From Contract Drafting to Software Specification: Linguistic Sources of Ambiguity

A Handbook

Version 1.0

Daniel M. Berry, Ph.D. Computer Science
dberry@uwaterloo.ca
Computer Systems Group
Computer Science Department
University of Waterloo
200 University Ave. West
Waterloo, Ontario N2L 3G1
Canada

Erik Kamsties, Ph.D. Computer Science
kamsties@sse.uni-essen.de
Software Systems Engineering
University of Essen
D-45117 Essen
Germany

Michael M. Krieger, Ph.D. Mathematics, J.D.
mkrieger239@earthlink.net
Computer Science Department, University of California at Los Angeles
and
Willenken Loh Stris Lee & Tran, 10920 Wilshire Blvd., Suite 150-68
both in
Los Angeles, CA 90024
U.S.A.

November 2003

Abstract

This handbook is about writing software requirements specifications and legal contracts, two kinds of documents with similar needs for completeness, consistency, and precision. Particularly when these are written, as they usually are, in natural language, ambiguity—by any definition—is a major cause of their not specifying what they should. Simple misuse of the language in which the document is written is one source of these ambiguities.

This handbook describes the ambiguity phenomenon from several points of view, including linguistics, software engineering, and the law. Several strategies for avoiding and detecting ambiguities are presented. Strong emphasis is given on the problems arising from the use of heavily used and seemingly unambiguous words and phrases such as “all”, “each”, and “every” in defining or referencing sets; positioning of “only”, “also”, and “even”; precedences of “and” and “or”; “a”, “all”, “any”, “each”, “one”, “some”, and “the” used as quantifiers; “or” and “and/or”; “that” vs. “which”; parallelism; pronouns referring to an idea; multiple adjectives; etc. Many examples from requirements documents and legal documents are examined. While no guide can overcome the careless or indifferent writer, this handbook is offered as a guide both for writing better requirements or contracts and for inspecting them for potential ambiguities.

1 Introduction

Both software requirements specifications (SRSs) and contracts for business transactions and other human interaction share many properties, including the need to be precise and accurate, to be self-consistent, and to anticipate all possible contingencies. SRSs are usually written in natural language, often augmented or enhanced by information in other notations, such as formulae, and diagrams. Only occasionally, one finds a completely formalized SRS using little natural language, except as commentary. Virtually every initial conceptual document for a system is written in a natural language. Virtually every request for proposal is written in a natural language. A recent online survey of businesses requiring software, conducted at Università di Trento in Italy and available at online.cs.unitn.it [59], shows that a majority of documents available for requirements analysis are provided by the user or are obtained by interviews. Moreover,

- 71.8% of these documents are written in common natural language,
- 15.9% of these documents are written in structured natural language, and
- only 5.3% of these documents are written in formalized language,

Contracts, as well as a broader range of documents having legal force, are written entirely in natural language, with common augmentation of minor arithmetic, e.g., dealing with percentages for royalty or estate distribution. Only rarely is information provided in other notations, and then only as amplification or clarification of the natural language text. While we refer here to only contracts, in fact this discussion applies also to most any legal document that acts as a recipe or standard for performance. Thus, these issues apply equally to, for example, wills, deeds, and statutes. For economy of exposition, we call all of these documents "contracts", although a few examples involve other types of documents.

Unintended ambiguity is the Achille's heel of SRSs and contracts, leading to diverging expectations in both cases and inadequate or undesirably diverging implementations in the former. Unintended ambiguity admits multiple interpretations of the underlying document. An ambiguous SRS can lead to two or more implementors writing interfacing code to operate under different assumptions, despite each programmer's confidence that he has programmed the correct behavior. Likewise, an ambiguous contract can lead to two or more parties performing in conflicting ways, despite each party's confidence that he has followed the contract to the letter.

Because an SRS is often included as part of a software development contract, ambiguity in an SRS can be a double whammy, when it spawns both improperly performing software and litigation.

The ambiguity problem is exacerbated if the author of the document, an SRS or a contract, is unavailable for questioning when the document is being interpreted. Commonly, the requirements engineers who wrote an SRS move on to other projects and become unavailable for questioning by the implementors [48]. In one particular kind of legal document, the will, the author is eternally unavailable for questioning when its legal mandates are to be performed.

Don Gause [30] lists the five most important sources of requirements failure as:

- failure to effectively manage conflict,
- lack of clear statement of the design problem to be solved,
- too much unrecognized disambiguation,
- not knowing who is responsible for what, and
- lack of awareness of requirements risk.

It is instructive that unrecognized disambiguation ranks as high as the other, universally recognized sources of failure.

Unrecognized or *unconscious* disambiguation is that process by which a reader, totally oblivious to other meanings of some text that he has read, understands the first meaning that comes to mind and takes it as the only meaning of the text. That unconsciously assumed meaning may be entirely wrong.

We are not talking about deliberate, intentional ambiguity,[1] perpetrated to cause an SRS or contract to be misunderstood by someone or to be understood differently by several. Thus, the constructive ambiguity of a carefully mediated peace contract between erstwhile intractable enemies is not the subject of this handbook.

What are the sources of ambiguity in written documents? Certainly, common mistakes, failure to recognize technical terms or terms of art—either by the writer or the reader, and common misconceptions about the meanings of words and phrases contribute, especially when a term, word, or phrase has both a technical or legal meaning and an everyday meaning. However, we believe that a common source of ambiguity in a written document is poor use of the natural language used to write the document. The writer making a language error has conveyed a meaning other than that intended; the reader, in not recognizing an error, has understood a meaning not intended by the writer. Also, a writer, in using the language unusually, correctly ends up conveying a meaning other than that intended to a reader who is not aware of the correct usage of or of the correct meaning of the language.

This handbook describes a number of types of ambiguities that were found by the authors in the course of their work and in some case studies examining the requirements elicitation process and the contract drafting process. The first and second authors found some in the course of work helping customers to identify their requirements, and the fourth author found some in the course of writing contracts for his legal clients. Due to requirements for privacy, the kinds of ambiguity found are described with different examples, extracted from publically available sources.

Most of the examples used in this handbook are simple sentences, whose meanings change according to different interpretations. Interestingly, correct use of the natural language eliminates most of the ambiguities. However, both writers and readers are often so unaware of the correct use of the language, that even when the writer is being correct and precise, the readers do not pick up the intended meaning and misconstrue a sentence. Thus, a sentence can be unambiguous from a linguistic point of view, but be ambiguous from a pragmatic point of view. That is, the sentence has only one meaning according to language rules, but people, not fully aware of the rules, misinterpret the sentence or think that there are more than one meaning. For example, "I only saw one movie today." means that the sayer did not also hear the movie today, he only saw it. This sentence is a perfectly reasonable to say if one has refused to pay an airline's exorbitant fee for the ear phones to be able to listen to the in-flight movie. However, most think that the sentence means that I saw only one movie today, because of the common, but incorrect practice of always putting the "only" directly before the verb of the sentence.

