause.

The command `\apapplobj` produces the boilerplate for the introduction to the clause.

EXAMPLE Remembering that `\aptitle` was set to `implicit drawing` in the preamble, the command `\apapplobj` prints:

This subclause specifies the application objects for the implicit drawing application protocol. Each application object is an atomic element that embodies a unique application concept and contains attributes specifying the data elements of the object. The application objects and their definitions are given below.

8.3.6 AP assertions

The command `\applasserthead` prints the heading for the application assertions subclause.

The command `\apassert` produces the boilerplate for the clause.

EXAMPLE Remembering that `\aptitle` was set to `implicit drawing` in the preamble, the command `\apassert` prints:

This subclause specifies the application assertions for the implicit drawing application protocol. Application assertions specify the relationships between application objects, the cardinality of the relationships, and the rules required for the integrity and validity of the application objects and UoFs. The application assertions and their definitions are given below.

8.3.7 AP mapping table/specification

The command `\mappinghead` prints the heading for the mapping table or mapping specification subclause. The heading text depends on whether or not `\mapspectrue` was put in the preamble.

The command `\apmapping` produces the boilerplate for the introduction to the AP mapping table or specification clause.

NOTE AICs are included in the boilerplate only if the command `\aicinaptrue` is included in the preamble.

EXAMPLE 1 By default, or when `\mapspecfalse` is in the preamble, the command `\apmapping` prints:

This clause contains the mapping table that shows how each UoF and application object of this part of ISO 10303 (see clause 4) maps to one or more AIM constructs (see annex A). The mapping table is organized in five columns.

Column 1) Application element: Name of an application element as it appears in the application object definition in 4.2. Application object names are written in uppercase. Attribute names and assertions are listed after the application object to which they belong and are written in lower case.

Column 2) AIM element: Name of an AIM element as it appears in the AIM (see annex A), the term "IDENTICAL MAPPING", or the term "PATH". AIM entities are written in lower case. Attribute names of AIM entities are referred to as <entity name>.<attribute name>. The mapping of an application element may result in several related AIM elements. Each of these AIM elements requires a line of its own in the table. The term "IDENTICAL MAPPING" indicates that both application objects of an application assertion map to the same AIM element. The term “PATH” indicates that the application assertion maps to the entire reference path.

Column 3) Source: For those AIM elements that are interpreted from the integrated resources or the application interpreted constructs, this is the number of the corresponding part of ISO 10303. For those AIM elements that are created for the purpose of this part of ISO 10303, this is the number of this part. Entities or types that are defined within the integrated resources have an AIC as the source reference if the use of the entity or type for the mapping is within the scope of the AIC.

Column 4) Rules: One or more numbers may be given that refer to rules that apply to the current AIM element or reference path. For rules that are derived from relationships between application objects, the same rule is referred to by the mapping entries of all the involved AIM elements. The expanded names of the rules are listed after the table.

Column 5) Reference path: To describe fully the mapping of an application object, it may be necessary to specify a reference path through several related AIM elements. The reference path column documents the role of an AIM element relative to the AIM element in the row succeeding it. Two or more such related AIM elements define the interpretation of the integrated resources that satisfies the requirement specified by the application object. For each AIM element that has been created for use within this part of ISO 10303, a reference path up to its supertype from an integrated resource is specified.

For the expression of reference paths the following notational conventions apply:

a) `[]` : enclosed section constrains multiple AIM elements or sections of the reference path are required to satisfy an information requirement;

b) `()` : enclosed section constrains multiple AIM elements or sections of the reference path are identified as alternatives within the mapping to satisfy an information requirement;

c) `{}` : enclosed section constrains the reference path to satisfy an information requirement;

d) `<>` : enclosed section constrains at one or more required reference path;

e) `||` : enclosed section constrains the supertype entity;

f) `->` : attribute references the entity or select type given in the following row;

g) `<-` : entity or select type is referenced by the attribute in the following row;

h) `[i]` : attribute is an aggregation of which a single member is given in the following row;

i) `[n]` : attribute is an aggregation of which member `n` is given in the following row;

j) `=>` : entity is a supertype of the entity given in the following row;

k) `<=` : entity is a subtype of the entity given in the following row;

l) `=` : the string, select, or enumeration type is constrained to a choice or value;

m) `\` : the reference path expression continues on the next line;

n) `*` : used in conjunction with braces to indicate that any number of relationship entity data types may be assembled in a relationship tree structure.

EXAMPLE 2 When `\mapspectrue` is in the preamble, the command `\apmapping` prints:

This clause contains the mapping specification that shows how each UoF and application object of this part of ISO 10303 (see clause 4) maps to one or more AIM constructs (see annex A). Each mapping specifies up to five elements.

Application element: The mapping for each application element is specified in a seperate subclause below. Application object names are given in title case. Attribute names and assertions are listed after the application object to which they belong and are given in lower case.

AIM element: The name of one or more AIM entity data types (see annex A), the term "IDENTICAL MAPPING", or the term "PATH". AIM entity data type names are given in lower case. Attributes of AIM entity data types are referred to as <entity name>.<attribute name>. The mapping of an application element may involve more than one AIM element. Each of these AIM elements is presented on a seperate line in the mapping specification. The term "IDENTICAL MAPPING" indicates that both application objects involved in an application assertion map to the same instance of an AIM entity data type. The term "PATH" indicates that the application assertion maps to a collection of related AIM entity instances specified by the entire reference path.

Source: For those AIM elements that are interpreted from any common resource, this is the ISO standard number and part number in which the resource is defined. For those AIM elements that are created for the purpose of this part of ISO 10303, this is "ISO 10303–" followed by the number of this part.

Rules: One or more global rules may be specified that apply to the population of the AIM entity data types specified as the AIM element or in the reference path. For rules that are derived from relationships between application objects, the same rule is referred to by the mapping entries of all the involved AIM elements. A reference to a global rule may be accompanied by a reference to the subclause in which the rule is defined.

Reference path: To describe fully the mapping of an application object, it may be necessary to specify a reference path involving several related AIM elements. Each line in the reference path documents the role of an AIM element relative to the AIM element in the line following it. Two or more such related AIM elements define the interpretation of the integrated resources that satisfies the requirement specified by the application object. For each AIM element that has been created for use within this part of ISO 10303, a reference path to its supertype from an integrated resource is specified. For the expression of reference paths and the relationships between AIM elements the following notational conventions apply:

`[]` enclosed section constrains multiple AIM elements or sections of the reference path are required to satisfy an information requirement;

`()` enclosed section constrains multiple AIM elements or sections of the reference path are identified as alternatives within the mapping to satisfy an information requirement;

`{}` enclosed section constrains the reference path to satisfy an information requirement;

`<>` enclosed section constrains at one or more required reference path;

`||` enclosed section constrains the supertype entity;

`->` attribute references the entity or select type given in the following row;

`<-` entity or select type is referenced by the attribute in the following row;

`[i]` attribute is an aggregation of which a single member is given in the following row;

`[n]` attribute is an aggregation of which member `n` is given in the following row;

`=>` entity is a supertype of the entity given in the following row;

`<=` entity is a subtype of the entity given in the following row;

`=` the string, select, or enumeration type is constrained to a choice or value;

`\` the reference path expression continues on the next line;

- `*` used in conjunction with braces to indicate that any number of relationship entity data types may be assembled in a relationship tree structure;

- `--` the text following is a comment (normally a clause reference).

8.3.7.1 AP mapping templates

The command `\aptemplatehead` prints the heading for the mapping template subclause (if any).

The command `\apmaptemplate` prints the boilerplate for the introduction to the clause. This refers to the UoFs in the AP. The first of the UoFs shall be labelled as `\label{;uof1}` and the last of the UoFs shall be labelled as `\label{;uoflast}`.

