Line Formatter Natural Language Specification

Consider the requirements specification given below for a simple line formatter and answer the following questions:

(a) How well does the specification do with respect to the following qualities: abstract, correct, unambiguous, complete, consistent and verifiable?

(b) For a requirement specification like that given, what are the advantages and disadvantages of maintaining both a formal specification and a natural language specification?

Meyer (1985), On Formalism in Specifications, provides one analysis of the shortcomings of the given natural language specification that you may find helpful in preparing your answer.

Line Formatter Specification

The program's input is a stream of characters whose end is signalled with a special end-of-text character, ET. There is exactly one ET character in each input stream. Characters are classified as:

- break characters - BL (blank) and NL (new line);
- non break characters - all others except ET;
- the end-of-text indicator - ET.

A *word* is a non-empty sequence of non break characters. A *break* is a sequence of one or more break characters. Thus, the input can be viewed as a sequence of words separated by breaks, with possible leading and trailing breaks, and ending with ET.

The program's output should be the same sequence of words as in the input, with the exception that an oversize word (i.e. a word containing more than MAXPOS characters, where MAXPOS is a positive integer) should cause an error exit from the program (i.e. a variable, Alarm, should have the value TRUE). Up to the point of an error, the program's output should have the following properties:

1. A new line should start only between words and at the beginning of the output text, if any.
2. A break in the input is reduced to a single break character in the output.
3. As many words as possible should be placed on each line (i.e. between successive NL characters).
4. No line may contain more than MAXPOS characters (words and BLs).