The irony of all this is that

1. two people who know the rules well or
2. two people who do not know the rules well

communicate well, but

3. one person who does know the rules well and one person who does not know the rules well

do not communicate well. The two who know the rules communicate well because the writer writes with precision and the reader understands what he means. The two who do not know the rules communicate well, because both misuse the language in commonly accepted ways. Thus, the reader probably understands the intent of the writer. However, when one knows the rules and the other does not, there is a heightened chance for miscommunication.

---

[1]These are our favorite intentional ambiguities:

> This paper fills a much needed gap in the literature.
> I most enthusiastically recommend this candidate with no qualifications whatsoever.
> I cannot say enough good things about this candidate or recommend him too highly.
> Lose up to 20 lbs. in 10 weeks! Guaranteed!

When the one who knows the rules is the writer, the result may sound strange to the reader, e.g., “I am he.” or “It is I.” are correct responses to “May I speak to Joe?” even though most would say “I am him.” or “It’s me.” The result may even be misunderstood. In the other direction, The rule-knowing reader can guess the intended meaning, especially if the mistake is a common one. However, there may be lingering doubt. The problem is worse when the rule-knowing reader does not know how well the writer knows the rules and is left wondering whether the sentence means what it says because the writer knows the rules or something else because the writer does not know the rules. Of course, the presence of common mistakes elsewhere in the document gives a strong clue to the reader that the writer does not know the rules. Finally, the problem is exacerbated when the reader simply has no clue as to how well the writer knows the rules. This phenomenon can be called a pragmatic ambiguity arising from ignorance of extension and uncertainty of intention.

Section 2 describes commonalities and differences between software requirements specifications and legal contracts. Section 3 defines “ambiguity” from several viewpoints. Section 4 reviews a number of techniques for detecting ambiguities in natural language documents. Section 5 is the heart of this handbook and it focuses on *avoiding* linguistic, lexical, structural, and referential ambiguities that occur in natural language requirements specifications and contracts. Section 6 applies these avoidance techniques to a number of example requirement specifications and legal documents. Section 7 reviews other writing guides for their advice on avoiding ambiguity. Section 8 concludes this handbook.

Some of the problems are language specific, while some are language independent. Therefore, the examples are given not only in English, but also in Portuguese, French, German, and Hebrew, respectively, as representatives of the Latin, German, and Semitic languages.[2] Sometimes a problem in one does not exist in others. We are not concerned here with ambiguities coming about as a result of translation from one language to another, but of potential ambiguities faced by native speakers of the languages as they work in their own languages. When mathematical notation makes the meaning of the sentence perfectly clear, an additional sentence in predicate calculus is provided.

Rather than surrounding all examples and words used as themselves with quotation marks, this handbook adopts a convention of typesetting examples and some words used as themselves in a sans-serif font, like these four words, while leaving the general narrative in a serif font, like these four words.

Each example has a reference number that is not part of the example, and this is typeset in the serif font. When an example numbered *E* has a small number of lines, generally in different languages, then line number *L* is addressed by *E:L*.

Ambiguities occur in different language units ranging from a single word on up through complete books. We use the term “text” to stand for a language unit of any size that has an ambiguity. In most of the examples, the ambiguous text is one sentence.

In order to avoid using variants of “he or she” as a third person singular subject pronoun of nondeterminate gender, only “he” or only “she” is used in any one text. The gender of the third person singular alternates by section, starting with “he” in Section 1. Of course, if an example that we are quoting has a variant of he or she, it will be left unchanged.

[2] In early drafts of this handbook, this intention may not be fully implemented. If you wish to volunteer to help us by providing missing translations, please contact the author Berry at dberry@uwaterloo.ca. We are seeking also to include translations in Chinese and Japanese. If you volunteer to provide these translations, contact Berry at the same address.

When a language uses a non-Latin alphabet, we give the give the examples in that language transliterated into the Latin alphabet to allow those who do not read the original alphabet a better chance to observe the patterns. We beg forgiveness of those who do read the original alphabet for not using that alphabet.

2 Commonalities and Differences

As mentioned in Section 1, SRSs and legal contracts have a number of commonalities including

1. the requirement to be precise and accurate, to be self-consistent, and to anticipate all possible contingencies, and
2. the fact that most such documents are written mostly in natural language.

Beyond these commonalities there are commonalities and differences in the contents and structure of the documents and in the processes by which the documents are generated and maintained.

2.1 Software and System Requirements Specifications

By "software requirements specifications (SRSs)", we mean specifications for mainly the software component of computer-based systems (CBSs). Strictly speaking, such a specification may deal with non-software, system issues. However, it is usual to talk about these specifications as software specifications, mainly because the most changeable part of a CBS is its software. The specifications state what can be assumed of the environment and what must be done by the system. In legal terminology, what can be assumed of the environment are the *responsibilities* of the environment and the *rights* of the system. Likewise, what must be done by the system are the *responsibilities* of the system and the *rights* of the environment.

2.2 Kinds of Legal Writing

The basic kinds of legal writing include

1. contracts,
2. trusts and wills, and
3. statutes

A contract constitutes requirements for at least two parties; it defines the rights and responsibilities of each party. The interactions between the parties can be viewed as interfaces. A trust or will constitutes requirements for disposition of one party's money, either before or after death. Observe that after death, it is difficult to verify intent of the author of the requirements. A statute constitutes requirements on all members of a society, be they individuals or organizations.

A contract, often written in the press of commercial activity, sometimes suffer from hasty drafting and less review. A trust or will is usually written more leisurely and can be more carefully reviewed. A statue is developed more slowly, undergoing greater review, as part of the deliberative legislative process. The distinction is not unlike that between manually proved theorems for homework or program verification as opposed to manually proved theorems for published mathematical papers [18]. Because proofs for publication are subjected to far more scrutiny and peer review than proofs for homework or program verification, published theorems are more reliable than homework or program verification theorems.[3]

The focus of this handbook is on contracts as a representative genre of legal documents. The purpose of a contract is similar to that of a requirements specification, that is, to describe completely the expected behavior of a group of humans or of a CBS under all foreseeable circumstances and contingencies.

2.3 Sources of Information

Both kinds of documents are written by one or more professionals in the software or law field, for a client who is not in the field. Occasionally, a professional in the relevant application domain participates in the writing. Both kinds of writing require the professional to elicit information from the client's representatives and to observe, interview, and negotiate with them. Both require the client's representatives to validate that the professionals have captured the client's intent.

[3] However, incorrect proofs, and even more rarely, incorrect theorems do get published, just as incorrect statutes get enacted.

2.4 Changes

A common and pervasive problem with software specifications is that of change. Requirements always change [46], and keeping up-to-date and consistent the requirements specification and all other documents that must be kept consistent with the requirements specification is a major problem and is the subject of a whole branch of SE, *change management*. For software, change seems relentless all through the lifecycle.

While the human endeavors covered by a contract may indeed suffer changes, the rate, urgency, pervasiveness, and unexpectedness of the changes are generally not as large as in SE. Moreover, contracts typically include a clause specifying a procedure by which the parties may agree to change the contract or by which there are new negotiations to cover unforeseen events that invalidate contract clauses or otherwise require changes. In practice, however, changes to contracts is not so nearly as big a problem as changes to software requirements.

