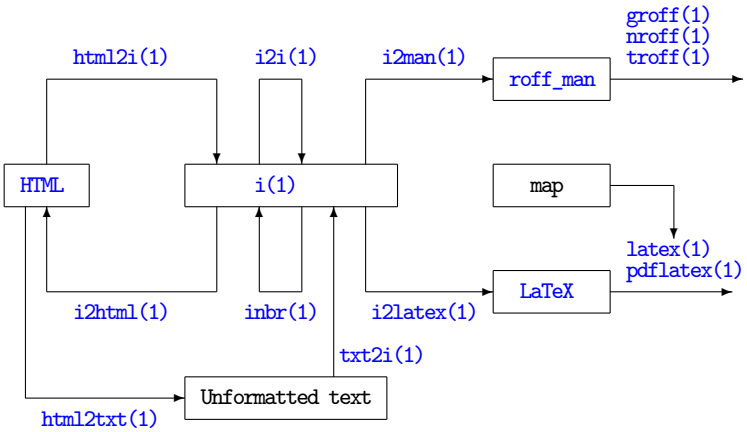
i(1)

2012-04-07 1.2

1 Introduction

This document is part of the `i` project which is available on–line at the site `http://i2i.sourceforge.net`.

1.1 Concepts of `i(1)`



The syntax of `i` (first roman numeral – pronounced *unus*) is a formal way to structure textual content. `i(1)` is designed with the following considerations:

- Documentation written in `i(1)` can be read by everybody without much difficulty. The text's structuring elements appear in a natural way. The reader's eye is hardly disturbed by syntactical ingredients. The `i(1)` syntax is formal, yet easy to read at the same time!
- Formatting is automated by piping the `i(1)` content through the `i2i(1)` or `inbr(1)` utilities. Editing should therefore best be carried out within a tool that allows plug–ins. An undo–function may also prove handy.
- `i(1)` can be used for short notes as well as for extensive documentation. Everything that is written in `i(1)` has the advantage that it can be distributed either directly (e.g. in an e–mail) or after being processed through the following tool chain:
  - `i2man(1)` transforms input written in `i(1)` into *roff_man*, an output format that is understood by the `groff(1)`, `troff(1)`, `nroff(1)`, etc. utility family.
  - `i2latex(1)` transforms input written in `i(1)` into *LaTeX*, an output format that is understood by the `latex(1)`, `pdflatex(1)`, etc. utility family.
- Documentation written in `i(1)` can be fed into any spelling or grammar checker that can deal with *HTML*. The `i2html(1)` and `html2i(1)` utilities are used for the conversion. Alternatively and highly recommended, the `ispegra(1)` utility can be used for the same purpose (combines the `i2html(1)`, `html2txt(1)` and `txt2i(1)` utilities).
- `i(1)` might be single octet encoded (*ISO–8859–1*, *ISO–8859–5*, *ISO–8859–7*, etc.) or varying octet number encoded (*UTF–8*). Option `-u` needs to be added in order to indicate that the `i(1)` input is *UTF–8* encoded.
- The tool chain around `i(1)` merely covers functionality that has not already been implemented elsewhere by utilities like `cvs(1)`, `diff(1)`, `make(1)`, `rcs(1)` and `vi(1)`.

1.2 Building blocks

Textual content written in i(1) is constructed with the following elements:

- At the lowest level, the entire textual content is a sequence of bytes. Some bytes have a special meaning (see table 2). Every byte is in a mode (see chapter 3).
- At the second level, the entire textual content is a sequence of tokens. Tokens are composed of the following three parts (the first or the second part may be missing):
  - The first part contains any number of I_BYTE_CARRIAGERETURN or I_BYTE_NEWLINE characters.
  - The second part contains any number of I_BYTE_SPACE characters.
  - The third part contains any number of bytes which are neither an I_BYTE_CARRIAGERETURN, an I_BYTE_NEWLINE nor an I_BYTE_SPACE character.

