The domain text is used to factor out of PIPS prettyprinter all layout considerations. It does not depend on PIPS internal representation.

The idea is to have higher level structures to manipulate text source more efficiently. Since a text is mainly a list of sentences and sentences a list of words, movings sentences or concatenating sentence is more efficient than working directly on arrays of char, since we just work on the sentence list.

The final layout (with indentation...) can be delayed through the use of `unformatted` sentences.

Text = `sentences:sentence*`

A text is a list of sentences.

Sentence = `formatted:string + unformatted`

Each sentence can be either formatted and the corresponding string must be printed out exactly as it is, or unformatted and line breaks and indentation must be generated.

Unformatted = `label:string x number:int x extra_margin:int x words:string*`

Because of the Fortran origin, but it is also useful in C, a label can be defined as a string for each sentence. The number field is the statement number. The indentation is defined by field extra_margin, extra because of Fortran 77 fixed input fields. Finally a list of words define the content of the sentence. The linefeed can be inserted between two words and the continuation handled if necessary.

Since in C we also have some `#pragma` between the label and the instruction, there is a little semantics gap compared to Fortran here...

FI: I assume that the text library depends on the output language.

FI: Fabien Coelho has developped another interface to output source code.