EXAMPLE 1 If there are three UoFs, then there should be headings of the form:

```
\mapuofhead{First UoF}\label{;uof1}
...
\mapuofhead{Second UoF}...
...
\mapuofhead{Third UoF}\label{;uoflast}
...
```

EXAMPLE 2 Assuming that there are three UoFs as in the previous example, the command `\apmaptemplate` prints:

This mapping specification includes mapping templates. A mapping template is a reusable portion of a reference path that defines a commonly used part of the structure of the application interpreted model. A mapping template is similar to a programming language macro. The mapping templates used in this part of ISO 10303 are defined in this subclause. Each mapping template definition has three components as follows:

- the template signature that specifies the name of the template and may also specify the names and the order of the formal parameters of the template;

- descriptions of the formal parameters of the template, if any;

- the template body that defines the reusable portion of a reference path and may indicate, through the use of the formal parameter names included in the template signature, the points at which the value parameters are supplied in each template application.

Each mapping template is used at least once in the reference paths specified in 5.1.2 to 5.1.4. Each such template application is a reference to the template definition, based on the pattern established by the template signature, and supplies the value parameters that are to be substitued for the formal parameters specified in the template definition. The full reference path can be derived by replacing any formal parameters in the template body by the value parameters specified in the template application and then substituting the completed template body for the template application.

The non-blank characters following the first '/' define the name of the mapping template. The name of the mapping template is given in upper case. The name of the template is followed by a list of parameter values, seperated by commas, enclosed in parentheses. Parameter values are given in lower case except in the case that the value parameter is a string literal that includes upper case characters.

The following notational conventions apply to the definitions and applications of templates:

- `/` marks the beginning and end of a template signature or a template application;

- `&` prefixes the name of a formal parameter within the definition of a template body;

`()` enclose the formal parameters in a template signature or the value parameters in a template application;

`,` separates formal parameters in a template signature or value parameters in a template application;

`' '` denotes a string literal that is used as a value parameter in a template application.

Value parameters that are not enclosed by quotes are EXPRESS data type identifiers.

This part of ISO 10303 uses the templates that are specified in the following subclauses.

The command `\sstemplates` prints the two subclauses for the SUBTYPE and SUPERTYPE templates.

EXAMPLE 3 In this document, and noting that the clause numbering is not the same as in a real AP document, the command `\sstemplates` prints:

8.3.7.2 SUBTYPE

The SUBTYPE mapping template specifies a reference to the mapping of a subtype of the current application object. Several such references may be included for one supertype application object.

NOTE This template definition only consists of a template signature, there is no matching template body. The template is included to ease the automatic processing of the mapping specification.

Mapping signature:

/SUBTYPE(application_object)/

Parameter definitions:

application_object: the application object that is a subtype of the current supertype application object and that has the entire or a part of the mapping specification of this supertype.

8.3.7.3 SUPERTYPE

The SUPERTYPE mapping template specifies a reference to the mapping of a supertype of the current application object. Several such references may be included for the subtype application object.

NOTE This template only consists of a signature, there is no matching body. The template is included to ease the automatic processing of the mapping specification.

Mapping signature:

/SUPERTYPE(application_object)/

Parameter definitions:

application_object: the application object that is a supertype of the current subtype application object and that has the entire or a part of the mapping specification of this subtype.

8.3.7.4 Template headings

There are three headings used within a mapping template.

The command `\signature` prints the underlined Mapping signature header.

The command `\parameters` prints the underlined Parameter definition header.

The command `\body` prints the underlined Template body header.

EXAMPLE The results of using the `\signature` and `\parameters` commands were illustrated in example 3 on page 27.

8.3.8 AIM short EXPRESS listing

The command `\aimshortexphead` prints the heading for the AIM EXPRESS short listing subclause.

The command `\apshortexpress` produces the boilerplate for the first paragraph of the clause.

NOTE AICs are included in the boilerplate only if the command `\aicinaptrue` is included in the preamble.

EXAMPLE 1 The command `\apshortexpress` without `\aicinaptrue` in the preamble produces:

This clause specifies the EXPRESS schema that uses elements from the integrated resources and contains the types, entity specializations, rules, and functions that are specific to this part of ISO 10303. This clause also specifies modifications to the text for constructs that are imported from the integrated resources. The definitions and EXPRESS provided in the integrated resources for constructs used in the AIM may include select list items and subtypes that are not imported into the AIM. Requirements stated in the integrated resources that refer to select list items and subtypes apply exclusively to those items that are imported into the AIM.

EXAMPLE 2 With `\aicinaptrue` set in the preamble the command `\apshortexpress` produces the following:

This clause specifies the EXPRESS schema that uses elements from the integrated resources and the AICs and contains the types, entity specializations, rules, and functions that are specific to this part of ISO 10303. This clause also specifies modifications to the text for constructs that are imported from the integrated resources and the AICs. The definitions and EXPRESS provided in the integrated resources for constructs used in the AIM may include select list items and subtypes that are not imported into the AIM. Requirements stated in the integrated resources that refer to select list items and subtypes apply exclusively to those items that are imported into the AIM.

8.3.9 AP conformance

The command `\confreqhead` prints the heading for the conformance requirements clause.

The command `\apconformance{`<*implementation methods*>`}` produces the boilerplate for the introduction to the clause.

The environment `\begin{apconformclasses}`<*item list*>`\end{apconformclasses}` provides some additional boilerplate.

EXAMPLE 1 The command `\apconformance{ISO 10303-21, ISO 10303-22}` prints:

Conformance to this part of ISO 10303 includes satisfying the requirements stated in this part, the requirements of the implementation method(s) supported, and the relevant requirements of the normative references.

An implementation shall support at least one of the following implementation methods: *ISO 10303-21, ISO 10303-22*.

Requirements with respect to implementation methods-specific requirements are specified in annex C.

The Protocol Information Conformance Statement (PICS) proforma lists the options or the combination of options that may be included in the implementation. The PICS proforma is provided in annex D.

EXAMPLE 2 The commands

```
\begin{apconformclasses}
\item first class;
\item second class;
\item last class.
\end{apconformclasses}
```

print:

This part of ISO 10303 provides for a number of options that may be supported by an implementation. These options have been grouped into the following conformance classes:

— first class;

— second class;

— last class.

Support for a particular conformance class requires support of all the options specified in this class.

8.3.10 EXPRESS expanded listing

The command `\aimlongexphead` prints the heading for the AIM expanded listing clause.

The command `\aimlongexp` produces the boilerplate for the introduction to the clause.

EXAMPLE The command `\aimlongexp` prints:

The following EXPRESS is the expanded form of the short form schema given in 5.2. In the event of any discrepancy between the short form and this expanded listing, the expanded listing shall be used.

8.3.11 AIM short names

The command `\aimshortnamehead` prints the heading for the AIM short names annex.

The command `\apshortnames` produces the boilerplate for the introduction to the AP short name annex.

EXAMPLE The command `\apshortnames` prints:

Table B.1 provides the short names of entities specified in the AIM of this part of ISO 10303. Requirements on the use of the short names are found in the implementation methods included in ISO 10303.

8.3.12 Implementation requirements

the command `\impreqhead` prints the heading for implementation method-specific reguirements.

The command `\apimpreq{`<*schema name*>`}` produces the boilerplate for the requirements on exchange structure.

EXAMPLE The command `\apimpreq{schema\_name}` prints:

The implementation method defines what types of exchange behaviour are required with respect to this part of ISO 10303. Conformance to this part of ISO 10303 shall be realized in an exchange structure. The file format shall be encoded according to the syntax and EXPRESS language mapping defined in ISO 10303-21 and in the AIM defined in annex A of this part of ISO 10303. The header of the exchange structure shall identify use of this part of ISO 10303 by the schema name '*schema_name*'.

8.3.13 AP PICS

The command `\picshead`, from the `step` package, prints the heading for the PICS annex.

The command `\picsannex` produces the boilerplate for the start of the AP PICS annex.

EXAMPLE The command `\picsannex` prints:

This clause lists the optional elements of this part of ISO 10303. An implementation may choose to support any combination of these optional elements. However, certain combinations of options are likely to be implemented together. These combinations are called conformance classes and are described in the subclauses of this annex.

This annex is in the form of a questionnaire. This questionnaire is intended to be filled out by the implementor and may be used in preparation for conformance testing by a testing laboratory. The completed PICS proforma is referred to as a PICS.

8.3.14 AAM annex

The command `\aamhead` prints the heading for the AAM annex.