  Table 3 gives a complete list of all i(1) tokens.
- At the third level, every token belongs to one of the following items:
  - heading items (see chapter 2.1)
  - caption items (see chapter 2.2)
  - bibliography and equation items (see chapter 2.3)
  - paragraph items (see chapter 2.4)
  - list items (see chapter 2.5)
  - quotation items (see chapter 2.6)
  - pre–formatted items (see chapter 2.7)
  - picture items (see chapter 2.8)
  - latex items (see chapter 2.9)
  - man items (see chapter 2.10)
  - interrupt items (see chapter 2.11)

  Items start and continue with a characteristic set of tokens (see table 4). Items may contain footnotes (see chapter 2.12).
- At the top level, items are arranged one– or two–dimensionally. A one–dimensional arrangement means that the items follow each other sequentially in the textual content. A two–dimensional arrangement means that the items of a column alternate with the ones of other columns (refer to chapter 4.2 for more details about tables). One–dimensional textual content can contain all kinds of items. Two–dimensional content can merely contain paragraph, list and quotation items.

2 Items

2.1 Heading items (I_ITEM_H1 to I_ITEM_H5)

Heading items structure the textual content and give a brief summary about what follows next. All heading items together form the basis from which a table of contents (TOC) may be derived.

Heading items start on a new line without any indentation. After the first token, heading items continue the same way as paragraph items (see chapter 2.4).

There are five levels of heading items. I_ITEM_H1 is the least and I_ITEM_H5 the most nested level. I_ITEM_H1, I_ITEM_H2, I_ITEM_H3 and I_ITEM_H4 are numbered heading items. I_ITEM_H5 items are without numbering. Numbering entities may appear in heading items in the following different forms (refer to chapter 2.1.1 for examples):

- In the final form, the complete numbering is present.
- In the neutral form, there are only I_BYTE_SPAN1 characters and the digits are omitted (see table 1).
- In the decrement form, the numbering is in the final form but marked by an I_BYTE_NBR_DECR character to be lowered to the next nesting level.

- In the increment form, the numbering is in the final form but marked by an I_BYTE_NBR_INCR character to be raised to the next nesting level.

The inbr(1) utility is used to get numbering entities into their final form.

2.1.1 Example

```
# Neutral heading one
#. Neutral heading two
#.. Neutral heading three
#... Neutral heading four
Heading five
#1 Final heading one
#1.1 Final heading two
#1.1.1 Final heading three
#1.1.1.1 Final heading four
#2> Heading one to be raised to a heading two
#2.1> Heading two to be raised to a heading three
#2.1.1> Heading three to be raised to a heading four
#3.1< Heading two to be lowered to a heading one
#3.1.1< Heading three to be lowered to a heading two
#3.1.1.1< Heading four to be lowered to a heading three
```

2.2 Caption items (I_ITEM_HFIG and I_ITEM_HTAB)

Caption items give a brief summary about the content of tables or figures. All caption items together form the basis from which the list of tables (LOT) and the list of figures (LOF) are derived.

Caption items start on a new line without any indentation. After the first token, caption items continue the same way as paragraph items (see chapter 2.4).

Numbering entities may appear in caption items in the following different forms (refer to chapter 2.2.1 for examples):

- In the final form, the complete numbering is present.
- In the neutral form, the digits are completely missing (see also table 1).

The inbr(1) utility is used to get numbering entities into their final form.

2.2.1 Example

```
#T Neutral table caption
#F Neutral figure caption
#T1 Table caption
#F1 Figure caption
```

2.3 Bibliography and equation items (I_ITEM_HBIB and I_ITEM_HEQU)

Bibliography and equation items are numbered items.

Bibliography and equation items start on a new line without any indentation. After the first token, bibliography and equation items continue the same way as paragraph items (see chapter 2.4).

Numbering entities may appear in bibliography and equation items in the following different forms (refer to chapter 2.3.1 for examples):

- In the final form, the complete numbering is present.
- In the neutral form, the digits are completely absent (see also table 1).

The inbr(1) utility is used to get numbering entities into their final form.

