Avoiding Ambiguities in Arabic Software User Requirements

Hanan Elazhary
Computers and Systems Department, Electronics Research Institute, Cairo, Egypt
hananelazhary@eri.sci.eg, hananelazhary@hotmail.com

Abstract

Errors in requirements elicitation cause much more expensive errors in later phases of software engineering. A major source of errors is the ambiguity of the natural languages initially used to write the user requirements. Most of the research studies in the literature have focused on ambiguity detection and correction after the user requirements are written. A better approach would be avoiding the introduction of ambiguities while the user requirements are being written to reduce time and cost. Unfortunately, this problem has been overlooked in the literature except for few research studies concerned with the English language. This paper is concerned with the Arabic language due to its complex structure and because the ambiguity of Arabic user requirements is a serious problem hindering the production of quality software in the middle east. The paper studies main sources of ambiguities in the Arabic language in addition to best practices in writing Arabic user requirements to avoid the introduction of ambiguities as suggested by interviewed Arabic software engineers or as suggested in the literature for English user requirements. Accordingly, the paper suggests a set of best practices suitable for Arabic user requirements based on both practical experience and theoretical background. Examples are provided for justification. Human subject testing and empirical evaluation have shown the effectiveness of the proposed practices.

Keywords: Ambiguity, Arabic, Requirements Elicitation, Requirements Engineering

1. Introduction

Requirements engineering is concerned with the elicitation and specification of the requirements of a given software system. Initially, user requirements describing high-level goals of a software system are elicited from the stakeholders in their native natural language [1]. This is especially true when developing software systems that process inputs expressed in the native language of the stakeholders [2-4]. Imprecision in the user requirements causes errors in later stages of software engineering [5]. Such errors are much more expensive to correct than the imprecision itself. Imprecision in the user requirements may be due to the inaccuracy, inconsistency and/or incompleteness of the requirements; but mainly due to the inherent ambiguity of the natural languages used to write these requirements. An ambiguous sentence is a sentence with unclear or indefinite meaning or a sentence with more than one meaning. Some ambiguities can be easily detected such as the case when a sentence has two possible meanings. But, many other types cannot be even realized such as the case when each reader interprets only one meaning for a given sentence, but the interpretation differs from one reader to the other.

Most of the research studies in the literature have focused on ambiguity detection and correction after the user requirements are written. A better approach would be avoiding the introduction of ambiguities while the user requirements are being written to reduce time and cost. Unfortunately, few research studies in the literature attempted to tackle this problem and, to the best of our knowledge, they all considered user requirements written in the English language [6-8]. Translating user requirements written in other natural languages to English to benefit from existing research studies would introduce more
ambiguities [9]. This fact has been emphasized by a research study [10] that has shown using a controlled experiment that using a native language for describing a software system has a positive effect on the correctness of the developed use case diagrams and on facilitating the task of assessing stakeholder needs to better understand the required system functions. This implies the need to consider user requirements written in natural languages other than English. In other words, techniques for avoiding ambiguities should be developed independently for different natural languages.

Arabic is the official native language of hundreds of millions of people in the Middle East and North Africa extending from the Arabian Gulf in the East to the Atlantic Ocean in the West. Thus, it is worth developing techniques for avoiding ambiguities in user requirements written in Arabic for the sake of such a large population. Another motivation behind this work is that practical experience has shown that the ambiguity of user requirements written in Arabic is a serious problem hindering the production of quality software in the Middle East. Additionally, Arabic is a challenging language due to its complex linguistic structure.

Towards this goal, we studied the relatively few best practices in writing English user requirements for avoiding the introduction of ambiguities as suggested by practitioners in the literature. Specifically, we considered best practices that are suitable for or that can be tailored to the Arabic language. Some of these practices were not explicitly suggested, but we induced them from research studies that attempt to detect ambiguities after the user requirements are written [11]. We also studied the main sources of ambiguities in the Arabic language that would affect user requirements. Additionally, we interviewed ten Arabic software engineers for suggested practices according to their experience. Accordingly, the paper presents a set of best practices that would help avoid the introduction of ambiguities while user requirements are being written in Arabic. In other words, the suggested practices are based on practical experience and on theoretical background. Examples are provided to justify the effectiveness of the suggested practices. This is further emphasized via human subject testing and empirical evaluation.

The paper is organized as follows: Section 2 presents related research in the literature and the suggested best practices in writing English user requirements provided that these practices are suitable for or can be tailored to the Arabic language. Section 3 discusses the main sources of ambiguities in the Arabic language that would affect user requirements with examples. Section 4 presents a set of suggested best practices in writing user requirements in Arabic to avoid the introduction of ambiguities with justifying examples. To be easily understood by non-Arabic speakers, the meaning of each Arabic word and/or each Arabic example sentence is provided in English. It is also worth noting that Arabic sentences are written and read from the right to the left. Section 5 discusses the results of human subject testing to further emphasize the effectiveness of the proposed practices. Finally, Section 6 provides the conclusions of the paper and directions for future research.