An indication of the difference in impact of changes to software and contracts is the special clause that is very common in contracts that specifies that if an arbitrary clause in the contract is invalidated, the rest of the contract still holds. The though of applying such an idea to software makes any programmer chuckle. If one part of a program fails, generally the whole program comes to a screeching halt, because each part of a program is connected to all other parts. In law, the hope is that this clause is workable, that the parties can find meaningful ways to apply the still valid clauses. However, in some cases, such a special clause carries the seeds of its own contradiction, as it is sometimes impossible to decompose the contract into an invalid part and a totally independent valid part. Often there are logical implications from one clause to another to the extent that the whole contract is invalidated by a single invalidated clause. This removal problem is not unlike the problem of throwing out or modifying an incorrect line of code without affecting the rest of the program. Just as a whole program can be broken by one invalid character of the code or by the change of one character of the code, a whole contract can be broken by one invalid clause or by the change of one clause.

2.5 Fixing Errors

In SE, it is known that the cost to fix a detected error goes up exponentially with the lifecycle stage [12]. That is, an error that costs X (where X is any measure of work or resources) to fix during requirements specification, costs about $10{\times}X$ to fix during coding and about $200{\times}X$ to fix after deployment. Therefore, it pays handsomely to find errors during requirements specification, so handsomely that it is worth spending a significant fraction of the total budget on requirements specification, e.g., even as much as 90%. Nevertheless, it is rare to spend more than 25% of the total budget on requirements specification because there is a perception that no one on the project is really working until the implementation of frozen requirements begins. The irony here, of course, is that requirements change so relentlessly that “frozen requirements” is an oxymoron.

In Law too, it costs almost nothing to change clauses during contract drafting. Changes after signing causes invocation of the contract’s change procedure, requiring several people’s review and approval, and in very severe cases, renegotiation of the contract. The additional review and approval costs time and money. Moreover, if the breakdown necessitating the changes extends so far that the contract cannot be fixed by simple changes to contract clauses, then the parties may end up litigating, which is astronomically expensive.

3 Definitions of Ambiguity

Ambiguity is a real-world phenomenon that rears its ugly head in many disciplines including writing, linguistics, philosophy, law, and—of course—software engineering, especially requirements engineering. We consider several definitions of ambiguity, namely from an English dictionary, from software engineering, from linguistics, and from the law. We classify the kind of ambiguity that is the subject of this handbook according to each definition. We also limit the discussion to natural language text, although it is clear that semi-formal and even formal descriptions are open to ambiguities, albeit a more limited number.

Note that the term “ambiguity” should probably be “multiguity” to remove the implication, arising from the prefix “ambi”, that there are only two meanings. However, we stick to the conventional word with the understanding that we are really talking about multiguities.

3.1 Dictionary Definition

The Merriam Webster English Dictionary [2] defines “ambiguity” as

1a) the quality or state of being ambiguous especially in meaning,
1b) an ambiguous word or expression, or
2) uncertainty,

where “ambiguous” means

1a) doubtful or uncertain especially from obscurity or indistinctness <eyes of an ambiguous color>,
1b) inexplicable, or
2) capable of being understood in two or more possible senses or ways.

Thus, there are two basic interpretations of “ambiguity”,

1) the capability of being understood in two or more possible senses or ways and
2) uncertainty.

Uncertainty in the sense of lack of sureness about something is ignored here, because this lack of sureness has to do with the writer’s and reader’s knowledge about the background; the requirement or contract sentence itself could be precise and clear.

3.2 Software Engineering Definition

There appears to be no single comprehensive definition of ambiguity in the software engineering literature. Each of the following definitions highlights only some aspects of ambiguity and omits others. The definitions together form a complete overview of the current understanding of ambiguity in SE.

The IEEE Recommended Practice for Software Requirements Specifications [43] says that “An SRS is unambiguous if, and only if, every requirement stated therein has only one interpretation.” Presumably, an SRS is ambiguous if it is not unambiguous.

The problem with the IEEE definition is that there is no unambiguous specification simply because for any specification, there is always someone who understands it differently from someone else, just as there are no bug-free programs [65]. There *are* mature, usable programs whose bugs are known; the users have learned to work around the bugs and get useful computation from them. In a similar manner, there are no unambiguous specifications. Rather, there are useful specifications, each of which is understood well enough by enough people that count, a majority of the implementors, a majority of the customers, and a majority of the users, that it is possible to implement software meeting the specifications that does what most people expect it to do in most circumstances.

Indeed, Davis [17] has suggested a test for ambiguity: “Imagine a sentence that is extracted from an SRS, given to ten people who are asked for an interpretation. If there is more than one interpretation, then that sentence is probably ambiguous.” The problem with this test is that, as in software testing, there is no guarantee that the eleventh person will not find another interpretation. However, this test does capture the essence of a useful SRS that is unambiguous for most practical purposes. Actually, we would go farther and say that the sentence *is* ambiguous. Davis provides two examples of ambiguity.

(E1) For up to 12 aircraft, the small display format shall be used. Otherwise, the large display format shall be used.

Assuming that small and large display formats are defined previously, the ambiguity lies in the phrase for up to 12. Does it mean for up to and including 12 or for up to and excluding 12?

(E2) Aircraft that are non-friendly and have an unknown mission or the potential to enter restricted airspace within 5 minutes shall raise an alert.

Assuming again that the relevant terms are defined, the ambiguity lies in the relative precedence of and and or, because we cannot assume the precedence rules of Boolean algebra for natural language utterances.

We believe that the first interpretation of the dictionary definition, the capability of being understood in two or more possible senses or ways, is underlying Davis's discussion of ambiguity.

Schneider, Martin, and Tsai [71] give another definition of ambiguity. "An important term, phrase, or sentence essential to an understanding of system behavior has either been left undefined or defined in a way that can cause confusion and misunderstanding. Note, these are not merely language ambiguities such as an uncertain pronoun reference, but ambiguities about the actual system and its behavior."

In addition to the IEEE definition, Schneider, Martin, and Tsai identify two possible categories of ambiguity, namely language ambiguities and software engineering ambiguities. This handbook deals with language ambiguities, i.e., ambiguities that can be spotted by any reader who has an ear for language. In contrast, software engineering ambiguities depend on the domain involved and can be spotted only by readers that have sufficient domain knowledge. Parnas, Asmis, and Madey give an example of a software engineering ambiguity. Their example is a requirement that was introduced without the reader's having been told that the water level that is the subject of the requirement varies continuously [64].

(E3) Shut off the pumps if the water level remains above 100 meters for more than 4 seconds.

This sentence has at least four interpretations.

1. Shut off the pumps if the *mean* water level over the past 4 seconds was above 100 meters.
2. Shut off the pumps if the *median* water level over the past 4 seconds was above 100 meters.
3. Shut off the pumps if the *root mean square* water level over the past 4 seconds was above 100 meters.
4. Shut off the pumps if the *minimum* water level over the past 4 seconds was above 100 meters.