2.3.1 Example

```
#B Neutral bibliography
#B1 Neutral bibliography
#E Neutral equation
#E1 Final equation
```

2.4 Paragraph items (I_ITEM__PAR)

Paragraph items contain ordinary text sequences. Paragraph items start with either a two–space token (I_TOK__BR or I_TOK_BR) or a one–space token (I_TOK_PAR). In exceptional cases (at the beginning of the document, after tables, after pre–formatted, picture, latex and man items), paragraph items may start with an I_TOK__ELSE or I_TOK_ELSE token. After the start token, paragraph items continue for any number of I_TOK__ELSE or I_TOK_ELSE tokens.

2.5 List items (I_ITEM__LI1 and I_ITEM__LI2)

List items are used where different aspects of the same topic need to be distinguished. There are two levels of list items:

- I_ITEM__LI1 items start with a two–space dash (I_TOK__LI1 or I_TOK_LI1 token) or a five–space token (I_TOK__BR1 or I_TOK_BR1) and continue the same way as paragraph items (see chapter 2.4).
- I_ITEM__LI2 items start with a five–space dash (I_TOK__LI2 or I_TOK_LI2 token) and continue the same way as paragraph items (see chapter 2.4).

2.5.1 Example

```
paragraph
- list
- list
   list
- list
   - list one
   - list one
paragraph
```

2.6 Quotation items (I_ITEM__QUOTE)

Quotation items are used where text sequences come directly from another source. Quotation items start with an eight–space token (I_TOK__QUOTE or I_TOK_QUOTE) and continue the same way as paragraph items (see chapter 2.4).

2.6.1 Example

```
paragraph
      quotation
paragraph
```

2.7 Pre–formatted items (I_ITEM_PRE and I_ITEM_PRE1)

Pre–formatted items are used where the number of I_BYTE_SPACE characters shall be retained. There are two levels of pre–formatted items:

- I_ITEM_PRE items start with a not–four–space vertical line (I_TOK__VLINE, I_TOK_-VLINE, I_TOK__VLINE_ or I_TOK_VLINE_ token) and continue for as many lines as they start with an I_TOK_VLINE token.

- I_ITEM_PRE1 items start with a four–space vertical line (I_TOK__VLINE1, I_TOK_-VLINE1, I_TOK__VLINE1_ or I_TOK_VLINE1_ token) and continue for as many lines as they start with an I_TOK_VLINE1 token.

2.7.1 Example

```
; for( i = 0; i < k; i++ ) {
:   printf("i=%d\n", i);
: }
```

2.8 Picture items (I_ITEM_PIC)

Picture items contain ASCII graphics and may include the following building blocks:

- text elements
- horizontal and vertical line elements
- arrows elements
- oval elements

With the exception of the i2latex(1) utility, the content of picture items is treated exactly the same way as that of pre–formatted items (see chapter 2.7). The number of I_BYTE_-SPACE characters is retained. Picture items start with an I_TOK__PIC or I_TOK_PIC token and continue for as many lines as they start with an I_TOK_VLINE token.

2.8.1 Example

```
, +-----+      /-----\
: | abc +----->| def +----->
: +-----+      \-----/
```

2.9 Latex items (I_ITEM_LATEX)

Latex items contain information which is understood by the latex(1), pdflatex(1), etc. utility family. The number of I_BYTE_SPACE characters is retained. Latex items start with an I_TOK_LATEX token and continue for as many lines as they start with an I_TOK_LATEX token.

2.10 Man items (I_ITEM_MAN)

Man items contain information which is understood by the groff(1), troff(1), nroff(1), etc. utility family. The number of I_BYTE_SPACE characters is retained. Man items start with an I_TOK_MAN token and continue for as many lines as they start with an I_TOK_MAN token.

2.11 Interrupt items (I_ITEM_INTR)

The i2i(1), inbr(1) and i2html(1) utilities process interrupt items transparently. The i2latex(1) and i2man(1) utilities suppress the usual output generation after an interrupt item and hand control over to the subsequent latex and man items. It is important to understand that i2latex(1) and i2man(1) output is syntactically incomplete after an interrupt item! Example 2.11.1 shows how a caption item is interrupted in order to continue with pure *LaTeX* and *roff_man* syntax. In the i2latex(1) output, an external graphics file is included. In i2man(1) output the name of the graphics file (in brackets) is mentioned.