The command `\apaamintro` produces the introductory boilerplate for the introduction of the AP annex on application activity models.

EXAMPLE The command `\apaamintro` prints:

The application activity model (AAM) is provided as an aid in understanding the scope and information requirements defined in this application protocol. The model is presented as a set of figures that contain the activity diagrams and a set of definitions of the activities and their data. Activities and data flows that are out of scope are marked with an asterisk.

8.3.15 AP AAM definitions

The command `\aamdefhead` prints the heading for the AAM definitions subclause.

The command `\apaamdefs` produces the boilerplate at the start of the AP subclause on AAM definitions and abbreviations.

EXAMPLE The command `\apaamdefs` prints:

The following terms are used in the application activity model. Terms marked with an asterisk are outside the scope of this application protocol.

The definitions given in this annex do not supersede the definitions given in the main body of the text.

8.3.16 AAM diagrams annex

The command `\aamfighead` prints the heading for the AAM diagrams subclause.

The command `\aamfigrange{`<*figure range*>`}` is used to store the activity model diagram figure range for later use.

EXAMPLE 1 For the purposes of this document we set

```
\aamfigrange{figures F.1 through F.n}
```

The command `\aamfigures` produces the boilerplate for the introduction to an APs AAM figure subclause.

EXAMPLE 2 Noting that we have set `\aamfigrange{figures F.1 through F.n}`, the command `\aamfigures` prints:

The application activity model diagrams are given in *figures F.1 through F.n.* The graphical form of the application activity model is presented in the IDEF0 activity modelling format [11]. Activities and data flows that are out of scope are marked with asterisks.

8.3.17 ARM annex

The command `\armhead` prints the heading for the ARM annex.

The command `\armintro` produces the boilerplate for the introduction to the ARM figures.

EXAMPLE The command `\armintro` prints:

This annex provides the application reference model for this part of ISO 10303. The application reference model is a graphical representation of the structure and constraints of the application objects specified in clause 4. The graphical form of the application reference model is presented in EXPRESS-G. The application reference model is independent from any implementation method. EXPRESS-G is defined in annex D of ISO 10303-11.

8.3.18 AIM EXPRESS-G annex

The command `\aimexpressghead` prints the heading for the AIM EXPRESS-G annex.

The command `\aimexpressg` produces the boilerplate for the introduction to an AP's AIM EXPRESS-G model.

EXAMPLE The command `\aimexpressg` prints:

The diagrams in this annex correspond to the AIM EXPRESS expanded listing given in annex A. The diagrams use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined in annex D of ISO 10303-11.

8.3.19 AIM EXPRESS listing

The command `\aimexpresshead` prints the heading for the AIM listing annex.

The command \apexpurls{<*short*>}{<*express*>] produces the boilerplate for the introduction to the AP annex listing short names and EXPRESS, where <*short*> is the URL for the short names and <*express*> is the URL for the EXPRESS.

EXAMPLE The command `\apexpurls{http:/www.short/}{http://www.express/}` prints:

This annex provides a listing of the complete EXPRESS schema specified in annex A of this part of ISO 10303 without comments or explanatory text. It also provides a listing of the EXPRESS entity names and corresponding short names as specified in annex B of this part of ISO 10303. The content of this annex is available in computer-interpretable form and can be found at the following URLs:

— Short names: <http://www.short/>

— EXPRESS: <http://www.express/>

If there is difficulty accessing these sites contact ISO Central Secretariat or contact the ISO TC 184/SC4 Secretariat directly at: sc4sec@cme.nist.gov.

NOTE The information provided in computer-interpretable form at the above URLs is informative. The information that is contained in the body of this part of ISO 10303 is normative.

Table 7 – AIC package parameterless heading commands

Command	Clause	Default text	Label
\aicshortexphead	C	EXPRESS short listing	;sesl

9 The aic package facility

The aic package provides commands and environments specifically for the ISO 10303 Application Interpreted Construct series of documents.

The use of this package requires the use of the step package.

9.1 Heading commands

The commands described in this subclause start document clauses with particular titles.

The commands that take no parameters are listed in Table 7.

9.2 Boilerplate commands

The following commands produce boilerplate text as specified by the Supplementary Directives.

9.2.1 Introduction text

The command \aicextraintro prints additional boilerplate for the Introduction to an AIC.

EXAMPLE The command \aicextraintro prints:

This part of ISO 10303 is a member of the application interpreted construct series. An application interpreted construct (AIC) provides a logical grouping of interpreted constructs that supports a specific functionality for the usage of product data across multiple application contexts. An interpreted construct is a common interpretation of the integrated resources that supports shared information requirements among application protocols.

9.2.2 Definition of AIC

The command \aicdef prints the definition of 'AIC'. It shall only be used within the definitions environment.

EXAMPLE The commands:

```
\begin{definitions}
\aicdef
\end{definitions}
```

produce:

9.2.2.1
application interpreted construct (AIC)
a logical grouping of interpreted constructs that supports a specific function for the usage of product data across multiple application contexts.

9.2.3 Short EXPRESS listing

The command `\aicshortexphead` prints the heading for the AIC short EXPRESS annex.

The command `\aicshortexpintro` prints boilerplate for the introduction to the short EXPRESS listing.

EXAMPLE The command `\aicshortexpintro` prints:

This clause specifies the EXPRESS schema that uses elements from the integrated resources and contains the types, entity data types specializations, and functions that are specific to this part of ISO 10303.

NOTE There may be subtypes and items of select lists that appear in the integrated resources that are not imported into the AIC. Constructs are eliminated from the subtype tree or select list through the use of the implicit interface rules of ISO 10303-11. References to eliminated constructs are outside the scope of the AIC. In some cases, all items of the select list are eliminated. Because AICs are intended to be implemented in the context of an application protocol, the items of the select list will be defined by the scope of the application protocol.

9.2.4 EXPRESS-G figures

The command `\expressghead`, from the `step` package, prints the heading for the EXPRESS-G diagrams annex.

The command `\aicexpressg` prints boilerplate for the introduction to the EXPRESS-G figures.

EXAMPLE The command `\aicexpressg` prints:

The diagrams in this annex are generated from the short listing given in clause 4 and correspond to the EXPRESS schemas specified in this part of ISO 10303. The diagrams use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined in annex D of ISO 10303-11.

10 The ats package facility

The `ats` package provides commands and environments specifically for the ISO 10303 Abstract Test Suite series of documents.

The use of this package requires the use of the `step` package.

10.1 Preamble commands

Certain commands shall be put in the preamble of an ATS document.

The command `\APnumber{`<*number*>`}` shall be put in the preamble, where <*number*> is the ISO 10303 part number of the corresponding AP.

EXAMPLE 1 For the purposes of later examples, the command `\APnumber{299}` has been put in the preamble. of this document.

The command `\APtitle{`<*title of AP*>`}` shall be put in the preamble, where <*title of AP*> is the ISO 10303 part title of the corresponding AP. This must be given in such a manner that it reads sensibly in a sentence of the form '…for ISO 10303-299, application protocol <*title of AP*>.'

EXAMPLE 2 For the purposes of later examples, the command `\APtitle{abstract painting}` has been put in the preamble of this document.

The command `\mapspectrue` shall be put in the preamble if the AP uses a mapping specification rather than a mapping table.

10.2 Heading commands

These commands start document clauses with particular titles. The commands that take no parameters are listed in Table 8.

The commands that take a parameter are listed in Table 9.

10.3 Keyword commands

Several keyword (headings) are used in documenting a test case. LaTeX commands for these keywords are given in Table 10.

10.4 Boilerplate commands

The following commands produce boilerplate text.

NOTE In the examples, the parameters of those commands that take them have been specified in *this font style* so that their effects can be seen in the printed text.