The software engineers building the pump did not notice this ambiguity and quietly assumed the fourth interpretation. Unfortunately, under this interpretation, with sizable, rapid waves in the tank, the water level can be dangerously high without triggering the shut off under this interpretation. In general, the interpretation of the ambiguity is very much a function of the reader's background. For example, in many other engineering areas, the standard interpretation would be the third. We, the authors of this handbook, actually did not see any ambiguity at all and assumed the fourth interpretation. We did not know that there were waves in the tank, because the text leading up to the example made no mention of waves. If the water were quiescent except when water is being added and its top surface were always essentially flat, all four interpretations would be identical in meaning, and the fourth interpretation provides the easiest implementation. Clearly, domain knowledge is need to know that there are waves and thus, that the third interpretation is needed.

According to Gause and Weinberg, ambiguity has two sources, missing information and communication errors [29]. Missing information has various reasons. For instance, humans make errors in observation and recall, tend to leave out self-evident and other facts, and generalize incorrectly. Communication errors that occur between the author and the reader are due to expression inadequacies in the writing.

This handbook deals with ambiguities caused by expression inadequacies. Gause and Weinberg [29] give an example of an ambiguity due to missing information.

(E4) Create a means for protecting a small group of human beings from the hostile elements of their environment.

They provide three interpretations:

1. an igloo (an indigenous home constructed of local building materials),
2. a Bavarian castle (a home constructed to impress the neighbors), and
3. a space station (a mobile home with a view).

The reason for these wide-ranging interpretations of the requirement is that a lot of issues are missing, namely the cost, material, size, shape, and weight of the means of protection, the other functions that shall be performed inside this means, and the nature of the environment. Moreover, the phrase *small group* is an example of an expression inadequacy; are we talking about 5, 20, or 100 people?

The view of Gause and Weinberg is shared by Harwell, Aslaksen, Hooks, Mengot, and Ptack [39] in their definition of an unambiguous requirement. "A requirement must be unambiguous in the sense that different users (with similar backgrounds) would give the same interpretation to the requirement. This has two aspects. On the one hand there is the aspect of grammatical ambiguousness, i.e. the poorly constructed sentence. On the other hand there is the aspect of ambiguousness arising from a lack of detail allowing different interpretations. The first of these can be measured or tested independently of author or user, but the second aspect can be measured only in conjunction with a set of users, since it depends on what assumptions the user makes automatically, i.e. as a result of the user's background knowledge."

A glance through the requirements engineering literature shows that different interpretations of ambiguity are in actual use. Some, including Davis [17] and Parnas, Asmis, and Madey [64] argue that ambiguity is not acceptable at all, because of the disastrous consequences that misinterpretations can have in software design and implementation. Here, the term "ambiguity" is interpreted as ambiguity caused by expression inadequacy, as in Example E4

Others, including Goguen [33] and Mullery [60], and Gause [30] argue that ambiguity is acceptable for a while, as are inconsistencies [20, 21]. Here, the term "ambiguity" is interpreted as ambiguity caused by missing information, as in Example E4. Goguen points out that "although natural language is often criticized, e.g. by advocates of formal methods, for its informality, ambiguity and lack of explicit structure, the features can be advantages for requirements. For example, these features can facilitate the gradual evolution of requirements, without forcing too early a resolution of conflicts and ambiguities that may arise from the initial situation; it is important not to prejudge the many trade-offs that will have to be explored later, such as cost versus almost everything else .... And finally, natural language can permit the 'diplomatic' resolution of conflicts through the careful construction of deliberate ambiguities; for example this is rather common in large government financed projects."

Mullery points out that a certain degree of ambiguity, and incompleteness and inconsistency as well, must be accepted over a period of time, which may well extend through into design and implementation stages. An unambiguous SRS is unattainable, because requirements are constantly changing. The quest for an unambiguous SRS, what is usually desired for contractual software development, costs additional effort. However, the total effort for producing a SRS is defined in the delivery contract; penalties for failure to deliver the SRS in time are specified. Thus, the additional effort to achieve a lack of ambiguity must be estimated carefully, keeping in mind the inevitable changes, both to the domain and to the system functionality. Attempting to achieve an unambiguous SRS results in far more work whenever a change is needed, or even worse, it makes keeping abreast of needed changes totally infeasible. Don Gause says that conflict is good because it exposes problems with the requirements [30]. On the other hand, too much unconscious ambiguity leads to failure, because people make unconscious design decisions or make conscious design decisions based on unconscious design assumptions, all based on unconscious disambiguation based on their initial reading.

In summary, there are two major types of ambiguities, language ambiguities and software engineering ambiguities. Some authors consider only expression inadequacy as source of ambiguity, others consider missing information as an additional source. That is, "ambiguity" can mean two slightly different things; *ambiguity* is ambiguous!

3.3 Linguistic Definitions

Linguistic ambiguity is investigated in linguistics [56] and in related fields, namely, computational linguistics [40, 4] and philosophy [55, 79]. This section presents a summarized view of ambiguity from these fields. First, the types of ambiguity considered in linguistics, computational linguistics, and philosophy are discussed. Second, related phenomena, such as vagueness, are described.

One can distinguish four broad classes of linguistic ambiguity, lexical ambiguity, syntactic ambiguity, semantic ambiguity, and pragmatic ambiguity. Note that this classification is not mutually exclusive. Rather, the ambiguity or ambiguities occurring in a single ambiguous text may be a combination of several kinds.

3.3.1 Lexical Ambiguity

Lexical ambiguity occurs when a word has several meanings. Lexical ambiguity can be subdivided into homonymy and polysemy.

Homonymy occurs when two different words have the same written and phonetic representation, but unrelated meanings and different etymologies, i.e., different histories of development. For example, the etymology of bank in the sense of an establishment for custody, loan, exchange, or issue of money is, according to Merriam-Webster's dictionary [2],

> Middle English, from Middle French or Old Italian; Middle French *banque*, from old Italian *banca*, literally bench, of Germanic origin; akin to Old English *benc*

while the etymology of bank in the sense of a rising ground bordering a lake, river, or sea is

> Middle English, probably of scandinavian origin; akin to Old Norse *bakki* bank; akin to Old English *benc* bench—more at BENCH Date: 13th century

Polysemy occurs when a word has several related meanings but one etymology. For example, green has several different, but related, meanings with a common etymology, according to Merriam-Webster's dictionary, such as

1: of the color green
3: pleasantly alluring
4: youthful, vigorous
5: not ripened or matured

Systematic polysemy occurs when the reason for the polysemy is confusion between classes, e.g., between unit and type, and between process and product [16]. As an example of unit-vs.-type ambiguity, I like this jacket can refer to an individual jacket or to a type of jacket. Process-vs.-product ambiguity occurs with everyday words like building, shot, and writing. The noun form of any of these can refer to a process or to the product of the same process. Another systematic polysemy is the behavior-vs.-disposition ambiguity. For example, This is a fast car can denote the current behavior of a particular car or the general capability of each element of a class of cars, such as Ferrari, independent of the current behavior of any particular Ferrari.