2. Related Work

This section presents related research in the literature and the suggested best practices in writing English user requirements to avoid the introduction of ambiguities provided that these practices are suitable for or can be tailored to the Arabic language. Unfortunately, few research studies in the literature attempted to tackle this problem. For example, Jain et al. [6] suggested the following set of best practices:

- Write complete sentences rather than bulleted buzz phrases
- Use entities and actions consistently and define them in a glossary
- Avoid using phrases with subjective meaning that can lead to ambiguity such as "easy to use"
• Use a specific syntax for each kind of requirements

Elazhary [7] suggested a set of rules to restrict the style of the written sentences such as using the apostrophe only to indicate possession, but also suggested the following set of best practices:
  • Write short active-voice declarative sentences
  • Do not use pronouns, possessive pronouns, or possessive adjectives
  • Each item is given a code in a glossary

Wiegers [8] suggested an even larger set of best practices that he induced from his relatively long experience in the field of requirements engineering:
  • Use proper grammar, accurate spelling, and well-constructed sentences
  • Use the word "shall" to identify a functional requirement, or pick an alternative word and use it consistently
  • Use annotations to indicate priority or different parts of a given sentence (stimulus, response, etc)
  • Use active-voice sentences
  • When writing requirements hierarchically, the parent requirement is written in the form of a header
  • Nested "or", "and", and "not" clauses are represented using a decision table or decision tree
  • Requirements involving compound operators are split into multiple requirements
  • Avoid negations
  • Define terms in a glossary
  • Use pronouns and abbreviations with caution
  • Avoid A/B constructs such as input/output
  • Avoid adverbs with several interpretations

Some research studies in the literature attempted to enforce such best practices and prevent or reduce the introduction of ambiguities through the use of templates or boilerplates. Unfortunately, most of these research studies considered specific domains such as scenarios [12], ontologies [13] and embedded systems [14]. RAT [6] is a relatively general template-based system, but considers only specific types of requirements. A similar system has been developed for the Arabic language to facilitate translation of the user requirements between Arabic and English [15]. Unfortunately, these systems are applicable to relatively simple sentences of specific constructs and so have not been widely adopted.

Some other researchers attempted to detect ambiguities after user requirements were written. Best practices in writing user requirements were thus implicit in their papers. For example, Kiyavitskaya et al [11] implicitly suggested the following set of best practices in writing user requirements in English:
  • When using a plural subject, you have to clarify whether you refer to each instance of the subject or to the plural subject as a whole
  • Avoid passive-voice sentences
  • Avoid using synonyms to refer to a single item
Yang et al. [16] developed a tool for analyzing anaphoric\(^1\) ambiguity in natural language requirements resulting from the difference in the interpretation of pronouns implicitly suggesting to try to avoid their usage. Wang [17] developed a tool for detecting overloaded ambiguity (when a word is used to refer to more than one item) and synonymous ambiguity (when two words are used to refer to a single item).

Semi-automated tools for ambiguity resolution have also been proposed in the literature. For example, Elazhary proposed two semi-automated interactive systems for ambiguity resolution, one for the English language [7] and the other for the Arabic language [18]. Gill et al. [19] also developed a semi-automated tool for ambiguity resolution in open source software requirements. Unfortunately, these systems are suitable for relatively simple sentences and require a lot of effort to inspect even a single sentence.

3. Main Sources of Ambiguities in the Arabic Language

Arabic is a challenging language because of its complex linguistic structure [20]. In this section, we discuss main sources of ambiguities in the Arabic language that would affect Arabic software user requirements.

3.1. The Diglossia Phenomenon

Diglossia has been defined by Ferguson [21] as a relatively stable language situation in which there exist a set of primary language dialects in addition to an often more grammatically complex superposed variety. This variety is used in a large and respected amount of written literature and is generally learnt by educated people to be used formally in writing and speaking, but never used for ordinary conversation. In other words, Diglossia [22] refers to the use of varieties of a single language in different situations within the same speech community\(^2\). Unlike English for example, Arabic can serve as a very good example of diglossia [23] since it is generally characterized by three diglossic situations:

- **Classical Arabic**, the language spoken by Arabic people more than fourteen centuries ago, is used in daily prayers.

- **Modern Standard Arabic (MSA)**, a variety of Arabic with continuous borrowings and innovations to meet the changing needs of Arabic speakers, is used by educated people in formal situations such as in the media, in speeches and when writing books and formal documents.

- **Regional dialects** are spoken by people in informal situations with family and friends and vary considerably from region to region; there are at least as many Arabic dialects as there are members of the Arab league [24]. Dialects vary even within a single country. In Egypt, for example, the used Arabic dialects vary considerably in different regions such as Cairo, Alexandria, Sinai, and the southern governorates.

In other words, these three varieties are used by a typical Arabic person on a daily basis. Each of the three varieties includes different vocabulary and has a different grammar. The vocabulary and grammar of classical Arabic and MSA are well established in dictionaries and books [25], but this is not true for regional dialects. Besides, a dialect used in a given region is not always easily understood by Arabic speakers in a different region. This presents a bigger challenge to non-Arabic speakers since they are taught MSA, but expected to communicate using local dialects [24].

---

\(^1\) The term anaphora refers to the use of an expression that needs another expression in context to be interpreted

\(^2\) The term speech community refers to a group of people sharing common expectations regarding the use of a given language.
3.2. The Absence of Sentence Boundaries

In English and Romance languages such as French, Spanish and Italian, a paragraph is typically divided into sentences and each sentence starts with a capital letter and ends with a full stop. Thus, boundaries of sentences can be easily identified. Unfortunately, this is not true in case of the Arabic language. It is common in Arabic to write an entire paragraph without any full stops except perhaps at the end. This is a serious source of ambiguities since two sentences can easily overlap. Example E1 shows an Arabic sentence that means "The system includes a sub-system that can add input numbers and includes a user-friendly graphical user interface".