2.11.1 Example

```
#F1 Photo
#I
#L }\includegraphics[scale=0.25]{photo.jpg}\end{ifigure}
#M  [ photo.jpg ]
#M .fi
```

2.12 Footnotes

Footnotes are sub–items. Footnotes cannot start in one item and end in the next. Footnotes begin with an I_TOK__ELSE or I_TOK_ELSE token, both of which start with an I_BYTE_NBR and I_BYTE_FN_BEG character sequence. Footnotes end with an I_TOK__ELSE or I_TOK_ELSE token, which contains an I_BYTE_NBR and I_BYTE_FN_END character sequence. Only the following items can contain footnotes:

- heading items (see chapter 2.1)
- caption items (see chapter 2.2)
- bibliography and equation items (see chapter 2.3)
- paragraph items (see chapter 2.4)
- list items (see chapter 2.5)
- quotation items (see chapter 2.6)

2.12.1 Example

```
        He was an old man who fished alone in a skiff in the Gulf
       Stream and he had gone eighty-four days now without taking
       a fish. #(opening of #B1#)
#B1 Hemingway, Ernest: The old man and the sea. 1952.
```

3 Modes

Every byte of the textual content is in a mode. The mode might change in the middle of a token. The following three aspects influence the mode:

- The item which the byte is part of.
- The token which the byte is part of.
- The presence of particular characters in the third part of the token.

3.1 Latex mode

All bytes of latex items (see chapter 2.9) are in this mode. Their particular treatment depends on the utility (i2i(1), inbr(1), i2html(1), i2latex(1) or i2man(1)).

3.2 Man mode

All bytes of man items (see chapter 2.10) are in this mode. Their particular treatment depends on the utility (i2i(1), inbr(1), i2html(1), i2latex(1) or i2man(1)).

3.3 Number anchor mode

All bytes of I_TOK_H1, I_TOK_H2, I_TOK_H3, I_TOK_H4, I_TOK_HBIB, I_TOK_HEQU, I_TOK_HFIG and I_TOK_HTAB tokens are in this mode. The cumulative content of all these tokens forms a document–wide numbering scheme. Each of these tokens represents a numbered destination (anchor) of the textual content and therefore needs to be unique. Refer to chapters 2.1.1, 2.2.1 and 2.3.1 in order to find examples of bytes in number anchor

mode. The inbr(1) utility is used for updating byte sequences in number anchor mode. Table 1 gives a complete list of bytes in number anchor mode in their neutral form. Bytes in anchor mode are defined to use normal font style.

Table 1: Number anchor entities

Item	Token	Neutral form
I_ITEM_H1	I_TOK_H1	#
I_ITEM_H2	I_TOK_H2	#.
I_ITEM_H3	I_TOK_H3	#..
I_ITEM_H4	I_TOK_H4	#...
I_ITEM_HFIG	I_TOK_HFIG	#F
I_ITEM_HTAB	I_TOK_HTAB	#T
I_ITEM_HBIB	I_TOK_HBIB	#B
I_ITEM_HEQU	I_TOK_HEQU	#E

3.4 Fixed–width font anchor mode

Only the tokens of heading, caption, bibliography, equation, paragraph, list and quotation items may contain bytes in fixed–width font anchor mode. Such bytes are separated from the rest of the token by a leading I_BYTE_CUT2 and a terminating I_BYTE_CUT1 character. The separation is done in such a manner that the rest of the token contains neither an I_BYTE_CUT1 nor I_BYTE_CUT2 character. Byte sequences in fixed–width font anchor mode represent a destination of the document–wide reference scheme and therefore need to be unique. They are defined to use fixed–width font style.

3.4.1 Example

```
#1 Heading ‘ONE’
Heading ‘TWO’
  Paragraph T‘HRE’E
```

3.5 Italic font anchor mode

Only the tokens of heading, caption, bibliography, equation, paragraph, list and quotation items may contain bytes in italic font anchor mode. Such bytes are separated from the rest of the token by a leading I_BYTE_CUT1 and a terminating I_BYTE_CUT2 character. The separation is done in such a manner that the rest of the token contains neither an I_BYTE_CUT1 nor I_BYTE_CUT2 character. Byte sequences in italic font anchor mode represent a destination of the document–wide reference scheme and therefore need to be unique. They are defined to use italic font style.