3.3.2 Syntactic Ambiguity

Syntactic ambiguity, also called *structural ambiguity*, occurs when a given sequence of words can be given more than one grammatical structure, and each has a different meaning. In the terminology of compiler construction, syntactic ambiguity occurs when a sentence has more than one parse. A syntactic ambiguity can be distinguished as an analytical, attachment, coordination, or elliptical ambiguity.

1. *Analytical ambiguity* occurs when the role of the constituents within a phrase or sentence is ambiguous.

Among the various patterns of analytical ambiguity that can occur is the structure of a complex noun group including modifier scope. For instance The Tibetan history teacher can be read as The (Tibetan history) teacher or The Tibetan (history teacher). More patterns are described by Hirst [40].

2. *Attachment ambiguity* occurs when a particular syntactic constituent of a sentence, such as a prepositional phrase or a relative clause, can be legally attached to two parts of a sentence. The most popular pattern of attachment ambiguity is a prepositional phrase that may modify either a verb or a noun. For instance, in the sentence

   (E5) The police shot the rioters with guns.

   the phrase with guns can be taken either as a modifier of the noun rioters or as a modifier of the verb shot, leading to two different interpretations: either the rioters were armed with guns when the police shot them, or the police used guns to shoot the rioters. More patterns are described by Hirst [40].

3. *Coordination ambiguity* occurs
   1. when more than one conjunction, and or or, is used in a sentence or
   2. when one conjunction is used with a modifier.

   An instance of the first kind is the sentence:

   (E6) I saw Peter and Paul and Mary saw me.

   This sentence can be read either as I saw (Peter and Paul) and Mary saw me or as I saw Peter and (Paul and Mary) saw me. A well-placed comma disambiguates a sentence like this one.

   An instance of the second kind is the phrase:

   (E7) young man and woman

   It can be read either as (young man) and woman or as young (man and woman).

4. An ellipsis is a gap in a sentence cause by omission of a lexically or syntactically necessary constituent. *Elliptical ambiguity* occurs when it is not certain whether or not a sentence contains an ellipsis. An example is:

   (E8) Perot knows a richer man than Trump.

   It has two meanings, that Perot knows a man who is richer than Trump is and that Perot knows a man who is richer than any man Trump knows. The first meaning corresponds to having no ellipsis, and the second corresponds to having an ellipsis, which is the implied knows coming just after Trump.

3.3.3 Semantic Ambiguity

Semantic ambiguity occurs when a sentence has more than one way of reading it within its context although it contains no lexical or structural ambiguity. Semantic ambiguity can be viewed as ambiguity with respect to the logical form, usually expressed in predicate logic, of a sentence. Semantic ambiguity can be caused by

1. coordination ambiguity,
2. referential ambiguity, and
3. scope ambiguity.

Coordination ambiguity is already discussed. Referential ambiguity is on the borderline between semantic and pragmatic ambiguity, because referential ambiguity can happen within a sentence or between a sentence and its discourse context. Thus, referential ambiguity is discussed in the section on pragmatic ambiguity.

The *quantifier operators* include such words as every, each, all, some, several, a, etc., and the *negation operators* include not. *Scope ambiguity* occurs when these operators can enter into different scoping relations with other sentence constituents. For example, in the sentence

(E9) All linguists prefer a theory.

the quantifiers all and a interact in two ways. When the scope of a includes the scope of all, this sentence means all linguists love the same one theory. When the scope of all includes the scope of a, this sentence means that each linguist loves a, perhaps different, theory.

Negations and quantifiers can interact ambiguously. For example, the sentence

(E10) **No one** has seen **a** pig with wings.

can be read as saying either that there exists no pig with wings or that there exists a mythical pig with wings that no one has ever seen. Another, attachment, ambiguity exists in this sentence, namely whether the pig has wings or whether wings are the instrument by which one sees pigs. In this case, domain knowledge about seeing organs tells the reader that wings are not used for seeing and that the first reading is the right one.

3.3.4 Pragmatic Ambiguity

Pragmatics is the study of the relations between language and context [55]. The research area of pragmatics has linguistic and philosophical roots. The scope of pragmatics, i.e., the phenomena that are considered pragmatic, and its boundary with semantics are subjects of ongoing discussions. In general, both pragmatics and semantics investigate the meaning of language; pragmatics is concerned with context-dependent meaning, while semantics is concerned with context-invariant meaning. In particular, phenomena, such as *deixis* discussed below, often straddle the semantics–pragmatics boundary.

Pragmatic ambiguity occurs when a sentence has several meanings in the context in which it is uttered. The context comprises the language context, i.e., the sentences uttered before and afterw, and the context beyond language, i.e., the situation, the background knowledge, and expectations of the speaker or hearer and the writer or reader. We distinguish referential ambiguity and deictic ambiguity. There are more topics of pragmatics, such as conversational implicature, speech acts, and conversational structure, which we do not discuss, because they do not concern contracts and requirements documents.

In traditional semantics, the relation between a word or phrase and the object of the real world that the word or phrase describes is called a *reference*. An *anaphor* is an element of a sentence that depends for its reference on the reference of another element, possibly of a different sentence. This other element is called the *antecedent* and must appear earlier in the same sentence or in a previous sentence. A *referential ambiguity* occurs when an anaphor can take its reference from more than one element, each playing the role of the antecedent. Anaphora include pronouns, e.g., it, they, definite noun phrases, and some forms of ellipses. An example of a referential ambiguity is:

(E11) The trucks shall treat the roads before **they** freeze.

The antecedent of the anaphor they can be either trucks or roads. Anaphora can refer also to sets of objects, compound objects, or verbs.

Ellipses can have the same effect as pronouns and definite nouns, as the following sentence shows.

(E12) ... If the ATM accepts the card, the user enters the PIN. If **not**, the card is rejected.

The word not is here an elliptical expression that refers either to the condition specified in the previous sentence or

to something written before that.

Deictic ambiguity occurs when pronouns, time and place adverbs, such as now and here, and other grammatical features, such as tense, have more than one reference point in the context. The context includes a person in a conversation, a particular location, a particular instance of time, or an expression in a previous or following sentence. An example of a pure deictic ambiguity is not given here for reasons of space, because giving it would require introducing a context first. In contrast to an anaphor, a deictic or another definite expression is often used to *introduce* a referent. An anaphoric reference is used to refer to the same entity thereafter. Note that a pronoun, in particular, can be anaphoric or deictic. When a pronoun itself refers to a linguistic expression, or chunk of discourse, the pronoun is deictic; when a pronoun refers to the same entity to which a prior linguistic expression refers, the pronoun is anaphoric [55]. One can draw an analogy to the pointer concept of the C programming language; a deictic expression is a pointer and an anaphoric expression is a pointer to a pointer. An ambiguity occurs also when a pronoun can be read as an anaphoric or as a deictic expression, as shown in the following example, which has has elements of scope, referential, and deictic ambiguities.