10.4.1 ATS introduction

The command `\atsintroendbp` produces the boilerplate for the end of the Introduction to an ATS.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299` and `\APtitle` was set to `abstract painting`, the command `\atsintroendbp` prints:

Table 8 – ATS package parameterless heading commands

Command	**Clause**	**Default text**
`\purposeshead`	C	Test purposes
`\aepurposehead`	SC	Application element test purposes
`\aimpurposehead`	SC	AIM test purposes
`\implementpurposehead`	SC	Implementation method test purposes
`\domainpurposehead`	SC	Domain test purposes
`\otherpurposehead`	SC	Other test purposes
`\gtpvchead`	C	General test purposes and verdict criteria
`\generalpurposehead`	SC	General test purposes
`\gvcatchead`	SC	General verdict criteria for all abstract test cases
`\gvcprehead`	SC	General verdict criteria for preprocessor abstract test cases
`\gvcposthead`	SC	General verdict criteria for postprocessor abstract test cases
`\atchead`	C	Abstract test cases
`\prehead`	SSC	Preprocessor
`\precoveredhead`	SSSC	Test purposes covered
`\preinputhead`	SSSC	Input specification
`\precriteriahead`	SSSC	Verdict criteria
`\preconstraintshead`	SSSC	Constraints on values
`\preexechead`	SSSC	Execution sequence
`\preextrahead`	SSSC	Extra details
`\posthead`	SSC	Postprocessor
`\postcoveredhead`	SSSC	Test purposes coverage
`\postinputhead`	SSSC	Input specification
`\postcriteriahead`	SSSC	Verdict criteria
`\postexechead`	SSSC	Execution sequence
`\postextrahead`	SSSC	Extra details
`\confclassannexhead`	NA	Conformance classes
`\postipfilehead`	NA	Postprocessor input specification file names
`\atsusagehead`	IA	Usage scenarios
`\apasserthead`	SSC	Application assertions

Table 9 – ATS package parameterized heading commands

Command	Clause	Parameterized title
`\apobjhead`	SSC	<*Application object n*>
`\aimenthead`	SSC	<*Entity name*>
`\atctitlehead`	SC	<*Title*>
`\confclasshead`	SC	Conformance class <*number*>

Table 10 – ATS package keyword commands

Command	Effect
`\atssummary`	`Test case summary:`
`\atscovered`	`Test purposes covered:`
`\atsinput`	`Input specification:`
`\atsconstraints`	`Constraints on values:`
`\atsverdict`	`Verdict criteria:`
`\atsexecution`	`Execution sequence:`
`\atsextra`	`Extra details:`

The purpose of an abstract test suite is to provide a basis for evaluating whether a particular implementation of an application protocol actually conforms to the requirements of that application protocol. A standard abstract test suite helps ensure that evaluations of conformance are conducted in a consistent manner by different test laboratories.

This part of ISO 10303 specifies the abstract test suite for ISO 10303-299, application protocol abstract painting. The abstract test cases presented here are the basis for conformance testing of implentations of ISO 10303-299.

This abstract test suite is made up of two major parts:

— the test purposes, the specific items to be covered by conformance testing;

— the set of abstract test cases that meet those test purposes.

The test purposes are statements of the application protocol requirements that are to be addressed by the abstract test cases. Test purposes are derived primarily from the application protocol's information requirements and AIM, as well as from other sources such as standards referenced by the application protocol and other requirements stated in the application protocol conformance requirements clause.

The abstract test cases address the test purpose by:

— specifying the requirements for input data to be used when testing an implementation of the application protocol;

— specifying the verdict criteria to be used when evaluating whether the implementation successfully converted the input data to a different form.

The abstract test cases set the requirements for the executable test cases that are required to actually conduct a conformance test. Executable test cases contain the scripts, detailed values, and other explicit information required to conduct a conformance test on a specific implementation of the application protocol.

At the time of publication of this document, conformance testing requirements had been established for implementations of application protocols in combination with ISO 10303-21 and ISO 10303-22. This part of ISO 10303 only specifies test purposes and abstract test cases for a subset of such implementations.

For ISO 10303-21, two kinds of implementations, preprocessors and postprocessors, must be tested. Both of these are addressed in this abstract test suite.

For ISO 10303-22, a class of applications will possess the capability to upload and download AP-compliant SDAI-models or schema instances to and from applications that implement the SDAI. By providing test case data that correspond with SDAI-models, this abstract test suite addresses such applications in a single-schema scenario.

10.4.2 ATS scope

The command `\scopeclause`, from the isov2 class, prints the heading for the Scope clause.

The command `\atsscopebp` produces boilerplate for an ATS *Scope* clause.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299`, the command `\atsscopebp` prints:

This part of ISO 10303 specifies the abstract test suite to be used in the conformance testing of implementations of ISO 10303-299. The following are within the scope of this part of ISO 10303:

— the specification of the test purposes associated with ISO 10303-299;

— the verdict criteria to be applied during conformance testing of an implementation of ISO 10303-299 using ISO 10303-21 or ISO 10303-22;

  NOTE The verdict criteria are used to ascertain whether a test purpose has been satisfactorily met by an implementation under test (IUT) within the context of a given test case.

— the abstract test cases to be used as the basis for the executable test cases for conformance testing.

The following are outside the scope of this part of ISO 10303:

— the creation of executable test cases;

— test specifications for tests other than conformance testing such as interoperability or acceptance testing;

— other implementation methods.

10.4.3 Test purpose

The command `\purposehead` prints the heading for the test purposes clause.

The command `\atspurposebp` prints boilerplate for the introduction to the clause.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299`, the command `\atspurposebp` prints:

This clause specifies the test purposes for this part of ISO 10303. Clauses 4.1 and 4.2 are describe the source and meaning of test purposes that are derived from the information requirements defined in ISO 10303-299, clause 4, and the AIM EXPRESS schema defined in ISO 10303-299, annex A. These test purposes are not repeated in this part of ISO 10303. However, through reference in a test case each specific element from the application elements of the AIM implicitly requires that the identified element, as specified in the test purpose statement, will be correctly instantiated by the implementation under test.

10.4.4 Application element test purposes

The command `\aepurposehead` prints the heading for the application element test purposes subclause.

The command `\aetpbp` prints boilerplate for the clause.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299`, the command `\aetpbp` prints:

Application element (AE) test purposes are implicitly derived from the AP information requirements and are not explicitly documented here. AE test purposes apply to the input specifications of both preprocessr and postprocessor test cases. AE test purposes are implicitly derived from the AP information requirements as follows:

— Application objects (see ISO 10303-299, 4.2): a test purpose derived from an application object is a simple statement of the object's name;

— Application object attributes (see ISO 10303-299, 4.2): test purposes derived from application object attributes are statements of the application object name with a specific attribute name;

— Application assertions (see ISO 10303-299, 4.3): test purposes derived from application assertions are statements describing the relationships between two application objects. Application assertion test purposes address the directions of relationships as well as the number (cardinality) of relationships.

They shall be interpreted as given in the following statement: the IUT shall preserve the semantic associated with the unique application element from which the test purpose was implicitly derived. This implies that the semantics of the application element are preserved by the IUT between the input and output of a test, according to the reference path specified by the mapping specification defined in ISO 10303-299, 5.1.

10.4.5 AIM test purposes

The command `\aimpurposehead` prints the heading for the AIM test purposes subclause.

The command `\aimtpbp` prints boilerplate for the clause.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299`, the command `\aimtpbp` prints:

Test purposes are implicitly derived from the AP AIM EXPRESS, and are not explicitly documented here. AIM test purposes are implicitly derived from the expanded EXPRESS listing contained in annex A of ISO 10303-299 as follows:

— AIM entity data types: a test purpose derived from an AIM entity data type is a simple statement of the entity data type name;

— AIM entity attributes: a test purpose derived from an AIM entity attribute is a statement of the AIM entity data type with a given attribute.

Aim test purposes shall be interpreted as given in the following statement: the postprocessor shall accept the input in accordance with the AIM EXPRESS structure corresponding to this test purpose. This implies that the semantics of the application element represented by the AIM element are preserved by the IUT between the input and output of a test according to the reference path specified in the mapping specification of the AP. This also implies no violations of any constraints (local rules or global rules) that apply to the AIM element. AIM test purposes apply to the input specifications of postprocessor test cases only.

10.4.6 Implementation method test purposes

The command `\implementpurposehead` prints the heading for the implementation method test purposes subclause.

