On the use of Ambiguity Measures in Requirements Analysis[*]

Mich Luisa

Department of Computer and Management Sciences
University of Trento
Via Inama 5, I-38100 Trento
mich@cs.unitn.it

Abstract: Software engineering involves numerous activities in which non-ambiguous artificial languages may be used. This is not always possible, however, and requirements analysis is by nature more closely connected to the use of natural language. An essential characteristic of natural language is ambiguity, understood as the possibility to interpret words or phrases in different ways. For documents used in software development, and particularly in the specification of requirements, the presence of diverse interpretations constitutes a serious problem. A first point to stress is that there are various types of ambiguity and that they can be detected and handled in different ways. In particular, a distinction can be drawn between the semantic ambiguities of words or phrases, and the syntactical ambiguities that arise from the various roles performed by words within a sentence and in connecting its parts together. To take into account the various levels of ambiguity, we have introduced a family of ambiguity measures. In this paper we present some preliminary findings of experiments regarding the applicability and effectiveness of these measures using the linguistic instruments employed in natural language processing.

Keywords: Quality of Requirements Documents, Ambiguity Measures, Natural Language Processing

1. Introduction

Natural language plays an important role in the elicitation and definition of requirements. There are several different reasons for this. First and foremost, natural language is essential for effective communication and cooperation among the people involved in a software development project (see, for example [Ma96]). Furthermore, the main objective of requirements analysis is to identify and understand potential problems before modeling the requirements [Da96], and a large part of the documentation available is usually in natural language [LK95]. With regard to this point we can also cite a market study which found that 79% of documents used for requirements analysis are couched in common natural language, 16% of them in structured natural language (e.g. templates, forms), and only 5% in formalised language.[1]

[*] This study was partly supported by the Italian M.U.R.S.T. national project SALADIN.

[1] This survey is part of research program of the Computer and Management Sciences Department of Trento University. The results are available at the site www.online.cs.unitn.it.

On the other hand, the flexibility and usability of natural language are the result of a process of optimization of diverse parameters which are essential for effective communication. In fact, good linguistic competence is based on the adjustment of language to pragmatic purposes [Gr75] in order to minimize the effort of communication. Thus, the "naturalness" of language is attained at the cost of a certain degree of ambiguity. For example, the use of scientific terms with unequivocal meanings may help reduce ambiguity, but it requires considerable cognitive effort not only by the speaker (or writers) but also by the listener (or reader).
Ambiguity often results in confusion, and the use of natural language in analysis requirements is no exception. For this reason, non-ambiguity is an attribute explicitly called for in models assessing the quality of requirements ([HE93][2], [Me85][3], [RR99])[4].
When considering ambiguity, a first point to stress is that there are various types of ambiguity [Le83], [Wa96]. The challenge is to identify them and bring them to the analyst's attention. This is the aim of our approach, which proposes the use of techniques developed in the realm of natural language processing (NLP) to begin to solve these two problems. In particular, to take into account the various levels of ambiguity, we have introduced a family of ambiguity measures [MG00]. The objective of this paper is to refine these measures and to present some general guidelines regarding linguistic instruments that can be useful in applying them.
There are already numerous research projects, proposals and ideas regarding the use of linguistic tools in requirements analysis, and their number continues to grow alongside progress in the area of NLP. They range from methods that facilitate the use of natural language in writing specifications (see, for example [RA98]), or "systematic reading techniques" for software inspection ([La00], [Sh00]), to projects for the development of conceptual models[5]. However, this paper focuses on the preliminary analysis of documents in order to detect ambiguities and signal them to the analyst or the user, regardless of the development method adopted. In this sense, the most similar approach is that described in [Wi97][6], which identified a number of categories that can be used to analyze specifications in natural language. Having a predefined checklist makes it easier to identify potential areas of ambiguity, but at the same time a predefined list could be limiting; this problem, however, can be overcome by using the appropriate linguistic tools. Another project with similar objectives puts forward an interface which draws on the functionality of a general purpose NLP system [MP95] to aid in developing specifications. In order to show the user the different interpretations of the phrases in question, the authors introduce "logicians' English". Given that an ideal approach should not require the user to learn new notations, we shall see how the NLP system LOLITA presents different parsing trees through the use of bracketing. At another level, it is