3.5.1 Example

```
#1 Heading ’one‘
Heading ’two‘
  Paragraph t’hre‘e
```

3.6 Fixed–width font mode

Firstly, all tokens of pre–formatted and picture items are in fixed–width font mode without exception.

Secondly, all tokens of heading, caption, bibliography, equation, paragraph, list and quotation items may contain bytes in fixed–width font mode. Such bytes are separated

from the rest of the token by a leading I_BYTE_CUT2 and a terminating I_BYTE_CUT2 character. The separation is done in such a manner that the rest of the token contains neither an I_BYTE_CUT1 nor I_BYTE_CUT2 character.

All content which is in fixed–width font mode is defined to use fixed–width font style. All byte sequences in fixed–width font mode that are identical to byte sequences in fixed–width or italic font anchor mode (see chapter 3.4 and 3.5) are automatically resolved with the i2latex(1) utility.

3.6.1 Example

```
 Paragraph (‘ONE‘, ‘one‘, ‘TWO‘, ‘two‘ ‘THREE‘
or ‘three‘)
```

3.7 Italic font mode

Only the tokens of heading, caption, bibliography, equation, paragraph, list and quotation items may contain bytes in italic font mode. Such bytes are separated from the rest of the token by a leading I_BYTE_CUT1 and a terminating I_BYTE_CUT1 character. The separation is done in such a manner that the rest of the token contains neither an I_BYTE_CUT1 nor I_BYTE_CUT2 character. The content in italic font mode is defined to use italic font style. All byte sequences in italic font mode that are identical to byte sequences in fixed–width or italic font anchor mode (see chapter 3.4 and 3.5) are automatically resolved with the i2latex(1) utility.

3.7.1 Example

```
 Paragraph ('ONE', 'one', 'TWO', 'two' 'THREE'
or 'three')
```

3.8 Default mode

Bytes are merely in default mode if none of the modes described in chapters 3.1 to 3.7 are suitable. This presumably applies to the remaining bytes of tokens which have been cut by I_BYTE_CUT1 and I_BYTE_CUT2 characters (see chapters 3.4, 3.5, 3.6 and 3.7). The content of such a byte sequence is defined to use normal font style. All byte sequences in default mode that are identical to byte sequences in number anchor mode (see chapter 3.3) are automatically resolved with the i2latex(1) utility (the inbr(1) utility is used for their update).

3.8.1 Example

```
Refer to #1, #2, #2.1, #2.2, #B1 or #B2
```

4 Arrangement

4.1 One–dimensional arrangement

In one–dimensionally arranged textual content, the items follow each other sequentially. Such content can contain all kinds of items (see chapter 2.1 to 2.11). Footnotes are supported, too (see chapter 2.12).

4.2 Two–dimensional arrangement (tables)

In two–dimensionally arranged textual content, items of different columns alternate with the ones of other columns. Such content can merely contain paragraph, list and quotation items (see chapters 2.4. to 2.6). Footnotes are supported, too (see chapter 2.12).

Tables start with a horizontal line (I_TOK_HLINE). Horizontal line tokens may include the following bytes:

- I_BYTE_VLINE characters indicate the start position of each column (as soon as an I_BYTE_VLINE character is changed to an I_BYTE_VLINE_ character, the column is marked to be removed by the i2i(1) utility).
- I_BYTE_SPAN1 characters indicate positions where two columns are merged together as multi–columns (as soon as an I_BYTE_SPAN1 character is changed to an I_BYTE_SPAN2 character, the column is marked to be removed by the i2i(1) utility).
- I_BYTE_ADD characters indicate a position where a new column is to be added by the i2i(1) utility.
- I_BYTE_HLINE1 or I_BYTE_HLINE2 characters indicate the width of the columns. I_BYTE_HLINE1 and I_BYTE_HLINE2 characters are mutually exclusive and must not be used in the same I_TOK_HLINE token (as soon as both are present, the token is regarded as an I_TOK_H5 token.