The command `\atsimtpbp` prints boilerplate for the clause.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299`, the command `\atsimtpbp` prints:

The following test purpose is derived from requirements in ISO 10303-21 and applies to preprocessors only.

other1 The IUT correctly encodes the AIM schema name in the exchange structure.

The following test purposes are derived from requirements in ISO 10303-21 and apply to postprocessors only.

other2 The IUT interprets the ISO 10303-21 header section present in the exchange structure.

other3 The IUT interprets the ISO 10303-21 SCOPE and EXPORT constructs present in the exchange structure.

other4 The IUT interprets the ISO 10303-21 user-defined entity constructs present in the exchange structure.

other5 The IUT interprets various representations of numbers present in the exchange structure in accordance with ISO 10303-21.

other6 The IUT interprets various sequences of symbols present in the exchange structure in accordance with ISO 10303-21.

10.4.7 General test purposes and verdict criteria

The command `\gtpvchead` prints the heading for the general test purposes and verdict criteria clause.

The command `\atsgtpvcbp` prints boilerplate for the clause

EXAMPLE The command `\atsgtpvcbp` prints:

General test purposes are statements of requirements that apply to all abstract test cases, all preprocessor abstract test cases, or all postprocessor abstract test cases. General verdict criteria are the means for evaluating whether the general test purposes are met. General verdict criteria shall be evaluated as a part of every executable test case to which they apply. Each general verdict criterion includes a reference to its associated test purpose.

10.4.8 General test purposes

The command `\generalpurposehead` prints the heading for the general test purposes subclause.

The command `\gtpbp` prints boilerplate for the suclause.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299`, the command `\gtpbp` prints:

The following are the general test purposes for this part of ISO 10303:

g1 The output of an IUT shall preserve all the semantics defined by the input model according to the reference paths specified in the mapping specification defined in clause 5 of ISO 10303-299.

g2 The output of a preprocessor shall conform to the implementation method to which the IUT claims conformance.

g3 The instances in the output of a preprocessor shall be encoded according to the mapping specification and the AIM EXPRESS long form defined in 5.1 and annex A of ISO 10303-299.

g4 A postprocessor shall accept input data which is encoded according to the implementation method to which the IUT claims conformance.

g5 A postprocessor shall accept input data structured according to the mapping specification and the AIM EXPRESS long form defined in 5.1 and annex A of ISO 10303-299.

10.4.9 General verdict criteria

The command `\gvcatchead` prints the heading for the general verdict criteria for all cases subclause.

The command `\gvatcbp` prints boilerplate for the subclause.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299`, the command `\gvatcbp` prints:

The following verdict criteria apply to all abstract test cases contained in this part of ISO 10303:

gvc1 The semantics of the input model are preserved in the output of the IUT according to the reference paths specified in the mapping specification defined in ISO 10303-299, clause 5 (g1).

10.4.10 General verdict criteria for preprocessor

The command `\gvcprehead` prints the heading for the general verdict criteria for preprocessor cases subclause.

The command `\gvcprebp` prints boilerplate for the subclause.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299`, the command `\gvcprebp` prints:

The following verdict criteria apply to all preprocessor abstract test cases contained in this part of ISO 10303:

gvc2 The output of a preprocessor conforms to the implementation method to which the IUT claims conformance (g2).

gvc3 The instances in the output of a preprocessor are encoded according to the AIM EXPRESS long form and mapping specification defined in ISO 10303-299, annex A and 5.1 (g3).

10.4.11 General verdict criteria for postprocessor

The command `\gvcposthead` prints the heading for the general verdict criteria for postprocessor cases subclause.

The command `\gvcpostbp` prints boilerplate for the subclause.

EXAMPLE Remembering that in the preamble `\APnumber` was set to `299`, the command `\gvcpostbp` prints:

The following verdict criteria apply to all postprocessor abstract test cases contained in this part of ISO 10303:

gvc4 The postprocessor accepts input data which is encoded according to the implementation method to which the IUT claims conformance (g4).

gvc5 The postprocessor accepts input data which is structured according to the AIM EXPRESS long form and mapping specification defined in ISO 10303-299, annex A and 5.1 (g5).

10.4.12 Abstract test cases

The command `\atchead` prints the heading for the abstract test cases clause.

The command `\atcbp` prints the first paragraph of the boilerplate for the clause.

EXAMPLE 1 The command `\atcbp` prints:

This clause specifies the abstract test cases for this part of ISO 10303. Each abstract test case addresses one or more test purposes explicitly or implicitly specified in clause 4.

The command `\atcbpii` prints paragraphs 3 and onwards of the boilerplate.

EXAMPLE 2 The command `\atcbpii` prints:

Each abstract test case has a subclause for the preprocessor test information and a subclause for each postprocessor input specification and related test information. The preprocessor and postprocessor input specifications are mirror images of each other: they represent the same semantic information. The preprocessor input model is presented in the form of a table with five columns:

— The Id column contains an identifier for the application object instantiated in a particular row. The identifier may be referenced as the value of an application assertion. The identifier is the lowest-level subclause number from ISO 10303-299, 4.2 where the application element that appears in that row of the table is specified.

— The V column specifies whether or not the element in that row of the table is assigned a verdict in this test case. A blank indicates that it is not assigned a verdict in this test case. A '*' indicates that it is assigned a verdict using a derived verdict criteria. The derived verdict criteria determine whether the semantics associated with the application element are preserved in the output of the IUT according to the reference paths specified in the mapping table defined in ISO 10303-299, 5.1. A number in the V column references a specific verdict criterion defined in the verdict criteria section that follows the preprocessor input specification table.

— The Application Elements column identifies the particular application element instance that is being defined by the table. For assertions the role is specified in parenthesis.

— The Value column specifies a specific value for the application element. For application objects and attributes the value column defines the semantic value for that element's instance in the input model. A '#<number>' in the column is a reference to an entity instance name in the postprocessor input specification where the corresponding value is specified. For assertions, this column holds a link to the related application object. A '<not_present>' indicates that the application element is not present in the input model.

— The Req column specifies whether the value in the Value column is mandatory (M), suggested (S) or constrained (C<n>), where 'n' is an integer referencing a note that follows the table. A suggested value may be changed by the test realizer. A mandatory value may not be changed due to rules in EXPRESS, rules in the mapping specification, or the requirements of the test purpose being assigned a verdict. Each constrained value references a note labelled C<number> at the end of the preprocessor input model table and may be modified according to specific constraints specified in it.

The postprocessor input specifications are defined using ISO 10303-299. The values in the postprocessor specifications are suggested unless declared mandatory or constrained by the preprocessor input table.

The abstract test case specifies all the verdict criteria that are used to assign a verdict during testing. Special verdict criteria for preprocessor and postprocessor testing are defined explicitly in each abstract test case subclause. The relevant derived verdict criteria for preprocessor and postprocessor testing are identified in the V column of the preprocessor input table.

10.4.13 Preprocessor

The command `\prehead` prints the title for the preprocessor subsubclause.

The command `\atcpretpc` prints boilerplate for the subclause.

EXAMPLE The command `\atcpretpc` prints:

In the preprocessor input specification table of this test case, the numbers in the Id column (ignoring the part beyond the decimal point, if any) whose rows are not empty in the V column identify the application objects that are covered by this test case. These Id numbers refer directly to the subclause numbers within ISO 10303-299, 4.2, where the application object is defined.

10.4.14 Postprocessor

The command `\posthead` prints the title for the postrocessor subsubclause.

The command `\atcposttpc` prints boilerplate for the subclause.

EXAMPLE The command `\atcposttpc` prints:

In the postprocessor input specification table of this test case, the numbers in the Id column (ignoring the part beyond the decimal point, if any) whose rows are not empty in the V column identify the application objects that are covered by this test case. These Id numbers refer directly to the subclause numbers within ISO 10303-299, 4.2, where the application object is defined.

10.4.15 Conformance class

The command `\confclassannexhead` prints the heading for the conformance classes annex heading.

The command `\atsnoclassesbp` prints the entire boilerplate for the *Conformance class* annex when the AP has no conformance classes.

EXAMPLE 1 Remembering that in the preamble `\APnumber` was set to `299`, the command `\atsnoclassesbp` prints:

Conformance to ISO 10303-299 is defined only in terms of the entire AP. Therefore, conformance requires that an implementation pass executable versions of all abstract test cases in clause 6.