---

[2] The authors have identified four groups of defects in requirements documents, so that apart from Ambiguity there is Redundancy, Inconsistency and Terminological errors (ARIT model).
[3] Ambiguity is one of the "seven sins of the specifier" defined by Meyer, the others being noise, silence, overspecification, contradiction, forward reference, wishful thinking.
[4] The authors introduced a "Quality Gateway" to test requirements and ambiguity is among the 10 items of the model.
[5] A thorough review of projects dealing with the use of linguistic instruments in software engineering is beyond the scope of this study. For a critical introduction see [Ry92]. For a good introductory text on the topic see [Bu97]; see also the papers of the NLDB conferences. A bibliography of projects related to the definition and modelling of requirements is given at www.nl-oops.cs.unitn.it.
[6] Web site of the project: http://satc.gsfc.nasa.gov/support/index.html

worthwhile to note the quality framework set forth in [Fa98]. With respect to this framework, which includes four levels of assessment, our approach provides support for quality assessment activities related to syntax and semantics.
The paper is organized as follows: the second section describes the various types of ambiguity and introduces the ambiguity measures for the identified levels. The third section looks at the feasibility of calculating these indices using linguistic instruments of differing complexity, among these the knowledge base of the NLP system LOLITA [LG94], [Mo96] and the outputs produced during the text parsing phase. To this end the paper refers to some preliminary findings of experiments designed to identify terms and phrases open to several interpretations and thereby to improve the quality of the requirements analysis process. Finally, the conclusions set out some proposals for further research.

2. Ambiguity measures

2.1 Ambiguity types

Words and sentences in natural language may correspond to a vast number of meanings. Each word in a sentence may correspond to many different concepts, and there may be more than one set of grammatical dependencies among the words in different sub-phrases of the sentence. This paper assumes the following levels of ambiguity:[7]

– semantic ambiguity: concerning the meaning of a word or phrase;
– syntactic ambiguity: concerning the various roles performed by words in sentences and possible grammatical constructions.

As regards individual words, there are those that can represent different senses or which can be used as both a verb and a noun (part of speech ambiguity). For example, in English a 'bank' is a financial institution or the edge of a river, and there is also the verb 'to bank'). At the phrase or sentence[8] level there may be semantic ambiguity due to the presence of ambiguous words, and syntactic or structural ambiguity due to the possibility of connecting the components of the phrase in different ways ('I saw the man in the park' is ambiguous if we have to decide who was in the park). There is also a third level of ambiguity, pragmatic ambiguities, which are more difficult to detect and resolve because they concern relations more than content. This paper focuses on the first two levels of ambiguity.
Another aspect to consider, is the role played by the context, which may influence the understanding of a phrase positively or negatively[9]. For purposes of this study, we can assume that, regardless of the context, by reducing the ambiguity of words and phrases we can improve the quality of the requirements. On the basis of the classification set out in this section, we now introduce a family of measures of ambiguity. These measures are given a general definition without reference to a particular linguistic instrument or NLP system.

[7] For a more in-depth study of the definition and classification of ambiguity see [Wa96].
[8] In linguistics, a phrase is a complete sub-sentence unit.
[9] For example, the phrase "Chocolate, I think." is incomprehensible apart from the contextual question: "What flavour of ice-cream does she like?".

2.2 Lexical ambiguity

At the level of individual words we can talk of lexical ambiguity[10] As we have seen, a word may have different meanings or senses (semantic ambiguity) or it may comprise different syntactic units or part of speech (syntactic ambiguity).
Let us suppose that we have a finite set W of N words.[11] The set W_i represents the possible meanings of a word w_i. We thus have:

$w_i \in W$ con $i = 1,\ldots,N$

$W_i \equiv \{m^i_j \;\; j = 1,\ldots,n_i\}$ with $n_i \geq 1$

Corresponding to each meaning of the word w_i is a possible syntactic role (e.g. noun, adjective, verb) which we denote with r_i.The number of possible roles is determined by grammar, and therefore should already be known.
Finally, the meanings of a word are used with differing frequencies. These frequencies can be used to weigh its ambiguity. Hence, associated with each word may be set of terms thus constituted:

$w_i \equiv \{<m^i_j, r^i_j, v^i_j> \mid j = 1,\ldots, n_i\}$

We now introduce the following measures of lexical ambiguity:

$\alpha(w_i) = f(n_i)$	**Semantic ambiguity:** function of the number of possible meanings.
$\alpha^*(w_i) = f(n_i, v_i)$	**Weighted semantic ambiguity:** function of the number of possible meanings weighted according to their frequency.
$\beta(w_i) = f(n(r_i))$	**Syntactic ambiguity:** function of the number of possible syntactic roles.
$\beta^*(w_i) = f(r_i, v_i)$	**Weighted syntactic ambiguity:** function of the number of possible roles weighted according to their frequency.