The first horizontal line of a table defines the number of columns and their width for the rest of the table. Tables continue for as many lines as the following rules are fulfilled:

- The line contains another horizontal line which corresponds to the table's first I_TOK_HLINE token in the number of columns. Such subsequent horizontal lines indicate the start of a new table row. The first horizontal line of a table with I_BYTE_HLINE2 characters marks the end of the table head to begin the table body. As soon as the horizontal line contains I_BYTE_ADD, I_BYTE_VLINE_ or I_BYTE_SPAN2 characters, a new table is started (only the very first horizontal line of a table can contain these modifier bytes).
- The line contains as many columns as indicated by the preceding horizontal line. The first column starts with an I_TOK_VLINE token. Each column is separated from its neighbour by an I_TOK__VLINE or I_TOK__VLINE1 token.

4.2.1 Example of a table without a head

```
:======:=====:==========
: one  : two : three
:------:-----.----------
: four : five
```

4.2.2 Example of a table with a head

```
:------:-----:----------
: one  : two : three
:======:=====.==========
: four : five
```

4.2.3 Example of a table where the middle column is to be removed

```
:------;-----:----------
: one  : two : three
:======:=====.==========
: four : five
```

4.2.4 Example of a table with an additional column

```
:------+---:-----:----------
: one  : two : three
:======:=====.==========
: four : five
```

4.2.5 Example of a table with various items

```
:-----------:-----:----------
: one        : two : three
:===========:=====.==========
: pre        : pre
:  - list    :        quote
:  - list    :
```

5 i(1) Reference

Table 2: Bytes with a special meaning

	Hex	*ISO–8859–1* and *UTF–8* representation
I_BYTE_ADD	0x2B	+
I_BYTE_CARRIAGERETURN	0x0D	
I_BYTE_CUT1	0x27	’
I_BYTE_CUT2	0x60	‘
I_BYTE_END	0x00	
I_BYTE_FN_BEG	0x28	(
I_BYTE_FN_END	0x29	)
I_BYTE_HLINE1	0x2D	-
I_BYTE_HLINE2	0x3D	=
I_BYTE_SPAN1	0x2E	.
I_BYTE_SPAN2	0x2C	,
I_BYTE_INTR	0x49	I
I_BYTE_LATEX	0x4C	L
I_BYTE_LI	0x2D	-
I_BYTE_MAN	0x4D	M
I_BYTE_NBR	0x23	#
I_BYTE_HBIB	0x42	B
I_BYTE_NBR_DECR	0x3C	<
I_BYTE_HEQU	0x45	E
I_BYTE_HFIG	0x46	F
I_BYTE_NBR_INCR	0x3E	>
I_BYTE_HTAB	0x54	T
I_BYTE_NEWLINE	0x0A	
I_BYTE_SPACE	0x20	
I_BYTE_NBSPACE	0xA0	
I_BYTE_VLINE	0x3A	:
I_BYTE_VLINE_	0x3B	;