The command `\confclasshead{`<*number*>`}` prints the heading for a conformance class <*number*> subclause.

The command `\confclassbp{`<*number*>`}` prints the boilerplate for the introduction to a conformance class subclause, where <*number*> is the number of the conformance class.

EXAMPLE 2 Remembering that in the preamble `\APnumber` was set to `299`, the command `\confclassbp{27}` prints:

To conform to conformance class *27* of ISO 10303-299, an implementation shall pass executable versions of the following abstract test cases:

10.4.16 Postprocessor input specification file names

The command `\postipfilehead` prints the heading for the postprocessor input file names annex.

The command `\pisfbp{`*<12 or 21>*`}{`*<url>*`}{`*<ref>*`}` prints the boilerplate for the annex.

EXAMPLE The command `\pisfbp{12}{http://www.mel.nist.gov/step/parts/parts3456/wd}{\ref{TabB1}}` prints:

This annex references a listing of the postprocessor input specifications for this part of ISO 10303 without comments or other explanatory text. These specifications are documented using ISO 10303-12. These specifications are available in computer-interpretable form and can be found at the following URL:

`<http://www.mel.nist.gov/step/parts/part3456/wd>`

If there is difficulty accessing this site contact the ISO Central Secretariat or contact the ISO TC184/SC4 Secretariat directly at: `sc4sec@cme.nist.gov`.

The postprocessor input specifications for each test case is supplied electronically via the Internet. Table B.1 lists the file name of the postprocessor input specification that is associated with the postprocessor subclause(s) of a test case.

Annex A
(normative)
Additional commands

A.1 Internal commands

The code implementing the various facilities includes many commands not described in the body of this document. Any command that includes the commercial at sign (`@`) in its name shall not be used by any author; the implementer of the package code reserves the right to modify or delete these at any time without giving any notice.

Internal commands that have names consisting only of letters may be used in a document at the author's own risk. These may be changed, but if so notification will be given.

A.2 Boilerplate

Much of the boilerplate text is maintained in separate `.tex` files and many of the commands that generate boilerplate merely `\input` the appropriate file.

Annex B
(normative)
Ordering of LaTeX commands

The LaTeX commands to produce an ISO 10303 document are:

```
\documentclass[<options>]{isov2}
\usepackage{stepv13}                                 % required package
\usepackage{irv12}                                   % for an IR document
\usepackage{apv12}                                   % for an AP document
\usepackage{aicv1}                                   % for an AIC document
\usepackage{atsv11}                                  % for an ATS document
\usepackage[<options>]{<name>}                       % additional packages
\standard{<standard identifier>}
\yearofedition{<year>}
\languageofedition{<parenthesized code letter>}
\partno{<part number>}
\series{<series title>}
\doctitle{<title on cover page>}
\ballotcycle{<number>}
\aptitle{<title of AP>}  % if doc is an AP
\aicinaptrue             % if doc is an AP that uses AICs
\mapspectrue             % if doc is an AP that uses mapping spec.
\APnumber{<number>}      % if doc is an ATS
\APtitle{<title>}        % if doc is an ATS
\mapspectrue             % if doc is an ATS and AP uses mapping spec.
  % other preamble commands
\begin{document}
\STEPcover{< title commands >}
\Foreword                          % start Foreword & ISO boilerplate
  \fwdshortlist                    % STEP boilerplate
\endForeword{<param1>}{<param2>}   % end Foreword & boilerplate
\begin{Introduction}               % start Introduction & boilerplate
  \aicextraintro           % extra boilerplate for an AIC
  \apextraintro              % extra boilerplate for an AP
  % your text
\end{Introduction}
\stepparttitle{<Part title>}
\scopeclause                       % Clause 1: Scope clause
  \apscope{<AP purpose>}           % boilerplate if an AP
   % text of scope
\normrefsclause                    % Clause 2: Normative references
  \normrefbp{<document type>}      % boilerplate
  \begin{nreferences}
    % \isref{}{} and/or \disref{}{} list of normative references
  \end{nreferences}
\defclause                         % definitions clause
  \partidefhead                    % defs from Part1 subclause
    % olddefinition list
  \refdefhead{<ISO 10303-NN>}      % defs from Part NN subclause
    % olddefinition list
  \otherdefhead                    % defs in this part
```



```
    % definition list
\symabbclause                      % Symbols & abbreviations clause
  % symbol lists
% THE BODY OF THE DOCUMENT
\bibannex                          % optional; the final Bibliography
  % bibliography listing
% the index
\end{document}
```

B.1 Body of a resource document

The body of a resource document has the following structure:

```
\schemahead{<Schema name>}         % repeat for each schema
  \introsubhead                    % intro subclause
     % text
  \fcandasubhead                   % concepts subclause
     % text
  \typehead{<Schema>}              % if type defs
     \atypehead{<type>}            % type heading
  \entityhead{<Schema>}{<group>}   % if entity defs
     \anentityhead{<entity>}       % entity heading
  \rulehead{<Schema>}              % if rule defs
     \arulehead{<rule>}            % rule heading
  \functionhead{<Schema>}          % if function defs
     \afunctionhead{<function>}    % function heading
% repeat above for each schema
\shortnamehead                     % Annex A: Short names of entities
  \irshortnames                    % boilerplate
  % list of short names
\objreghead                        % Annex B: Information object registration
  \docidhead                       % Document identification subclause
    \docreg{<param1>}                                    % boilerplate
  \schemaidhead                    % Schema identification subclause
% Either (for single schema)
     \schemareg{<6 parameters>}    % boilerplate
% Or (for multiple schemas) repeat:
     \aschemaidhead{<schema name>} % Schema id subsubclause
       \schemareg{<6 parameters>}
\listingshead                       % Annex C: Computer interpretable listings
  \expurls{<short>}{<express>}      % boilerplate
\expressghead                       % Annex D: EXPRESS-G figures
  \irexpressg                       % boilerplate
  %  EXPRESS-G diagrams
\techdischead                       % optional Technical discussions
  % text
\exampleshead                       % optional Examples
  % text
```

B.2 Body of an application protocol

The body of an AP document has the following structure:

```
\inforeqhead                    % Clause 4: Information requirements
  \apinforeq{<param1>}          % boilerplate
  \uofhead                      % Clause 4.1: Units of functionality
    \begin{apuof}               % boilerplate
      % \item list of UoFs
    \end{apuof}
    \auofhead{<UoF1>}           % repeat for each UoF
      % text
    \applobjhead                % Clause 4.2: Application objects
      \apapplobj                % boilerplate
        % text
    \applasserthead             % Clause 4.3: Application assertions
      \apassert                 % boilerplate
        % text
\aimhead                        % Clause 5: Application interpreted model
  \maptablehead                 % Clause 5.1: Mapping table/specification
    \apmapping                  % boilerplate
    \maptemplatehead            % if mapping templates used
      \apmaptemplate            % template boilerplate
      \sstemplates              % sup/sub templates
      \templatehead
        % text
     \mapuofhead{<Uof>}         % mapping for <UoF>
       \mapobjecthead{<object>}
         % mapping for <object>
         \mapattributehead{<attr>}
           % mapping for <attr>
  \aimshortexphead              % Clause 5.2: AIM EXPRESS short listing
    \apshortexpress             % boilerplate
      % text
\confreqhead                    % Clause 6: Conformance requirements
  \apconformance{<param1>}      % boilerplate
  \begin{apconformclasses}      % optional boilerplate
    % \item list
  \end{apconformclasses}
     % text
\aimlongexphead                 % Annex A: AIM EXPRESS expanded listing
  \aimlongexp                   % boilerplate
     % text
\aimshortnameshead              % Annex B: AIM short names
  \apshortnames                 % boilerplate
     % text
\impreqhead                     % Annex C: Impl. specific reqs
  \apimpreq{<schema name>}      % boilerplate
\picshead                       % Annex D: PICS
  \picsannex                    % boilerplate
     % text
\objreghead                     % Annex E: Information object registration
  \docidhead                    % Annex E.1: Document identification
    \docreg{<param1>}           % boilerplate
  \schemaidhead                 % Annex E.2: Schema identification
    \apschemareg{<6 params>}    % boilerplate
```



```
\aamhead                       % Annex F: Application activity model
  \aamfigrange{<figure range>} % Figure range for AAM diagrams
  \apaamintro                  % boilerplate
      % text
  \aamdefhead                  % Annex F.1: AAM defs and abbreviations
    \apaamdefs                 % boilerplate
       % text
  \aamfighead                  % Annex F.2: AAM diagrams
    \aamfigures                % boilerplate
       % IDEF0 diagrams
\armhead                       % Annex G: Application reference model
   \armintro                   % boilerplate
     % ARM figures
\aimexpressghead               % Annex H: AIM EXPRESS-G
  \aimexpressg                 % boilerplate
     % AIM figures
\listingshead                  % Annex J: Computer interpretable listings
  \apexpurls{<short>}{<express>}   % boilerplate
\apusagehead                   % optional Annex: AP usage
   % text
\techdischead                  % optional Annex: Technical discussions
   % text
```