In this sentence, it is not clear whether the system or the sub-system includes a user-friendly graphical user interface. In English, for example, this would be written as follows: "The system includes a sub-system that can add input numbers. The sub-system includes a user-friendly graphical user interface".

Re-writing the Arabic sentence in example E1 in a similar manner as shown in example E2 is not generally considered a convenient writing style.

3.3. The Existence of Translated and Transliterated Named Entities

MSA includes many translated and transliterated named entities, with non-standard spelling [26]. For example a named entity such as the city of Boston could be spelled in many different ways when written in Arabic: "بوسطن", "بوسطن", "بوسطن" or "بوستن". This can be a source of ambiguity if such a name is not realized by the reader or not used consistently. This is also true about translated technical words. For example the word "computer" is translated to "كمبيوتر", "حاسوب", "حاسب", "حسب" or "حاسب".

3.4. The Ambiguity of Named Entities

In English and other Romance languages, capitalization and punctuation play an important role in the task of realizing named entities [24]. Unfortunately, this is not true about the Arabic language and MSA, where there is no capitalization and writers do not stick to strict punctuation rules.

Thus, a named entity such as "علي" in the sentence in example E3, that means "The apple fell from Ali", can be interpreted as the name "Ali" or "a high place".

3.5. The Presence of Homographs and Absence of Diacritics

Arabic is characterized by the abundance of homographs, which are words with similar spelling, but different meanings and possibly different sounds. For example, the word "صبر" can mean "patience", "was patient", or "cactus plant". Maamouri and Bies [27] showed 21 different interpretations of the Arabic word "ثمن". It was estimated that the average number of ambiguities for a token in most languages is 2.3, while in MSA it can be as large as 19.2 [24]. The reason for this is that MSA has no short vowels. Diacritics, which are marks above or below the letters, can make up for the absence of short vowels. Unfortunately, MSA is typically written without diacritics.

---

1 The term transliterate refers to pronouncing letters and words using characters of another alphabet
2 A token is a string of one or more characters that is significant as a group.
3.6. The Relatively Free-Word Order

Unlike English, the Arabic language has a relatively free-word order [20]. For example, there are generally four different forms of a given declarative sentence such as the sentence "The first sub-system includes the second sub-system":

- subject-verb-object as shown in Example E4, which means "The first sub-system includes the second sub-system"

:E4

- object-verb-subject as shown in Example E5, which means "The second sub-system includes the first sub-system"; this is grammatically correct in Arabic.

:E5

- verb-subject-object as shown in Example E6, which means "Includes the first sub-system the second sub-system"; this is grammatically correct in Arabic.

:E6

- verb-object-subject as shown in Example E7, which means "Includes the second sub-system the first sub-system"; this is grammatically correct in Arabic.

:E7

The two forms object-verb-subject and verb-object-subject are confusing and can be very ambiguous. Besides, a verb-subject-object sentence can be easily interpreted as a verb-object-subject sentence and vice versa causing a serious ambiguity problem. For example, in examples E6 and E7, it is not clear whether the first sub-system is included in the second sub-system or vice versa.

Passive-voice sentences also have two different forms, which are both confusing since it is not clear who or what is doing the action:

- verb-object as shown in Example E8, which means "Is included the second sub-system"; this is grammatically correct in Arabic.

:E8

- object-verb as shown in Example E9, which means "The second sub-system is included".

:E9

In these two sentences, it is clear that the second sub-system is included, but it is not clear what includes it or what it is included in.

3.7. The Pro-Drop Property of the Arabic Language

Arabic is a pro-drop language; it allows subject pronouns to drop provided that the subject is recoverable [24]. In example E10, which means "Includes the second sub-system", it is not clear what includes the second sub-system. This can be interpreted in different ways according to the context and is grammatically correct in Arabic. Besides it can be interpreted as a passive-voice sentence as that in example E8.

:E10

It is clear that this pro-drop property of the Arabic language can be a serious source of ambiguities since such a sentence can have several different interpretations.

3.8. Transitive and Intransitive Verbs
In Arabic many verbs can be both transitive and intransitive. A transitive verb needs an object to complete the meaning of the corresponding sentence, while an intransitive verb does not. Combined with the passive voice and the pro-drop property, this can be a serious source of ambiguities [20].

The sentence in example E11 can be interpreted as "The second sub-system resumes" (verb-subject), "The second sub-system is resumed" (passive-voice verb-object) or "Resumes the second sub-system" (verb-object with a dropped subject pronoun).

4. Suggested Best Practices in Writing Arabic User Requirements

In this section, we present a set of suggested best practices when writing Arabic user requirements in an attempt to reduce possible sources of ambiguities. Some of these practices have been suggested in the literature for English user requirements and are applied to or tailored for the Arabic language and few are suggested by Arabic software engineers. We suggested some other practices for coping with main sources of ambiguities in the Arabic language. Examples provided in Section 3 are referenced and additional examples are given to justify the effectiveness of the application of these practices. The practices are listed and categorized in Table 1.