Table 3: Tokens

Token	Number of new lines in first part	Number of spaces in second part	Third part: definitions–list according to *ISO–14977*
I_TOK_H1	any	= 0	I_BYTE_NBR [NBR] [I_BYTE_NBR_INCR]
I_TOK_H2	any	= 0	I_BYTE_NBR [NBR] I_BYTE_SPAN1 [NBR] [(I_BYTE_NBR_INCR | I_BYTE_NBR_DECR)]
I_TOK_H3	any	= 0	I_BYTE_NBR [NBR] I_BYTE_SPAN1 [NBR] I_BYTE_SPAN1 [NBR] [(I_BYTE_NBR_INCR | I_BYTE_NBR_DECR)]
I_TOK_H4	any	= 0	I_BYTE_NBR [NBR] I_BYTE_SPAN1 [NBR] I_BYTE_SPAN1 [NBR] I_BYTE_SPAN1 [NBR] [I_BYTE_NBR_DECR]
I_TOK_HBIB	any	= 0	I_BYTE_NBR I_BYTE_HBIB [NBR]
I_TOK_HEQU	any	= 0	I_BYTE_NBR I_BYTE_HEQU [NBR]
I_TOK_HFIG	any	= 0	I_BYTE_NBR I_BYTE_HFIG [NBR]
I_TOK_HTAB	any	= 0	I_BYTE_NBR I_BYTE_HTAB [NBR]
I_TOK_INTR	any	= 0	I_BYTE_NBR I_BYTE_INTR
I_TOK_MAN	any	= 0	I_BYTE_NBR I_BYTE_MAN
I_TOK_LATEX	any	= 0	I_BYTE_NBR I_BYTE_LATEX
I_TOK_HLINE	any	= 0	(* see chapter 4.2 *)
I_TOK_H5	any	= 0	– (I_TOK_H1 | I_TOK_H2 | I_TOK_H3 | I_TOK_H4 | I_TOK_HBIB | I_TOK_HEQU | I_TOK_HFIG | I_TOK_HTAB | I_TOK_INTR | I_TOK_MAN | I_TOK_LATEX | I_TOK_HLINE)
I_TOK_VLINE1	> 0	= 4	I_BYTE_VLINE
I_TOK__VLINE1	= 0	= 4	I_BYTE_VLINE
I_TOK_VLINE1_	> 0	= 4	I_BYTE_VLINE_
I_TOK__VLINE1_	= 0	= 4	I_BYTE_VLINE_
I_TOK_VLINE	> 0	!= 4	I_BYTE_VLINE
I_TOK__VLINE	= 0	!= 4	I_BYTE_VLINE
I_TOK_VLINE_	> 0	!= 4	I_BYTE_VLINE_
I_TOK__VLINE_	= 0	!= 4	I_BYTE_VLINE_
I_TOK_PIC	> 0	any	I_BYTE_SPAN2
I_TOK__PIC	= 0	any	I_BYTE_SPAN2
I_TOK_PAR	> 1	= 1	– (I_TOK_VLINE | I_TOK_VLINE_ | I_TOK_PIC)
I_TOK_LI1	> 0	= 2	I_BYTE_LI
I_TOK__LI1	= 0	= 2	I_BYTE_LI
I_TOK_BR	> 0	= 2	– (I_TOK_VLINE | I_TOK_VLINE_ | I_TOK_PIC | I_TOK_LI1)

Token	Number of new lines in first part	Number of spaces in second part	Third part: definitions–list according to *ISO–14977*
I_TOK__BR	= 0	= 2	– (I_TOK__VLINE | I_TOK__VLINE_ | I_TOK__PIC | I_TOK__LI1)
I_TOK_LI2	> 0	= 5	I_BYTE_LI
I_TOK__LI2	= 0	= 5	I_BYTE_LI
I_TOK_BR1	> 0	= 5	– (I_TOK_VLINE | I_TOK_VLINE_ | I_TOK_PIC | I_TOK_LI2)
I_TOK__BR1	= 0	= 5	– (I_TOK__VLINE | I_TOK__VLINE_ | I_TOK__PIC | I_TOK__LI2)
I_TOK_QUOTE	> 0	= 8	– (I_TOK_VLINE | I_TOK_VLINE_ | I_TOK_PIC)
I_TOK__QUOTE	= 0	= 8	– (I_TOK__VLINE | I_TOK__VLINE_ | I_TOK__PIC)
I_TOK_ELSE	> 0	> 0	– (I_TOK_VLINE | I_TOK_VLINE_ | I_TOK_VLINE1 | I_TOK_VLINE1_ | I_TOK_PIC | I_TOK_PAR | I_TOK_LI1 | I_TOK_BR | I_TOK_LI2 | I_TOK_BR1 | I_TOK_QUOTE)
I_TOK__ELSE	= 0	> 0	– (I_TOK__VLINE | I_TOK__VLINE_ | I_TOK__VLINE1 | I_TOK__VLINE1_ | I_TOK__PIC | I_TOK__LI1 | I_TOK__BR | I_TOK__LI2 | I_TOK__BR1 | I_TOK__QUOTE)

NBR = (* 1...2^30 *);