B.3 Body of an AIC

The body of an AIC document has the following structure:

```
\aicshortexphead               % Clause 4: EXPRESS short listing
  \aicshortexpintro            % boilerplate
  \fcandasubhead               % Clause 4.1 fundamental concepts
    % text
  \typehead{<Schema>}          % if type definitions
     \atypehead{<type>}        % repeat for each type
  \entityhead{<Schema>}{}      % if entity defs
     \anentityhead{<entity>}   % repeat for each entity
  \functionhead{<Schema>}      % if function defs
     \afunctionhead{<function>} % repeat for each function
\shortnamehead                 % Annex A: Short names of entities
  \shortnames                  % boilerplate
\objreghead                    % Annex B: Information object registration
  \docidhead                   % Annex B.1: Document identification
    \docreg{<version no>}      % boilerplate
  \schemaidhead                % Annex B.2: Schema identification
    \schemareg{<6 parameters>} % boilerplate
\expressghead               % Annex C: EXPRESS-G diagrams
  \aicexpressg               % boilerplate
\listingshead                  % Annex D: Computer interpretable listings
  \expurls                     % boilerplate
\techdischead                  % optional Annex: Technical discussions
```

B.4 Body of an ATS document

The body of an Abstract Test Suite document has the following structure:

```
\purposeshead                  % Clause 4: Test purposes
  \atspurposebp    % boilerplate
  \aepurposehead               % 4.1 Application element test purposes
    \aetpbp                    % boilerplate
    \apobjhead{<object>}       % 4.1.n
    ...
  \aimpurposehead              % 4.2 AIM test purposes
    \aimtpbp                   % boilerplate
    \aimenthead{<entity>}      % 4.2.n
    ...
  \implementpurposehead        % (optional) 4.3 Implementation t.p
    \atsimtpbp                 % boilerplate
    % text
  \domainpurposehead           % (optional) 4.2+ Domain test purposes
    % text
  \otherpurposehead            % (optional) 4.2+ Other test purposes
    % text
\gtpvchead                     % Clause 5: General t.p and verdict criteria
  \atsgtpvcbp                  % boilerplate
  \generalpurposehead          % 5.1 General test purposes
    \gtpbp                     % boilerplate
    ...
  \gvcatchead                  % 5.2 General verdict criteria for all ATC
    \gvatcbp                   % boilerplate
    ...
  \gvcprehead                  % 5.3 General verdict criteria for preprocessor
    \gvcprebp                  % boilerplate
     ...
  \gvcposthead                 % 5.4 General verdict criteria for postprocessor
    \gvcpostbp                 % boilerplate
    ...
\atchead                       % Clause 6: Abstract test cases
  \atcbp                       % boilerplate (para 1)
  % your para 2
  \atcbpii                     % boilerplate (paras 3+)
  \atctitlehead{<title>}       % 6.n an abstract test case
    \prehead                   % 6.n.1 Preprocessor
      \precoveredhead..        % Test purposes covered
        \atcpretpc             % boilerplate
      \preinputhead            % Input specification
        % text
      \precriteriahead         % Verdict criteria
        % text
      \preconstrainthead       % Constraints on values
        % text
      \preexechead             % (optional) Execution sequence
        % text
      \preextrahead            % (optional) Extra details
```



```
        % text
    \posthead                   % 6.n.2 Postprocessor
      \postcoveredhead          % Test purposes covered
        % text
        \atcposttpc             % boilerplate
      \postinputhead            % Input specification
        % text
      \postcriteriahead         % Verdict criteria
        % text
      \postexechead             % (optional) Execution sequence
        % text
      \postextra                % (optional) Extra details
        % text
\confclassannexhead             % Annex A: Conformance classes
  \atsnoclassesbp               % boilerplate if no conformance classes, else
  \confclasshead{<number>}      % A.n Conformance class <number>
    \confclassbp{<number>}      % boilerplate
    % text
  ...
\postipfilehead                 % Annex B: Postprocessor input file names
  \pisfbp{..}{..}{..}                       % boilerplate
   ...
\objreghead                     % Annex C: Information object registration
  \docreg{<partno>}             % registration boilerplate
\atsusagehead                   % Annex D: Usage scenarios
  % text
```


Annex C
(normative)
Information object registration

To provide for unambiguous identification of an information object in an open system, the object identifier

{ iso standard 10303 part(3456) version(-1) }

is assigned to this part of ISO 10303. The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Annex D
(informative)
Deprecated, deleted, new and modified commands

This release has involved many internal changes to the LaTeX .sty files. In particular boilerplate text is, as far as possible, maintained in external .tex files in order to save memory space within the LaTeX processor.

D.1 New commands

The commands that are new in this release are:

— `\bibieeeidefo` STEP: reference to IDEF0 document;

— `\exampleshead` STEP: clause heading;

— `\expressgdef` STEP: location of EXPRESS-G definition;

— `\Theseries` STEP: print `\series` argument;

— `\theseries` STEP: print `\series` argument in lowercase;

— `\ifanir`, `\anirtrue`, `\anirfalse` STEP: flag for an IR document;

— `\ifhaspatents`, `\haspatentstrue`, `\haspatentsfalse` STEP: flag for known patents;

— `\ifmapspec`, `\mapspectrue`, `\mapspecfalse` STEP: flag for mapping specification;

— `\ixent` STEP: index an EXPRESS ENTITY;

— `\ixenum` STEP: index an EXPRESS ENUMERATION;

— `\ixfun` STEP: index an EXPRESS FUNCTION;

— `\ixproc` STEP: index an EXPRESS PROCEDURE;

— `\ixrule` STEP: index an EXPRESS RULE;

— `\ixsc` STEP: index an EXPRESS SUBTYPE_CONSTRAINT;

— `\ixschema` STEP: index an EXPRESS SCHEMA;

— `\ixselect` STEP: index an EXPRESS SELECT;

— `\ixtype` STEP: index an EXPRESS TYPE;

— `\maptableorspec` STEP: prints 'table' or 'specification';

— `\xword` STEP: prints an EXPRESS keyword;

— `\apmaptemplate` AP: boilerplate;

- `\apusagehead` AP: clause heading;
- `\ifidefix`, `\idefixtrue`, `\idefixfalse` AP: flag for an IDEF1X ARM;
- `\ifmaptemplate`, `\maptemplatetrue`, `\maptemplatefalse` AP: flag for using mapping templates;
- `\mapattributehead` AP: clause heading;
- `\mapobjecthead` AP: clause heading;
- `\mapuofhead` AP: clause heading;
- `\sstemplates` AP: boilerplate;
- `\templateshead` AP: clause heading;
- `\atcposttpc` ATS: boilerplate;
- `\atcpretpc` ATS: boilerplate;
- `\atsimtpbp` ATS: boilerplate;
- `\atsusagehead` ATS: clause heading.

D.2 Modified commands

The commands that have been modified in this release are:

- STEP: The `\Introduction` command is now the `Introduction` environment, with no argument;
- `\irexpressg` IR: takes no argument;
- `\aimexpressg` AP: takes no argument;
- `\aicexpressg` AIC: takes no argument;
- `\atcbpii` ATS: takes no argument;
- `\atspurposebp` ATS: takes no argument;
- `\pisfbp` ATS: takes 3 arguments.