4.1. Practice P1: Stick to Modern Standard Arabic

Practice P1 suggests writing user requirements using MSA and avoiding the use of any dialects. This is to reduce the introduction of ambiguities due to the following reasons:

- MSA has well established vocabulary and grammar in dictionaries and books.
- Unlike classical Arabic, MSA is understandable by educated Arabic speakers no matter what their dialects are. In other words, MSA is a common language among educated Arabic speakers.
- MSA is typically the variety taught to (and thus understandable by) non-Arabic speakers.
- Allowing the use of dialects would introduce serious ambiguities as discussed in Section 3.1.

In other words, user requirements written using MSA are understandable by educated Arabic speakers with different dialects and by non-Arabic speakers who learnt Arabic as a second language.

4.2. Practice P2: Use English Technical Words within Arabic Sentences

Practice P2 suggests using a mix of Arabic and English. In other words, we should write all the requirements in Arabic except the technical words, that are written in English or at least accompanied by English translation. This practice, thus, helps avoid the ambiguities introduced by the use of translated words as discussed in Section 3.3.

4.3. Practice P3: Write Short Sentences as Much as Possible

It is sometimes advised to write sentences that are not longer than about 25 words [7] or to write each requirement as a single sentence [6], but such advices are not always valid or effective [8]. For example, shortening a sentence may introduce ambiguities due to omitting words needed to clarify the intended meaning. On the other hand, writing a requirement as a single sentence may require elongating the sentence leading to the
introduction of ambiguities. Thus, the third practice P3 merely suggests writing short sentences as much as possible.

4.4. Practice P4: Write Complete Sentences with Correct Grammar and Spelling

Practice P4 suggests writing complete sentences with correct grammar and spelling. In other words, the user requirements should not be written as bulleted phrases for example [6]. In Arabic, errors in grammar are worse than in English due to the relatively free form of the Arabic sentences as discussed in Section 3.6. Incorrect grammar can change the form of the sentence to another form with correct grammar, but different meaning (than the intended). Similarly, incorrect spelling can introduce ambiguities especially if a word is replaced by another word with similar sound, correct spelling, but different meaning. For example the meaning of the sentence "the user can access his account" is totally changed if we write "the user can assess his account". The spelling mistake in this later sentence can be easily overlooked since, though the word "assess" replaced the word "access" and changed the meaning, the word "assess" has a similar sound and correct spelling.

4.5. Practices P5: Write Only Active-Voice Sentences

Practice P5 suggests writing only active-voice sentences as suggested in the literature to clearly specify the actor/agent and the action [7, 8, 11].

4.6. Practices P6: Write Sentences of the Form Subject-Verb-Object

Practice P6 suggests writing sentences of the form subject-verb-object. It helps avoid ambiguities introduced not only by passive-voice sentences, but also by other forms of active-voice sentences in the Arabic language (verb-subject-object, object-verb-subject and verb-object-subject) as discussed in Section 3.6. It also helps avoid ambiguities introduced due to the pro-drop property of the Arabic language (allowing the drop of subject pronouns) as discussed in Section 3.7. and the problem of transitive and intransitive verbs discussed in Section 3.8.

4.7. Practice P7: Avoid Negative (or Inverse) Requirements as Much as Possible

Practice P7 suggests avoiding negative (or inverse) requirements as much as possible. This is because negative or inverse requirements can be a serious source of ambiguities [8] and are typically confusing. For example the sentence shown in example E12, which means "The user shall not access an account other than his" is better rewritten in a clearer form as shown in example E13, which means "The user shall access only his account".

4.8. Practice P8: Write Sentences in a Consistent Fashion

Practice P8 suggests writing sentences in a consistent fashion. Some researchers suggested associating specific words with different types of requirements. For example, Wiegears [8] suggested associating the word "shall" with functional requirements or choosing another word and using it consistently. On the other hand, Jain et al. [6] suggested associating a unique identifier with each requirement type. But, this later suggestion would be overwhelming. So, practice P8 merely suggests writing sentences in a consistent fashion. This helps separate user requirements from other information in the requirements document.
4.9. Practice P9: Sentences Should Have Clear Boundaries

Practice P9 suggests that sentences should have clear boundaries. This can be achieved by separating sentences using full stops or writing them in the form of a bulleted list (of complete sentences) for example. This practice helps address the problem of absent sentence boundaries in Arabic user requirements as discussed in Section 3.2.

4.10. Practice P10: Decompose a Sentence Involving Logical Operators into Multiple Sentences Provided It Is Logically Decomposable

Practice P10 suggests decomposing a sentence involving logical operators into multiple sentences provided it is logically decomposable. Some research studies claim that the presence of the word "and" in a sentence indicates the presence of two requirements and so should be decomposed into two separate sentences [8]. This is not always the case. For example, the sentence "The user shall press the Ctrl, Alt and Delete keys to activate the Task Manager" should not be decomposed into two or more sentences. A better suggestion would be, as indicated in practice P10, to always try to decompose a given sentence involving logical operators into multiple sentences provided it is logically decomposable. For example, the sentence shown in example E14, which means "The user shall press Ctrl and C or Ctrl and Insert to copy selected text to the clipboard" can be very ambiguous.

It can be interpreted for example as "The user shall press ((Ctrl and C) or Ctrl) and Insert to copy selected text to the clipboard" rather than "The user shall press (Ctrl and C) or (Ctrl and Insert) to copy selected text to the clipboard".

Since this sentence is logically decomposable, it is better that we decompose it into two sentences such as the sentence shown in example E15, which means "The user shall press Ctrl & C to copy selected text to the clipboard" and the sentence shown in example E16, which means "The user shall press Ctrl & Insert to copy selected text to the clipboard". This reduces the inherent ambiguity introduced by logical operators and eliminates the need to use brackets.