Table 4: Items

Item	Definitions–list according to *ISO–14977*
I_ITEM__LI1	(I_TOK__LI1 | I_TOK_LI1 | I_TOK__BR1 | I_TOK_BR1) {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM__LI2	(I_TOK__LI2 | I_TOK_LI2) {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM__PAR	(I_TOK__BR | I_TOK_BR | I_TOK__ELSE | I_TOK_ELSE | I_TOK_PAR) {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM__QUOTE	(I_TOK__QUOTE | I_TOK_QUOTE) {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM_H1	I_TOK_H1 {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM_H2	I_TOK_H2 {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM_H3	I_TOK_H3 {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM_H4	I_TOK_H4 {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM_H5	I_TOK_H5 {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM_HBIB	I_TOK_HBIB {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM_HEQU	I_TOK_HEQU {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM_HFIG	I_TOK_HFIG {I_TOK__ELSE | I_TOK_ELSE}
I_ITEM_HTAB	I_TOK_HTAB {I_TOK__ELSE | I_TOK_ELSE}

Item	Definitions–list according to *ISO–14977*
I_ITEM_LATEX	I_TOK_LATEX {I_TOK_LATEX | I_TOK__VLINE1 | I_TOK__VLINE1_ | I_TOK__VLINE | I_TOK__VLINE_ | I_TOK__PIC | I_TOK__BR | I_TOK__LI1 | I_TOK__BR1 | I_TOK__LI2 | I_TOK__QUOTE | I_TOK__ELSE}
I_ITEM_MAN	I_TOK_MAN {I_TOK_MAN | I_TOK__VLINE1 | I_TOK__VLINE1_ | I_TOK__VLINE | I_TOK__VLINE_ | I_TOK__PIC | I_TOK__BR | I_TOK__LI1 | I_TOK__BR1 | I_TOK__LI2 | I_TOK__QUOTE | I_TOK__ELSE}
I_ITEM_PIC	(I_TOK__PIC | I_TOK_PIC) {I_TOK_VLINE | I_TOK__VLINE1 | I_TOK__VLINE1_ | I_TOK__VLINE | I_TOK__VLINE_ | I_TOK__PIC | I_TOK__BR | I_TOK__LI1 | I_TOK__BR1 | I_TOK__LI2 | I_TOK__QUOTE | I_TOK__ELSE}
I_ITEM_PRE	(I_TOK__VLINE_ | I_TOK_VLINE_ | I_TOK__VLINE | I_TOK_VLINE) {I_TOK_VLINE | I_TOK__VLINE1 | I_TOK__VLINE1_ | I_TOK__VLINE | I_TOK__VLINE_ | I_TOK__PIC | I_TOK__BR | I_TOK__LI1 | I_TOK__BR1 | I_TOK__LI2 | I_TOK__QUOTE | I_TOK__ELSE}
I_ITEM_PRE1	(I_TOK__VLINE1_ | I_TOK_VLINE1_ | I_TOK__VLINE1 | I_TOK_VLINE1) {I_TOK_VLINE1 | I_TOK__VLINE1 | I_TOK__VLINE1_ | I_TOK__VLINE | I_TOK__VLINE_ | I_TOK__PIC | I_TOK__BR | I_TOK__LI1 | I_TOK__BR1 | I_TOK__LI2 | I_TOK__QUOTE | I_TOK__ELSE}
I_ITEM_INTR	I_TOK_INTR {I_TOK_INTR}

6 Bibliography

HTML HyperText Markup Language 4.0 Specification.
`http://www.w3.org/TR/1998/REC-html40-19980424`

ISO–8859–1 ISO/IEC 8859–1 Information technology. 8–bit single–byte coded graphic character sets. Part 1. Latin alphabet No. 1. 1998.

ISO–8859–5 ISO/IEC 8859–5 Information technology. 8–bit single–byte coded graphic character sets. Part 5. Latin/Cyrillic alphabet. 1999.

ISO–8859–7 ISO/IEC 8859–7 Information technology. 8–bit single–byte coded graphic character sets. Part 7. Latin/Greek alphabet. 2003.

ISO–14977 ISO/IEC 14977. Information technology. Syntactic metalanguage. Extended BNF. 1996.

LaTeX The document markup language and document preparation system for the TeX typesetting program. *LaTeX* was written by Leslie Lamport in the early 1980s. The current version is *LaTeX2e*.

UTF–8 RFC 3629. UTF–8, a transformation format of ISO 10646. 2003.

roff_man The groff/nroff/troff macros to support generation of man pages.