The most intuitive way to determine the semantic ambiguity of a word is to consider the number of possible meanings; thus we have:

$\alpha(w_i) = n_i$

Using these definitions, in the case of a single meaning or a single role for a sentence unitary ambiguities are obtained, while nil values are obtained for unknown words[12].

2.3 Phrase and sentence ambiguity

The meaning of a phrase or of a sentence depends on the interpretation given to both the words comprised in it and the number of possible parsing trees (parsing forest).

[10] For a series of studies on lexical ambiguity see [Go89].
[11] The number of words varies over time with the introduction of new ones: $N = N(t)$.
[12] This case has to be handled carefully and used to signal the unknown words to the analyst. If it is wished to emphasise non-ambiguity with a nil value of the ambiguities, it is sufficient to redefine them by subtracting 1. In this case, the measures assume a negative value for words devoid of meaning

Let S_k be the set of the parsing trees for the sentence s_k, where S_k is empty if the sentence is not syntactically acceptable.

$$S_k \equiv \{t^k_l \quad l = 1,\dots n_k\}$$

It is possible to assign a penalty to each of these trees so that those obtained by parsing the sentence can be ordered taking into account the choices made in their construction. Hence, associated with each sentence is a set of triplets thus constituted:

$$s_k \equiv \{<t^k_l, p^k_l, g(\alpha^k_l)> \quad l = 1,\dots n_k\}$$

Where $g(\alpha^k_l)$ takes into account the semantic ambiguity of the words present in a phrase. Given that the effect of parsing is to establish the syntactic role of words, such ambiguity can be inferior to that of the word in isolation. Thus for the m_k words in a phrase:

$$\alpha^k_l(w_i) \leq \alpha(w_i) \qquad \text{with } i = 1, \dots, m_k$$

In this sense, it is possible to affirm that $\alpha(w_i)$ constitutes the highest limit of ambiguity for the word which is in isolation and for which there is no contextual information.
We can thus say that for the ambiguity of a phrase:

$\gamma(s_k) = f(g(\alpha^k_l))$	**Semantic ambiguity:** takes into account the ambiguity of the words in the phrase.
$\gamma^*(s_k) = f(g(\alpha^{*k}_l))$	**Weighted semantic ambiguity:** this takes into account the weighted semantic ambiguity of the words.
$\delta(s_k) = f(n_k)$	**Syntactic ambiguity:** function of the number of possible parsing trees.
$\delta^*(s_k) = f(n_k, p_k)$	**Weighted syntactic ambiguity:** function of the number of possible parsing trees and the penalty associated with them.

For the syntactic ambiguity of a phrase we can assume:

$$\delta(s_k) = n_k$$

where considerations are the same as those made for the semantic ambiguity of a word in phrases having only one parsing tree.
Furthermore, bearing in mind that ambiguity is combinatorial - that is to say, to obtain the ambiguity of a sentence, the numbers of alternatives for each locus of ambiguity must be multiplied rather than added, we used the following formula for the semantic ambiguity of a phrase:

$$\gamma(s_k) = 1/n_k \cdot \Sigma_{l=1,\dots n_k}(\Pi_{i=1,\dots m_k} \alpha^k(w_i))$$

and similarly, for the weighted semantic ambiguity:

$$\gamma^*(s_k) = 1/n_k \cdot \Sigma_{l=1,\dots n_k}(\Pi_{i=1,\dots m_k} \alpha^{*k}(w_i))$$

In these formulas it is necessary to avoid that a word have an ambiguity value of 0. Using the number of meanings as a measure of semantic ambiguity guarantees that this will not happen.
While for weighted syntactic ambiguity:

$$\delta^*(s_k) = \varepsilon \cdot (1/n_k \cdot \Sigma_{l=1,\dots n_k} p_l)$$

where ε is a constant used as a scale that takes into account the penalty values of the NLP system being used. In general, whether for a phrase or of a word, the choice of functions can be guided by certain desirable characteristics that ambiguity measures should have, and these are:

- intuitive: because they aid in comprehension and applicability;
- general purpose and domain independent: because they can be applied for different purposes and in different domains;
- usable: because they can be obtained using diverse linguistic tools.