(E13) Every student thinks she is a genius.

When the scope of the quantifier every includes she, she refers, as an anaphoric expression, to student, which is thus known to be feminine. When the scope of the quantifier every does not include she, she refers, as a deictic expression to a contextually specified specific female, who may or may not be one of the students.

3.3.5 Vagueness and Generality

Vagueness and *generality*, also called *indeterminacy*, are closely related to ambiguity. Cruse distinguishes a general word or phrase from an ambiguous word or phrase [16]. Each is open to more than one interpretation. However, they have different statuses in human communication as shown by cousin and bank in the following sentences:

(E14) Sue is visiting her cousin.

(E15) We finally reached the bank.

Cousin, in the first example, can refer to either a male or a female cousin. However, the sentence can function as a satisfactory communication without either the hearer perceiving, or the speaker intending to convey, anything concerning the gender of the referent, because cousin has a general meaning. The general meaning covers all the specific possibilities, not only with regard to gender, but also with regard to an unbounded number of other matters, such as height, age, eye color, etc.

Also Bank, in the second example, can be interpreted in more than one way, as we have seen before, as a river bank or as a money bank, but it has no general meaning covering these possibilities. Furthermore, the interpretation cannot be left undecided. Each of the speaker and hearer must select a meaning. Moreover, they must select the same reading if the sentence is to play its role in a normal conversational exchange.

Cousin is *general* with respect to gender. Bank is *ambiguous*. In other words, the two meanings, male cousin and female cousin, are associated with the same word cousin, whose meaning is more general than that of either; they do not represent distinct senses of cousin. On the other hand, the meanings of bank are distinct.

A statement is considered *vague* if it admits *borderline cases* [5]. For example, tall is vague, because a man of 1.8 meters in height is neither clearly tall nor clearly not tall. No amount of conceptual analysis and empirical investigation can settle whether a 1.8-meter man is tall. Moreover, tallness is culture relative. A person regarded as tall in a pygmy tribe would be considered short in an NBA basketball team. Borderline cases are inquiry resistant, and the inquiry resistance recurses. That is, in addition to the lack of clarity of the borderline cases, there is lack of clarity as to where the lack of clarity begins [75].

Linguistic vagueness has a special application in software requirements. A requirement is vague if it is not clear how to measure whether the requirement is fulfilled or not. A non-functional requirement, e.g., fast response time, is often vague, because there is no precise way of describing and measuring it, short of arbitrary quantification, which leaves us wondering if the essence of the requirements have been captured.

The only difference between vagueness and generality is the existence of borderline cases. A vague expression is inquiry resistant because of borderline cases, while a general expression can be made more precise. For example, cousin is general, but can be described more precisely when necessary. Certainly, there is no doubt as to whether or not a person is a cousin. Moreover, there is no doubt as to which cousins are male and which are female. That is, cousin is not vague.

The subjects of this handbook are

- lexical, syntactic, and semantic ambiguities caused by poor use or understanding of the language of the requirements document,
- only referential and similar pragmatic ambiguities, and
- vagueness.

Generality is not covered in this handbook, because its apparent ambiguity is relative to the expectations of the writer and reader.

3.3.6 Language Error

Our experience has identified another category of pragmatic ambiguity, *language error*. As is the case with the categories described in Section 3.3.5, language error may not be mutually exclusive of other categories. A language error ambiguity occurs when a grammatical, punctuation, word choice, or other mistake in using the language of discourse leads to text that is interpreted by a receiver as having a meaning other than that intended by the sender.

For example,

(E16) Every light has their switch.

has a grammatical error that is commonly committed by present-day, even native, English speakers. The error is that of considering every *X*, which is singular, as plural although it precedes a correct singular verb, as in

(E17) Everybody brings their lunch.

In the case of Example E16, the reader does not know if the intended meaning is

(E18) Every light has its switch.,

that is

(E19) Each light has its switch.,

or is

(E20) All lights have their switch.,

which could mean either of:

(E21) All lights share their switch.

(E22) Each light has its own switch.

Basically, because of the error, the reader does not know how many switches there are per light.

Many times, a language error ambiguity is at the same time another kind of ambiguity, especially an extension versus intention ambiguity. The sender does not know an error has been committed, and the receiver may or may not know that an error has been committed. If the receiver does not know, she may or may not understand it as intended. If she does know, she may or may not be able to make a good guess as to what is intended, but in the end she may be left wondering.

The reason this new category is needed is that sometimes there is a language error, but no extension versus intention ambiguity. Sometimes, there is a linguistic mistake only if the intention is one way but not if it is another way. For example, in

(E23) Everybody brings their lunch.

everyone knows that the intended meaning is

(E24) Everybody brings his lunch.

even though their, being plural, is incorrectly used with the singular Everybody; here we have a language error without an extension versus intention ambiguity. In

(E25) I only smoke Winstons.

if the intention is to say,

(E26) I smoke only Winstons.

there is the language error of a misplaced only. However, if the intention is to make it clear, in an admittedly strange conversation about eating Winston cigarettes, that one only smokes and does not eat Winstons, then there is no language error. However, someone not privy to the whole conversation, and hearing only Example E25, may understand Example E26, which would be contrary to the intention, even though the intention is in fact what is said by the sentence, according to the rules about placement of only.

3.4 Legal Definition

A good source of information of the treatment of ambiguity in the law is Black's Legal Dictionary [10], which is the source for this section. The legal definition of ambiguity is an operational one describing what to do with an ambiguity when it happens rather than what it is. The linguistic source and type of ambiguity seems to be irrelevant other than to establish that there *is* an ambiguity. Then comes the real work to sort out the responses of the parties to the ambiguity. Because of the different focus, the categories of ambiguities in law are different from those in linguistics.

The law, at least in Great Britain, the U.S., and for that matter, any place whose laws are based on British law, has a tradition of interpreting an ambiguity against the perpetrator of the ambiguity. Expressed in Latin, *AMBIGUUM FACTUM CONTRA VENDITOREM INTERPRETANDEM EST.*, that is, "An ambiguous contract is to be interpreted against the seller." Illustrating the same principle are some other Latin sentences: *AMBIGUUM PLACITUM INTERPRETARI DEBIT CONTRA PROFERENTUM.*, that is, "An ambiguous plea ought to be interpreted against the party pleading it.", and *AMBIGUITAS CONTRA STIPULATOREM EST.*, that is, "Doubtful words will be construed most strongly against the party

using them."[4] Thus, the writer of a contract is strongly encouraged to write the contract as clearly and precisely as possible so that even if not everyone agrees on the desirability of the status quo expressed by the contract, everyone agrees at least on the meaning of the contract.

In building its definition of ambiguity, Black's defines some terms of art. "Ambiguity upon the *FACTUM*" expresses the idea that an ambiguity in relation to the very foundation of an instrument is distinguished from an ambiguity in regards to the construction of its terms. The kinds of ambiguities described in this document are of the second kind. As shall be seen, the authors intend to be precise, but lack of skill with the language of the contract prevent them from stating exactly what they mean, even though what they mean may be quite clear and agreeable to all parties.