D.3 Deleted commands

The commands that have been deleted in this release are:

- `\fwddivlist` STEP: used in Foreword;
- `\fwdpartslist` STEP: used in Foreword;
- `\introend` STEP: was for use at the end of the Introduction;

- `\irschemaintro` IR: has been replaced by `\schemaintro`;
- `\apintroend` AP: has been replaced by `\apextraintro`;
- `\apschemareg` AP: use `\schemareg` instead;
- `\apmappingtable` AP: has been replaced by `\apmapping`;
- `\armfigures` AP: has been replaced by `\armintro`;
- `\maptablehead` AP: has been replaced by `\mappinghead`;
- `\modelscopehead` AP: was the heading for a 'Model scope' annex;
- `\aicexpressghead` AIC: use `\expressghead` instead;
- `\aicshortnames` AIC: use `\expurls` instead;
- `\aicshortnameshead` AIC: use `\shortnamehead` instead;
- `\excludepurposehead` ATS: was the heading for an 'Exclude purposes' clause.

Annex E
(informative)
LaTeX, the Web, and *ML

ISO are becoming more interested in electronic sources for their standards as well as the traditional camera-ready copy. Acronyms like PDF, HTML, SGML and XML have been bandied about. Fortunately documents written using LaTeX are well placed to be provided in a variety of electronic formats. A comprehensive treatment of LaTeX with respect to this topic is provided by Goossens and Rahtz [4].

SGML (Standard Generalized Markup Language) is a document tagging language that is described in ISO 8879 [6] and whose usage is described in [8], among others. The principal mover behind SGML is Charles Goldfarb from IBM, who has authored a detailed handbook [7] on the SGML standard.

The concepts lying behind both LaTeX and SGML are similar, but on the face of it they are distinctly different in both syntax and capabilities. ISO is migrating towards electronic versions of its standard documents and, naturally, would prefer these to be SGML tagged. Like LaTeX, SGML has a concept of style files, which are termed DTDs, and both systems support powerful macro-like capabilities. SGML provides for logical document markup and not typesetting — commercial SGML systems often use TeX or LaTeX as their printing engine, as does the NIST SGML environment for ISO 10303 [9].

NIST have SGML tagged some STEP documents using manual methods, which are time consuming and expensive. In about 1997 there was a NIST effort underway to develop an auto-tagger that would (semi-) automatically convert a LaTeX tagged document to one with SGML tags. This tool assumed a fixed set of LaTeX macros and a fixed DTD. The design of an auto-tagger essentially boils down to being able to convert from a source document tagged according to a LaTeX style file to one which is tagged according to an SGML DTD. Fully automatic conversion is really only possible if the authors' of the documents to be translated avoid using any 'non-standard' macros within their documents. There is a program called ltx2x available from SOLIS, which replaces LaTeX commands within a document with user-defined text strings [10]. This can be used as a basis for a LaTeX to whatever auto-tagger, provided the LaTeX commands are not too exotic.

HTML is a simple markup language, based on SGML, and is used for the publication of many documents on the Web. XML is a subset of SGML and appears to being taken up by every man and his dog as *the* document markup language. HTML is being recast in terms of XML instead of SGML. PDF is a page description language that is a popular format for display of documents on the Web.

LaTeX documents can be output in PDF by using pdfLaTeX. Instead of a .dvi file being produced a .pdf file is output directly. The best results are obtained when PostScript fonts rather than Knuth's cm fonts are used. Noting that the isov2 class provides an `\ifpdf` command, a general form for documents to be processed by either LaTeX or pdfLaTeX is

```
\documentclass{isov2}
\usepackage{times}     % PostScript fonts Times, Courier, Helvetica
\ifpdf
  \pdfoutput=1         % request PDF output
  \usepackage[pdftex]{graphicx}
\else
```



```
  \usepackage{graphicx}
\fi
...
```

There are several converters available to transform a LaTeX document into an HTML document, but like ltx2x they generally do their own parsing of the source file, and unlike ltx2x are typically limited to only generating HTML. Eitan Gurari's TeX4ht suite is a notable exception (see Chapter 4 and Appendix B of [4]). It uses the .dvi file as input, so that all the parsing is done by TeX, and can be configured to generate a wide variety of output formats. A set of TeX4ht configuration files are available for converting STEP LaTeX documents into HTML[4)].

It is highly recommended that for the purposes of ISO 10303, document editors refrain from defining their own LaTeX macros. If new generally applicable LaTeX commands are found to be necessary, these should be sent to the editor of this document for incorporation into the isov2 class, the step package and/or appropriate other packages.

Some other points to watch when writing LaTeX documents that will assist in translations into *ML are given below. Typically, attention to these points will make it easier to parse the LaTeX source.

— Avoid using the `\label` command within clause headings or captions. It can just as easily be placed immediately after these constructs.

— Avoid using the `\index` command within clause headings or captions. It can just as easily be placed immediately after these constructs.

— Use all the specified tagging constructs when defining an EXPRESS model — this will also assist any program that attempts to extract EXPRESS source code and descriptive text from a document.

[4)] Later, configuration files for XML output will be developed.

Annex F
(informative)
Obtaining LaTeX and friends

LaTeX is a freely available document typesetting system. There are many public domain additions to the basic system, for example the iso.cls and step.sty styles. The information below gives pointers to where you can obtain LaTeX etc., from the Internet.

LaTeX runs on a wide variety of hardware, from PCs to Crays. Source to build a LaTeX system is freely available via anonymous ftp from what is called CTAN (Comprehensive TeX Archive Network). There are three sites; pick the one nearest to you.

— `ftp.dante.de` CTAN in Germany;

— `ftp.tex.ac.uk` CTAN in the UK;

— `ctan.tug.org` CTAN in the USA;

The top level CTAN directory for LaTeX and friends is `/tex-archive`. CTAN contains a wide variety of (La)TeX sources, style files, and software tools and scripts to assist in document processing.

NOTE CTAN is maintained by the TeX Users Group (TUG). Their homepage `<http://www.tug.org>` should be consulted for the current list of CTAN sites and mirrors.

Bibliography

[1] LAMPORT, L., *LaTeX — A Document Preparation System,* Addison-Wesley Publishing Co., 2nd edition, 1994

[2] WILSON, P. R., *LaTeX for standards: The LaTeX package files user manual,* NISTIR, National Institute of Standards and Technology, Gaithersburg, MD 20899. June 1996.

[3] GOOSENS, M., MITTELBACH, F. and SAMARIN, A., *The LaTeX Companion,* Addison-Wesley Publishing Co., 1994

[4] GOOSENS, M. and RAHTZ, S., *The LaTeX Web Companion — Integrating TeX, HTML, and XML,* Addison-Wesley Publishing Co., 1999

[5] CHEN, P. and HARRISON, M.A., *Index preparation and processing,* Software–Practice and Experience, 19(9):897–915, September 1988.

[6] ISO 8879:1986, *Information processing — Text and office systems — Standard Generalized Markup Language (SGML)*

[7] GOLDFARB, C.F., *The SGML Handbook,* Oxford University Press, 1990.

[8] BRYAN, M., *SGML — An Author's Guide to the Standard Generalized Markup Language,* Addison-Wesley Publishing Co., 1988.

[9] PHILLIPS, L., and LUBELL, J., *An SGML Environment for STEP,* NISTIR 5515, National Institute of Standards and Technology, Gaithersburg, MD 20899. November 1994.

[10] WILSON, P. R., *LTX2X: A LaTeX to X Auto-tagger,* NISTIR, National Institute of Standards and Technology, Gaithersburg, MD 20899. June 1996.

[11] *IDEF0 (ICAM Definition Language 0),* Federal Information Processing Standards Publication 183, Integration Definition for Information Modeling (IDEF0), FIPS PUB 183, National Institute for Standards and Technology, December 1993.

[12] IEEE Std 1320.2–1998, *Standard for Conceptual Modeling Language — Syntax and Semantics for IDEF1X.*

Index