On this theme, a study is underway which is looking at the possibility of using different functions; for example introducing for α a logarithmic function, which is not applied for words having only one meaning and is indefinite for words without meaning. From a theoretical perspective this study points the way to further research into the links between ambiguity and the information contained in a word [Gu97]. However, this function is less intuitive both in identifying ambiguous terms and in calculating the semantic ambiguity of a phrase. Therefore, we have chosen to adopt the simplest method for our research and to focus on the application of measures introduced earlier in this paper.

3. Application of ambiguity measures

In this section, we summarize some of the preliminary results and problems encountered during early experiments with ambiguity measures in requirements analysis, and we make suggestions regarding issues for further research. Current experiments deal principally with the choice of the linguistic instruments that can be used to identify the different types of ambiguity and to point out the ambiguity to the analyst. To illustrate this point we refer here to two different application, one for lexical ambiguity measures (menu commands of Netscape[13]) and one for sentence ambiguity measures (ABC Video). As regards the measures for lexical ambiguity (defined for single terms), it is not frequently necessary to evaluate ambiguity of isolated words. A practical application of the measures of lexical ambiguity was done to assess the menu commands of Netscape. To evaluate the semantic (α) and syntactic ambiguity (β) of the names of the commands we used two different systems, Wordnet[14] and LOLITA[15].

The analysis pointed out that the commands have very different levels of ambiguity. Using Wordnet, α assumes values in a range from 1 to 28 (for the *back* command), where β goes from 1 to 4. These results are comparable with those obtained using LOLITA, where for α we obtained lower average values. On the basis of this experiment, the following points can be underlined:

– Command names with the highest values of semantic ambiguity should be further analyzed, possible substituting the names with others less ambiguous names.

[13] Copyright © Netscape Communications Corporation

[14] WordNet - a Lexical Database for English, Cognitive Science Laboratory, Princeton University: www.cogsci.princeton.edu/~wn/w3wn.html

[15] A description of LOLITA and the use of commands to assess ambiguity measures, with examples of the use of annotation is found in [MG00].

However, these values can seem surprising and should be used with caution[16].
– The presence of command names with high value of syntactic ambiguity suggests that the menu commands could also be analyzed to standardize their syntactic role (e.g., to have all nouns or all verbs).
– The lexical ambiguity measures can be useful when setting up or re-designing an interface or help menu, not to mention the eventual benefits when training new end-users.
– On a theoretical level, the difference in the values obtained with Wordnet and LOLITA reflects the need to take into account the size of the dictionary or of the knowledge base used (the larger the dictionary, the higher the average number of meanings included).
The assessment of sentence ambiguity requires the use of parsing instruments that are able to produce parsing trees corresponding to the different interpretations and are able to determine the semantic ambiguity of terms within the context of the sentence. With this in mind, we used the NLP system LOLITA for our experiments, as it produces a list of penalty-ordered trees, in which also the ambiguity of terms is given. To obtain this information, the LOLITA commands *pasbr* and *tp* can be used. The effect of the first command on an input sentence is to produce all the syntactic trees with two representations, the first of which also shows the semantic ambiguity of the terms of the sentence, while the second – based on the use of bracketing - enables understanding of the interpretation associated with the tree (see figure 1). The command *tp* produces the penalty values associated with the trees, in addition to information regarding the context. Penalties can be interpreted as measures of the effort made by the NLP system, and therefore of the effort required to interpret sentences. In LOLITA, penalties are classified into four groups of increasing seriousness:
– Usage penalties, to indicate less common constructs[17] ($p<30$).
– Minor feature clashes, e.g. wrong concordance ($30<p<100$).
– Major feature clashes[18], ($100<p<1000$).
– Structural problems, e.g. missing or repeated parts of speech ($p>1000$)[19].
A possible rule for using these values is as follows: if the parsing of a sentence produces only trees with penalties higher than 1000, the analyst should adjust the requirements text. For penalties of less than 1000, if trees with equal penalties are obtained for a sentence, these can be considered *essential* ambiguities which require further information in order to resolve the ambiguity. Due to limited space we can mention here only some of the results for the ABC Video case (the text reference is in the Appendix)[20]. In this case we used the output of the *tp* and *pasbr* commands of LOLITA to evaluate the semantic and syntactic ambiguity for sentences (γ and δ, respectively). We obtained a single syntactic tree (γ) only for four sentences (4, 7, 10, 12) with penalties in all four groups mentioned above. The sentences having a higher number of