AMBIGUITAS VERBORUM LATENS VERIFICATIONE SUPPLETUR; NAM QUOD EX FACTO ORITUR AMBIGUUM VERIFICATIONE FACTI TOLLITUR.. That is, "A latent ambiguity in the language may be removed by evidence; for whatever ambiguity arises from an extrinsic fact may be explained by extrinsic evidence." The remedy of supplying evidence would probably be the remedy for all of the cases here. Each ambiguity in this handbook is an inadvertent ambiguity arising from lack of skill with the language of the document. The writer would be asked what he meant and the reply would be taken as the evidence.

Finally, Black's defines ambiguity as "Doubtfulness; doubleness of meaning [cite].[5] Duplicity, indistinctness, or uncertainty of meaning of an expression used in a written instrument [cite]. Want of clearness or definiteness, difficult to comprehend or distinguish; of doubtful import [cite]." This definition is followed by comments in fine print: "Ambiguity of language is to be distinguished from unintelligibility and inaccuracy, for words cannot be said to be ambiguous unless their signification seems doubtful and uncertain to persons of competent skill and knowledge to understand them [cite]." This observation appears not to be applicable to the kinds of ambiguity presented in this handbook unless one says that these ambiguities arise from linguistic incompetence. However, probably a majority of native English speakers are incompetent in the ways described herein. "It does not include uncertainty arising from the use of peculiar words, or of common words in a peculiar sense [cite]." The question to be asked is whether correct use of only is considered peculiar to most observers, while the statistically normal, but incorrect way of using only is considered acceptable and clear to most observers.

"[Ambiguity] is *latent* where (sic) the language employed is clear and intelligible and suggests but a single meaning, but some extrinsic fact or extraneous evidence creates a necessity for interpretation or a choice among two or more possible meanings, as where a description apparently plain and unambiguous is shown to fit different pieces of property [cite]." In each of the examples of this handbook, the language employed is clear and intelligible, and it has only one meaning. However, the extrinsic fact of linguistic incompetence on the part of the writer or the reader creates other meanings, either intended or unintended.

"A *patent* ambiguity is that which appears on the face of the instrument and arises from the defective, obscure, or insensible language used. [cite]" Do writer's linguistic mistakes constitute defective, obscure, or insensible language? The irony is that if the writer and reader are incompetent in exactly the way, the meaning received may very well be the meaning sent. A problem arises when the writer and reader have different competences with the language of the document. In such a case, the meaning received may not be the meaning sent.

[4]Interestingly, Black's adds, *AMBIGUIS CASIBUS SEMPER PRÆSUMITUR PRO REGE.*, that is, "In doubtful cases, the presumption always is in favor of the crown." Presumably, in a non-monarchy, the sentence is generalized to refer to the state. It is not clear whether this sentence is saying that the state is allowed to write an ambiguous contract and still have its interpretation stand or it is saying that in the event of a dispute over meaning by two non-state parties, the state gets the spoils of that dispute. Here, we have a pragmatic ambiguity based on an uncertainty. If sufficient context or missing information were available, the uncertainty would disappear.

[5]Here and elsewhere, "[cite]" means that the original cites a specific precedent-setting court case or a well-known legal scholar. The name of the case or the scholar is not very relevant to this handbook.

One of the important roles of a contract is memory, in at least two different ways:

1. The recollections and expectations of the original parties diverge with time.
2. In business life, the original definers of the deal, and hence of the contract, move on and are replaced by people who do not know the deal except from the written documentation.

When ambiguity is discovered in a contract, it is construed against the drafter. To prevent an application of this doctrine against one party to a contract, many contracts carry a clause naming *all* parties of the contract as joint drafters. With such a clause, a truly ambiguous clause would not be construed against any one party. It would end up being invalidated and would become the subject of renegotiation under the contract's specified change procedure. Moreover, as mentioned, many contracts carry another clause declaring that if any part of the contract is invalidated, the rest of the contract still applies.

4 Techniques for Dealing with Ambiguity

This section presents techniques discovered in related research for reducing the level of ambiguity in natural language requirements. Techniques can be divided into three groups according to the requirements engineering activities in which they are applied, namely requirements elicitation, requirements documentation, and requirements validation.

For *requirements elicitation*, at least two strategies can be distinguished to minimize ambiguity. First, a context must be established, because language is interpreted always in context, and if this context is not made explicit and agreed to by all the stakeholders in an elicitation session, misinterpretations are likely. Second, the requirement engineer's paraphrasing what she understood from the customers' and users' statements in her own words is an effective way for the requirements engineer to get the customers and users to spot their own ambiguities. Several communication techniques support these strategies [8, 13].

For *requirements documentation*, at least three strategies can be distinguished to avoid ambiguity in the written requirements. First, as described in detail in Sections 5, 6, and 7, the precision of natural language can be increased. Second, more contextual information can be provided in order to allow the reader to resolve ambiguities herself. Third, conventions on how ambiguous phrases shall be interpreted can be set up between the writer and the reader. These strategies are the topics in the remainder of this section.

For *requirements validation*, at least four strategies can be distinguished to detect ambiguities. First is the formalization of informal requirements [74, 80]. Since a formal language enforces precision, the hope is that ambiguities can be exposed in the process of being as precise as is necessary for formalization. Second is searching for particular patterns of ambiguity, as done in reading techniques for requirements inspections [29, 6, 47, 49]. Some natural language processing tools help also to detect ambiguities [44, 31, 32, 23, 58]. Third is comparing the interpretations of a document by different stakeholders; if they differ, there is an ambiguity in the original document. Fourth is communicating an interpretation back to the requirements author, after which she can easily point out misinterpretations. These last two are perhaps the most effective strategies for finding ambiguities in requirements specifications, but they demand more resources than other strategies and are therefore applicable only in situations in which the cost of not finding ambiguities is at least the cost of the resources required to find the ambiguities. For example, Easterbrook and Callahan applied the third strategy using SCR as the language for expressing an interpretation for validating part of the requirements for the International Space Station [22].

To illustrate the various strategies for minimizing ambiguities in requirements documentation, the requirement

(E27) An aircraft that is non-friendly and has an unknown mission or the potential to enter restricted airspace within 5 minutes shall raise an alert.

is used. It suffers from a semantic ambiguity regarding the precedences of and and or.

4.1 Documentation: Increasing the Precision of Natural Language

Glossaries, style guides, sentence patterns, and controlled languages increase the precision and decrease the ambiguity of natural language. A glossary or dictionary defines important terms and phrases used in a requirements document. Thus, it helps avoid lexical ambiguity. It requires considerable effort to create and validate a glossary, but the initial effort pays off since it can be reused for future projects within the same application domain. For example, Kovitz gives detailed guidelines for creating a glossary [52]. Designations by Jackson [45] and the Language Extended Lexicon by Leite [54, 38, 50] are other techniques for grounding terms in reality.