4.11. Practice P11: Represent Requirements Involving Complex Constructs Using a Decision Tree

Practice P11 suggests representing requirements involving complex constructs using a decision tree. Complex constructs such as nested if-then-else can be a serious source of ambiguity and missing requirements. These are better represented using a decision tree [8] as indicated by practice P11. This helps clarify such a requirement by representing it appropriately.

4.12. Practice P12: A Parent Requirement is Represented as a Header

Practice P12 suggests representing a parent requirement as a header. This practice deals with the special case when there is a parent requirement explained by a set of children requirements. In this case, it is suggested to write the parent requirement as a heading [8]. This is much better than writing the parent requirement as a complete sentence, which can be very confusing. For example consider the following parent-children construct:

- The system shall allow copying and pasting text to and from the clipboard:
  - The user shall press Ctrl & C to copy selected text to the clipboard
Although this construct involves only two requirements, it may be interpreted as involving three if the parent is interpreted as a separate requirement. It is better and much clearer to re-write it as follows:

- Copying and pasting text:
  - The user shall press Ctrl & C to copy selected text to the clipboard
  - The user shall press Ctrl & V to paste copied text from the clipboard

4.13. Practice P13: It is Preferable to Use Annotations to Clarify Each Part of a Given Sentence

Practice P13 suggests using annotations to clarify each part of a given sentence. Some research studies [6] suggest using special templates to be able to recognize the role of each part of a given sentence (conditions, stimulus, response, agent, action, object, etc). Others suggest using annotations for this purpose [8]. Since this could be very cumbersome, practice P13 states that this is preferable. An example of this practice is the sentence shown in example E17, which means "The user [agent] shall press [action, stimulus] Ctrl & C [object] to copy [response] selected [modifying text] text to the clipboard [location adverb of copy]".


Practice P14 suggests using annotations to indicate priority. Requirements such as those in examples E15 and E16 can have relative priorities. Some organizations use separate words to indicate priority, but this could be overwhelming and confusing. Thus Practice P14 makes a better suggestion, that is explicitly annotating sentences to indicate priority [8]. So for example, the sentence in example E15 can be annotated as "high priority" and that in example E16 can be annotated as "medium priority".

4.15. Practice P15: When Using Plurals, Clarify Whether You Refer to Each Instance of the Plural Subject or to the Plural Subject as a Whole

Practice P15 suggests that when using plurals, we should clarify whether we refer to each instance of the plural subject or to the plural subject as a whole [11].

The sentence in example E18 means "Users shall obtain a monthly report". This sentence can mean "All users shall obtain a single monthly report" as shown in example E19 or "Each user shall obtain a separate monthly report" as shown in example E20.

As suggested by Practice P15, one of the two sentences in examples E19 and E20, for example, should be written instead of that in example E18 according to the intended meaning.
4.16. Practice P16: Include a Glossary of Used Terminology with Corresponding Definitions

Practice P16 suggests including a glossary of used terminology with corresponding definitions. Using two terms to refer to the same item can be very ambiguous if the two terms were interpreted as referring to two different items. Thus, practice P16 aims at maintaining consistency in the used terminology [6-8] via a glossary that has the following advantages:

- helps reduce ambiguity (since the meaning of each term is explained in the glossary)
- helps avoid inconsistency in the used terminology since asking users to include all used terminology in one single glossary will encourage them to avoid inconsistency to reduce effort
- helps alert the user in case of an inconsistency; the user will most probably be alerted when providing the same definition twice for two different terms

In Arabic user requirements, such a glossary has the additional advantage of reducing ambiguities due to the following factors:

- translated and transliterated named entities with non-standard spelling as discussed in Section 3.3.
- named entities without capitalization and punctuation as discussed in Section 3.4.
- the presence of homographs and absence of diacritics as discussed in Section 3.5.

4.17. Practice P17: Avoid A/B Constructs

Practice P17 suggests avoiding A/B constructs. A/B constructs such as "انًذخلاخ/انًخشجاخ", which means "input/output" can be very ambiguous [8]. It is not clear whether it means "input and output", "input or output", "input is similar to output" or "input is the opposite of output" for example. Thus, practice P17 suggests avoiding such constructs to avoid this inherent ambiguity.

4.18. Practice P18: Avoid Abbreviations Unless Defined in the Glossary

Practice P18 suggests avoiding abbreviations unless defined in the glossary. Abbreviations can be a source of ambiguity if not interpreted at all or not similarly interpreted by all readers [8]. Thus practice P18 suggests avoiding such abbreviations unless defined in the glossary.

4.19. Practice P19: Avoid Using Pronouns, Possessive Pronouns and Possessive Adjectives If Ambiguous

Practice P19 suggests avoiding using pronouns, possessive pronouns, and possessive adjectives if ambiguous. As mentioned before, anaphoric ambiguity is a serious problem that results from the use of an expression that needs another expression in context to be interpreted. This applies to pronouns, possessive pronouns, and possessive adjectives that can cause ambiguities when used without clarifying what they refer to (also referred to as referential ambiguity [28] or the implicature problem [29]). Thus, practice P19 suggests avoiding their usage if ambiguous.