[16] In fact, they point out the role of the graphic symbols within the context comprising all applications with which agreements of usage and meaning have been established.
[17] E.g., if a word has both a noun and an adjective form, and is used in apposition to another noun, the adjective form is usually preferred, as in 'human behaviour'.
[18] E.g. (apparent) dative or infinitive use of inappropriate verbs.
[19] If the system is not able to produce a parsing tree, there are serious problems with the phrase.
[20] The version used for the experiment is part of the Experimental Material of the ESEG research group: www.cs.umd.edu/projects/SoftEng/ESEG/

trees are number 1 (10 trees), number 15 (12 trees), and number 17 (18 trees). All of these sentences (along with numbers 2 and 14) fall into the fourth penalty group. In addition, there is high variability even for semantic ambiguity (γ), which ranges from a few units (sentence 10) up to a million (sentence 17). For example, for the first sentence the different interpretations derive from the fact that "at least" can refer to either the action of choosing or to the number; similarly, "for rental" can correspond to the video or to the choice. In fact, the high penalty level assigned to the parsing trees here (>1000) is due to "for rental", and decreases to less than 30 when "to rent" is substituted.

```
pasbr

information:

Customers select at least 1 video for rental.
(…)

sen
  missing_det
    comnoun CUSTOMER [Plur,Sexed,Per3]
  transvp
    verb SELECT [Pres,NoPer3S]
    missing_det
      anoun01
        intense_adject
          adverb AT LEAST
          adj_quantity 1
        relprep01
          comnoun VIDEO [Sing,Neutral,Per3] * 2
          prepp
            prepNormRel FOR
            missing_det
              comnoun RENTAL [Sing,Neutral,Per3] * 2
(Customers (select ((at least 1) (video (for rental)))))
```

Figure 1 Extract of the output from the *pasbr* command

Sentence number 15 also represents a case of essential ambiguity because the calculation of "past-due" can be attributed to either the "option" or the "employee".
A general consideration, yet an important one, is that there is not a linear relationship between the length of a sentence and its ambiguity. Nonetheless, especially where there is a large number of parsing trees, there is the need to determine the reliability of the measures produced by the system in use. Using LOLITA, which has a hierarchically organised knowledge base, it is possible to have suggestions regarding the appropriateness of terms and regarding the preparation of a glossary for the application. In practice, for terms with a high level of semantic ambiguity, it is possible to determine if there is a more precise meaning. For example, in the case study of ABC Video System, 11 terms have a semantic ambiguity above 6 (in the parsing trees); 4 of these terms are verbs (mark, read, return, take); 3 are present in the case glossary prepared by the author (account, form, card) corresponding to 2/3 of the terms used in the text. Another highly ambiguous term is "order", which in this case can be considered a synonym of "rental" [Co94]. For some terms with an ambiguity value between 3 and 5, this apparent lack of clarity can be corrected taking into consideration that the terms refer to the computer domain (enter, print, store, create, calculate, compute). Care should be taken, nonetheless, because "return" in this case means "give back" and does not refer to a button on the keyboard.

4. Conclusions

This paper has described some ambiguity measures and has looked at some preliminary findings of experiments and studies regarding the applicability and effectiveness of these measures using linguistic instruments. On a theoretical level, several areas requiring further research have emerged. The most important of these deal with the choice of functions in the formulas introduced for the diverse types of ambiguity and the study of their properties. Furthermore, experiments done thus far have suggested that it could be useful to introduce another type of ambiguity measure, one that takes into account the "points" which serve to differentiate the syntactic trees ("structural" ambiguity). The applicability and efficacy of the ambiguity measures described here was investigated with instruments of increasing complexity, arriving at the use of a system that enables a deep analysis of texts and that are able to offer sophisticated support regarding the adjustments made to requirements texts and the creation of a glossary for the application. From a practical standpoint, it is necessary to look into the possibility of integrating several linguistic instruments within one environment so as to give the analyst a choice of instruments during each work session. This would lead to the eventual integration with early life tools for the conceptual modelling, as in the prototype NL-OOPS set forth in previous research [Mi96], [MG99].

References

[Bu94] Burg JFM. Linguistic Instrument in Requirements Engineering, IOS, Amsterdam, 1997.

[Co94] Conger SA. The new software engineering, 1994.

[Da98] Davis AM. The Harmony in Rechoirments. IEEE Software, March/April 1998; 6-8.

[Fa98] Fabbrini F, Fusani M, Gervasi V, Gnesi S, Ruggieri S. Achieving Quality in Natural Language Requirements. In: Proc Int SW Quality Week, S.Francisco CA, May 1998.