Consider, for example, the sentence in example E21 which means "He shall open his file". In this sentence, it is not clear to whom the pronoun "He" refers. It may be replaced by the sentence shown in example E22, which means "The user shall open his file". In this later example, it is clear that possessive pronoun "his" refers to the user and so may
not be avoided. But, it is better to re-write it as shown in example E23, which means "The user shall open the file of the user". Though this is not considered a good writing style in English as well as in Arabic, in user requirements, reducing ambiguities when writing a sentence is more important than a good writing style. This is also justified by the fact that the possessive pronoun "his" can refer to another person (or even another item in Arabic) in context.

4.20. Practice P20: Avoid Terminology with Subjective Meanings

Practice P20 suggests avoiding terminology with subjective meanings. Another suggested best practice in the literature is avoiding adverbs, adjectives and verbs whose meaning is subjective such as "quickly", "easy to use" and "support" [8, 29-30] and the same practice has been adopted for the Arabic language. Thus, a list of common words with subjective meanings should be developed. Examples include: "خفض" meaning "minimizing", "سرع" meaning "maximizing", "تحسن" meaning "improving", "سيط" meaning "optimizing", "سسهل الاستخدام" meaning "user-friendly", "بسيط" meaning "simple", "ضياع" meaning "often", "طبيعي" meaning "normal", "معادلة" meaning "usual", "كبيرة" meaning "large", "قوي" meaning "robust", "محسن" meaning "improved", "يسهول" meaning "efficient", "يدعم" meaning "support" and "مرن" meaning "flexible".

4.21. Practice P21: Place a Limiting Word Directly Preceding the Term or Item It Refers to

Practice P21 suggests placing a limiting word directly preceding the term or item it refers to. A common source of ambiguity is misplacing the word "only" and other limiting words such as "merely" and "also" [11]. Limiting words in Arabic include " فقط" meaning "only", "مجرد" meaning "merely", "كذلك" meaning "also" and "أيضا" meaning "too". Thus, practice P21 suggests placing such a limiting word directly preceding the item or term it refers to. The sentence in example E24 means "The user shall open one file only".

In this example, it is not clear whether the sentence means "The user shall only open one file" (the only thing the user can do is opening one file) as shown in example E25 or "The user shall open only one file" (the user can open no more than one file) as shown in example E26.

One of the two sentences in examples E25 and E26 (for example) should be written instead of that in example E24 according to the intended meaning. In these sentences the word "only" is written directly before the term or item it refers to. This is also adopted in example E13.

4.22. Practice P22: Place an Adjective Such That It Is Clear Which Item or Term It Refers to

Practice P22 suggests placing an adjective such that it is clear which item or term it refers to. Similar to limiting words discussed in Section 4.21, adjectives can be a source of ambiguity [20]. The sentence in example E27 means "The user shall open the second subsystem file".
### Table 1. Suggested Best Practices for Avoiding the Introduction of Ambiguities while Writing Arabic Software User Requirements

<table>
<thead>
<tr>
<th>Practice category</th>
<th>Practice statement</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language</strong></td>
<td></td>
</tr>
<tr>
<td>P1:</td>
<td>Stick to Modern Standard Arabic</td>
</tr>
<tr>
<td>P2:</td>
<td>Use English technical words within Arabic sentences</td>
</tr>
<tr>
<td><strong>sentence style</strong></td>
<td></td>
</tr>
<tr>
<td>P3:</td>
<td>Write short sentences as much as possible</td>
</tr>
<tr>
<td>P4:</td>
<td>Write complete sentences with correct grammar and spelling</td>
</tr>
<tr>
<td>P5:</td>
<td>Write only active-voice sentences</td>
</tr>
<tr>
<td>P6:</td>
<td>Write sentences of the form subject-verb-object</td>
</tr>
<tr>
<td>P7:</td>
<td>Avoid negative (or inverse) requirements as much as possible</td>
</tr>
<tr>
<td>P8:</td>
<td>Write sentences in a consistent fashion</td>
</tr>
<tr>
<td>P9:</td>
<td>Sentences should have clear boundaries</td>
</tr>
<tr>
<td><strong>complex sentences</strong></td>
<td></td>
</tr>
<tr>
<td>P10:</td>
<td>Decompose a sentence involving logical operators into multiple sentences provided it is logically decomposable</td>
</tr>
<tr>
<td>P11:</td>
<td>Represent requirements involving complex constructs using a decision tree</td>
</tr>
<tr>
<td>P12:</td>
<td>A parent requirement is represented as a header</td>
</tr>
<tr>
<td><strong>clarification</strong></td>
<td></td>
</tr>
<tr>
<td>P13:</td>
<td>It is preferable to use annotations to clarify each part of a given sentence</td>
</tr>
<tr>
<td>P14:</td>
<td>Use annotations to indicate priority</td>
</tr>
<tr>
<td>P15:</td>
<td>When using plurals, clarify whether you refer to each instance of the plural subject or to the plural subject as a whole</td>
</tr>
<tr>
<td>P16:</td>
<td>Include a glossary of used terminology with corresponding definitions</td>
</tr>
<tr>
<td><strong>avoided words</strong></td>
<td></td>
</tr>
<tr>
<td>P17:</td>
<td>Avoid A/B constructs</td>
</tr>
<tr>
<td>P18:</td>
<td>Avoid abbreviations unless defined in the glossary</td>
</tr>
<tr>
<td>P19:</td>
<td>Avoid using pronouns, possessive pronouns, and possessive adjectives if ambiguous</td>
</tr>
<tr>
<td>P20:</td>
<td>Avoid terminology with subjective meanings</td>
</tr>
<tr>
<td><strong>word placement</strong></td>
<td></td>
</tr>
<tr>
<td>P21:</td>
<td>Place a limiting word directly preceding the term or item it refers to</td>
</tr>
<tr>
<td>P22:</td>
<td>Place an adjective such that it is clear which item or term it refers to</td>
</tr>
<tr>
<td><strong>double checking</strong></td>
<td></td>
</tr>
<tr>
<td>P23:</td>
<td>Rephrase each sentence at least twice</td>
</tr>
</tbody>
</table>
In this example, it is not clear whether this means that "The user shall open the second file of the sub-system" as shown in example E28 or "The user shall open the file of the second sub-system" as shown in example E29.