[Go89] Gorfein S. (ed) Resolving Semantic Ambiguity. Springer-Verlag, 1989.

[Gr75] Grice HP. Cooperative Principle. 1975. In: Cole P. and Morgan JL (eds) Syntax and Semantics; Vol. 3: Speech Acts; San Diego CA, Academic Press, pp. 41-58.

[Gu97] Guiasu S. Information theory with applications. Mc-Graw-Hill, 1997.

[IE93] IEEE Std 830-1993. Recommended Practice for SW Requirements Specifications. Dec 2, 1993.

[La00] Laitenberg O, Atkinson C, Schlich M, El Emam K. An experimental comparison of reading techniques for defect detection in UML design documents. J of S&SW, North Holland-Elsevier 2000; 53: 183-204.

[Le83] Levinson. Pragmatics. Cambridge University Press, 1983.

[LG94] Long D, Garigliano R. Reasoning by Analogy and Causality: Model and Applications. Ellis Horwood, Chichester, UK 1994.

[LK95] Loucopoulos P, Karakostas V. System Requirements Engineering. McGraw-Hill, London, 1995.

[Ma96] Macaulay LA. Requirements Engineering. Springer, 1996.

[Ma95] Macias B, Pulman SG. A method for controlling the production of specifications in natural language. The Computer J 1995; (38) 4: 310-318.

[Me85] Meyer B. On formalism in specification. IEEE Software 2(1):6-26, January 1985.

[MG99] Mich L, Garigliano R. Ambiguity measures in Requirements Engineering. In: Proc. Int Conf on SW-Theory&Practice-ICS2000, 16th IFIP WCC, Beijing, China, 21-25 Aug, Feng Y, Notkin D, Gaudel M. (eds), House of Electronics Industry,2000:39-48.

[MG99] Mich L, Garigliano R. The NL-OOPS project: Object-Oriented Modelling using the Natural Language Processing System LOLITA. In: Flied G, Mayr HC (eds). Proc 4th Int Conf on Applic of NL to IS, NLDB'99, Schriftenreihe OCG 129, 1999, 215-218.

[Mi96] Mich L. NL-OOPS: From Natural Language to OO Requirements using the Natural Language Processing System LOLITA, J of NLE, Cambridge, 2 (2): 161-187, 1996.

[Mo96] Morgan R, Garigliano R, Callaghan P, Poria S, Smith M, Urbanowicz A, Collingham R, Costantino M, Cooper C & the LOLITA Group. Description of the LOLITA System as used in MUC-6. Proc 6th ARPA-MUC, Morgan Kaufmann, 1996.

[RR99] Robertson S, Robertson J. Mastering the Requirements Process. Addison Wesley, 1999.

[RA98] Rolland C, Achour C. Ben Guiding the construction of textual use case specifications. Data and Knowledge Engineering 1998; 25: 125-160.

[Ry99] Ryan K. The Role of Natural Language in Requirements Engineering. IEEE 1992, 240-243.

[Sh00] Shull F, Lanubile F, Basili VR. Investigating Reading Techniques for Object-Oriented Framework Learning, IEEE Transactions on SWE, 2000; (26) 11: 1101.

[Wa96] Walton D. Fallacies Arising from Ambiguity. Kluwer, 1996.

[Wi97] Wilson MW, Rosenberg LH, Hyatt LE. Automated Analysis of Requirements Specifications. In Proc. Int Conf.on SW Engineering, May 1997.

Appendice – General description of the ABC Video System

1. Customers select at least one video for rental.
2. The maximal number of tapes that a customer can have outstanding on rental is 20.
3. The customer's account number is entered to retrieve customer data and create an order.
4. Each customer gets an id card from ABC for identification purposes.
5. This id card has a bar code that can be read with the bar code reader.
6. Bar code Ids for each tape are entered and video information from inventory is displayed.
7. The video inventory file is updated.
8. When all tape Ids are entered, the system computes the total bill.
9. Money is collected and the amount is entered into the system.
10. Change is computed and displayed.
11. The rental transaction is created, printed and stored.
12. The customer signs the rental form, takes the tape(s) and leaves.
13. To return a tape, the video bar code ID is entered into the system.
14. The rental transaction is displayed and the tape is marked with the date of return.
15. If past-due amounts are owed they can be paid at this time; or the clerk can select an option which updates the rental with the return date and calculates past-due fees.
16. Any outstanding video rentals are displayed with the amount due on each tape and the total amount due.
17. Any past-due amount must be paid before new tapes can be rented.