One of the two sentences in examples E28 and E29 (for example) should be written instead of that in example E27 according to the intended meaning. It should be noted that this would be hard in case of multiple consecutive adjectives and may be handled using annotations as suggested by Practice 13 discussed in Section 4.13.

4.23. Practice P23: Rephrase Each Sentence at Least Twice

Practice P23 suggests rephrasing each sentence at least twice. As mentioned in Section 1, some ambiguities can be easily detected when a sentence has two possible meanings. Other types of ambiguities cannot be even realized when each reader interprets only one meaning for a given sentence, but the interpretation differs from one reader to the other. Sources of ambiguities especially ambiguous words (such as words with subjective meaning) are too many to be captured and too cumbersome to be addressed. Thus, practice P23 suggests rephrasing each sentence in the user requirements at least twice.

Though this suggestion might seem awkward, it can be justified by the fact that, typically, when we hear a sentence that we do not understand, we ask the speaker to rephrase it. This alerts the speaker to address possible sources of ambiguities in the sentence and to rephrase it clarifying the ambiguities. Similarly, when we write a sentence that we feel might be ambiguous, we typically rephrase it in other words. A sentence written as shown in example E27 can be rephrased, for example, as shown in example E28 or E29 according to the intended meaning. In this case, the sentence in example E27 would be disambiguated.

5. Human Subject Testing Results & Discussion

The goal of human subject testing was to assess the effectiveness of the proposed practices. All the practices have been considered in the test except practices P3, P8 and P14 for the following reasons:

- Practice P3 is concerned with shortening long sentences. In the Arabic language, a long sentence is a very good opportunity for the introduction of different sources of ambiguities. Consequently, to disambiguate a long sentence, it is not enough to shorten it, but other relevant practices have to be applied too. Thus, it is very hard to assess this practice in isolation.

- Practice P8 has to do with consistently writing the different classes of requirements to separate them from other information in the requirements document and so cannot be assessed in isolation.

- Practice P14 deals with the special case of specifying priorities.

In this test, three questionnaires have been prepared. The first questionnaire Q1 included sample sentences violating the proposed practices; two corresponding to each practice. Each sentence was followed by a number of multiple choice questions to try to figure out the understood meaning. Participants were allowed to select one or more
answers or to write down any alternative understood meaning. More than one answer or a wrong answer indicated ambiguity.

The second questionnaire Q2 was similar to the first one, but the practices were applied. In other words, each practice was applied to its two corresponding sentences in Q1 in an attempt to disambiguate them.

Practices P13 (using annotations) and P23 (rephrasing sentences) were not included in Q1 and Q2. This is because they are relatively general practices that can be applied to many ambiguous sentences to disambiguate them. So, to test the effect of the other practices in isolation, we postponed using these two practices to the third questionnaire Q3. In other words, Q3 was similar to Q2, but in Q3 practices P13 and P23 were applied to all partially disambiguated sentences of Q2.

A group of 20 Arabic-speaking participants in the age range of 18 to 45 was formed. They were given the questionnaires one after the other. After solving the three questionnaires, the group members were further interviewed to make sure they selected the answers they really understood and did not make random selections. The percentage of ambiguities encountered in the three questionnaires are shown graphically in Figure 1. The analysis and discussion of the results are provided in the following sub-sections.

5.1. Analysis of Results

Paired samples t-test was applied to the corresponding percentage of ambiguities in Q1 and Q2 since the difference was reasonable symmetric around the mean. Our hypothesis $H_0$ was that there is no difference between the percentage of ambiguities in the two questionnaires. We obtained a $t$-value of 10.28. Using this $t$-value and degrees of freedom of 17, we get a $p$-value of $1.03552 \times 10^{-08}$. Thus, with confidence of 95%, we reject the null hypothesis. In other words, there is a significant difference in the percentage of ambiguities between Q1 and Q2.

5.2. Discussion of Results

Violating practice P1 (stick to Modern Standard Arabic (MSA)) in Q1 resulted in about 97% ambiguity that was reduced to 0% after obeying it in Q2. The reason for this is that in Q1, we utilized a Saudi Arabian dialect while the questionnaire was given to Egyptian participants. One participant was able to figure out the correct meaning since her grandmother was Saudi Arabian and spoke this dialect. By rewriting the sentences using MSA, the meaning was clarified to all participants.

Practice P2 was very useful because most of the participants were more aware of the English names of technical words than the Arabic ones. For example, the word "administrator" is translated to "المدير" in Arabic. This latter word is very uncommon and could not be realized by most of the participants.

Practice P7 suggests avoiding negative (or inverse) requirements as much as possible. It was noticed that a single negation such as that shown in example E12 can be understood by participants with some effort, but double negation is worse. A double negation is shown in example E30, which means "The user shall not communicate with another user unless the administrator did not instruct the system not to allow him." In addition to the confusion due to double negation, it is not clear who of the two users is not allowed to communicate. It was also noticed that it is very hard to rephrase a sentence with double negation and eliminate negation totally. In other words, in case of double negation, negation seems to be unavoidable.

E30 : المستخدم سوف لا يتصل بمستخدم آخر إلا لو كان المدير لم يوجه النظام بمنع السماح له بذلك.
Composite and complex sentences addressed in practices P10, P11 and P12 are inherently ambiguous and need much effort from the users to decide their (possibly incorrect) meaning.

The following was found to be overly ambiguous and lead to confusion and uncertainty: plurals, using two terms to refer to a single item, A/B constructs, undefined abbreviations, pronouns, possessive pronouns, possessive adjectives, and terminology with subjective meanings.

Practices P21 and P22 by themselves may not be sufficient to fully address ambiguities. Practice P21 addresses the ambiguity of limiting words such as the word "only". This practice suggests placing a limiting word directly preceding the term or item it refers to. But, it was noticed that unless the user is aware of this criterion, the sentence may still suffer from ambiguity. This is also true for adjectives addressed in Practice P22.

In case of the Arabic language, the practices themselves unless applied carefully, can lead to more ambiguity. For example, in the sentence in example E28, it is not clear whether the user shall open the file of or for the second subsystem. This is not the case in example E29 in which the sentence has been written carefully.

A major source of ambiguities are prepositional phrases such as those in the sentence shown in example E31, which means "The system shall open the window in the first subsystem in the second sub-system".
In this sentence, it is not clear whether the window will be open in the first sub-system, which is in the second sub-system or whether the window, which is in the first sub-system, will be open in the second sub-system. Such ambiguity could be handled using practices P13 (annotations) and/or P23 (rephrasing the sentence) in an intelligent way as shown in example E32, which means "The system shall open the window, which is in the first subsystem and this will be in the second subsystem" or as shown in example E33, which means "The system shall open the window and this will be in the first subsystem, which is in the second subsystem" according to the intended meaning.

---

**Questionnaire Q1:**
- The user shall open the second subsystem file

**Questionnaire Q2:**
- The user shall open the second file off/for the subsystem

**Questionnaire Q3:**
**After applying practice P23:**
- The user shall open the second file off/for the subsystem
- The user shall open the second file belonging to the subsystem

**After applying practice P13:**
- The user [agent] shall open [action] the second [modifying file] file [object] off/for the sub [modifying system] system [owner of the file]
- The user shall open the second file belonging to the subsystem
Another major source of ambiguity is the use of multiple consecutive adjectives as shown in the sentence in example E34, which means "The system shall open a colored graphics window". In this sentence, it is not clear whether the word "colored" refers to the "graphics" or to the "window".

This ambiguity could similarly be handled using practice P13 (using annotations) and/or using practice P23 (by rephrasing the sentence intelligently) as shown in example E35, which means "The system shall open a window for colored graphics" or as shown in example E36, which means "The system shall open a colored window for graphics" according to the intended meaning.

In the third questionnaire, not surprisingly, all sentences have been disambiguated. In summary, practices P13 and P23 have been found to be the most useful for disambiguating sentences (beside applying the relevant practices). The only problem is that applying these two practices require additional effort. Also, unless applied intelligently, they may not be of much effect. Besides, their application is subjective and differs from one user to another according to the realized ambiguity and the ability to annotate and rephrase sentences intelligently.

Figure 2 shows a sample sentence from the three questionnaires. Each sentence is accompanied by its English translation. After applying practice P22 in Q2, it was still not clear whether the open file was the file of the subsystem or was open for the subsystem. This was clarified in Q3 through the use of Practices P13 and P23.

6. Conclusions

This paper is concerned with practices needed to avoid the introduction of ambiguities when writing Arabic software user requirements. Towards this goal, we studied practices suggested by practitioners in the literature for the English language but are general enough to be used for or tailored to the Arabic language. We also studied the main sources of ambiguities in the Arabic language that would affect user requirements. This is in addition to interviewing a number of Arabic software engineers. Accordingly, a set of practices for writing Arabic user requirements have been suggested. In other words, the suggested practices are based on both practical experience and theoretical background. We provided 36 examples to justify the effectiveness of the suggested practices. We also emphasized this further via human subject testing and empirical evaluation.

This research study is original since, to the best of our knowledge, this is the first paper that addresses avoiding the introduction of ambiguities when writing Arabic user requirements, which is a serious problem in the middle east hindering the development of quality software. This paper is the first step towards a comprehensive set of practices. Practical application of the suggested practices and further research may lead to discovering additional practices or rephrasing the current ones. As a future work, we intend to develop an automated tool that helps in the application of these practices. We also intend to study and address other sources of imprecision in the elicited user requirements such as inaccuracy, incompleteness and/or inconsistencies.
References


Authors

Hanan Elazhary, earned her B.Sc. and M.Sc. degrees from the Department of Electronics and Electrical Communications, Cairo University. She earned her Ph.D. degree in Computer Science and Engineering from the University of Connecticut, USA. Currently, she is an associate professor in the Computers and Systems Department, Electronics Research Institute, Cairo, Egypt.