A style guide is a set of rules on good practices in requirements writing. Style guides have been developed to help requirements authors in writing requirements [14, 63, 52, 70, 25, 37]. They can be used also during requirements analysis and validation to check the requirements for possible problems. An example of a rule that is found in many style guides is "Use active voice rather than passive voice.", because passive voice blurs the actor in a requirement. A rule that solves the ambiguity problem in Example E27 is "Insert braces to avoid syntactic and semantic ambiguity." Applying this rule to the example yields

(E28) An aircraft that is non-friendly and (has an unknown mission or the potential to enter restricted airspace within 5 minutes) shall raise an alert.

which makes clear that or binds stronger than and.

Sentence patterns have been proposed to give the requirements author support in articulating requirements by Rolland and Proix [68] and Rupp and Goetz [69]. An example of a pattern, written in extended Backus-Naur form, for activities that are performed by a system without user interaction, as in the running example, is

(E29) [when?] [under what conditions?] THE SYSTEM SHALL | SHOULD | WILL <process> <thing to be processed> [<process detail>*]

The requirement must be rewritten slightly to fit the pattern:

(E30) If [an aircraft is non-friendly] and [has an unknown mission or the potential to enter restricted airspace within 5 minutes], the system shall <raise> <an alert>.

Notice that the and–or ambiguity is resolved in the same way as suggested by the rule to use braces.

A controlled language is a precisely defined subset of natural language for the use in specific environments. The objective of a controlled language is to increase the readability and understandability of any kind of technical documentation. This improvement is accomplished by reducing the inherent ambiguity of natural language through a restricted grammar and a fixed vocabulary. There is a genuine need for tool support for writing requirements documents in order to enforce the grammar and the fixed vocabulary of the controlled language. Several controlled languages with tool support have been proposed for RE. The most recent ones are Attempto Controlled English (ACE) by Fuchs and Schwitter [26, 27] and the CREWS Scenario Authoring Guidelines by Ben Achour [7]

4.2 Documentation: Providing more Context Information

Comments, rationales, fit criteria, test cases, inverse requirements, and traceability information support the strategy of providing more context information. A *comment* can be used to explain the background of a requirement. A comment that would allow the reader to disambiguate Example E27 is:

(E31) Comment: A non-friendly aircraft is acceptable as long as it has a known mission or is more than 5 minutes away.

A *rationale* describes why a requirement is needed. Like a comment, a rationale can help disambiguate a requirement.

A *fit criterion* describes a condition that a software product must fulfill in order to satisfy a requirement. A *test case*, a more elaborated form of a fit criterion, describes a possible input and its expected output explicitly. Each fit criterion thus provides contextual information and leaves less room for interpretation. Possible test cases for Example E27 are:

(E32) *Input*: Friendly aircraft.
Output: No alarm.

(E33) *Input*: Non-friendly aircraft and enters airspace within 5 minutes.
Output: Raise alarm.

An *inverse requirement* describes functionality that the software product does not perform. Inverse requirements are often misused to express non-functional requirements, e.g., the system must not lose user data, which is actually a reliability requirement. However, in its essence, an inverse requirement rules out possible interpretations of one or more functional requirements. Thus, the inverse requirement disambiguates the functional requirements. Consequently, a true inverse requirement has no test case. If one can derive a test case for an inverse requirement, the inverse requirement is probably actually a non-functional requirement. An inverse requirement specifically helps to reduce pragmatic ambiguity, generality, and vagueness, as in the following example:

(E34) Requirement: The vending machine offers refreshing drinks.

(E35) Inverse requirement: The vending machine does not offer tea, coffee, and alcoholic drinks.

Therefore, most likely, the vending machine offers soft drinks, i.e., carbonated drinks, fruit juices, and water. The inverse requirement reduces the ambiguity of refreshing drinks by eliminating some potential meanings.

Traceability information on the dependencies between requirements, i.e., requirements–requirements traceability, also helps to disambiguate a requirement, if the links help identify closely related requirements that provide enough contextual information [36].

A tool called the Requirements Apprentice, developed by Reubenstein and Waters, tries to automatically disambiguate a new requirement by putting the requirement into a relation with existing requirements using the tool's pattern matching algorithm [66]. The proposed disambiguation is echoed by the tool to the user for user feedback and selection.

More information on augmenting a requirement with a comment, a rationale, a fit criterion, and a test case, and on inverse requirements and traceability is offered by Sommerville and Sawyer [74] and by Robertson and Robertson [67].

4.3 Documentation: Setting Up Conventions for Interpretation

There are no specific techniques for the strategy of setting up conventions, because this strategy is very pragmatic. Following this strategy, a convention in the case of the example requirement could be:

(E36) When a logical condition has more than one of "and" or "or", then the precedence rules of predicate logic are to be followed.

This convention must be clear to both the writer and the reader. Otherwise, misinterpretations can occur. In Example E27, the requirement would have to be rewritten to communicate the correct interpretation.

(E37) An aircraft that is non-friendly and has an unknown mission or **that is non-friendly and has** the potential to enter restricted airspace within 5 minutes shall raise an alert.

The bold-faced text was added to force the correct meaning on the assumption that and binds tighter than or.

Generally, ambiguity and lack of precision are major problems of natural language requirements. Thus, these problems should be addressed during all requirement engineering activities, from elicitation through validation. During documentation, a combination of all three strategies should be followed to help minimize ambiguity.

In the rest of this handbook, we provide support for unambiguous specification of requirements. More precisely, our aim is to increase the precision of natural language requirements by a providing style guide that touches on issues covered at best only cursorily by other style guides.

5 Avoiding Ambiguities in Natural Language Specifications and Contracts

This section covers a variety of common linguistic, lexical, structural, scope, referential, and language-error ambiguities that appear in both requirements specifications and legal contracts. In each case, we

1. give a succinct example of a general problem,
2. give common solutions to the general problem, applied to the succinct example,
3. describe drawbacks of the common solutions,
4. offer a correct solution to the general problem, again applied to the succint example,

Then, in some cases, we discuss the general principles, if any, involved, perhaps at length. Somewhere in the discussion, we classify the ambiguity by one or more of the classifications described in Section 3. In any given case, one or more of these items may be omitted as not being relevant to the discussion.

5.1 Ambiguous, Vague, and Uncertain Words

The Oxford English Dictionary says that the 500 most used words in English have on average 23 meanings.[6] We restrict the following discussion to those words that are appear to be precise, but actually are not. The lack of precision can be due to ambiguity, vagueness, or uncertainty.

The following commonly used words are ambiguous and should be avoided or used with care in a requirement or contract sentence:

- and: It is used to denote concurrency of events or actions, it is used to denote that several conditions are to be met, it is used to denote a temporal order of events or actions, and it is used in an enumeration implying no order at all. The problem can be fixed by splitting a sentence containing it into several sentences, if possible, or writing what is really meant, using at the same time for concurrency of events or actions, using and then for temporal order, thus reserving the simple and to denote that several conditions are to be met concurrently.
- any: Writers may intend it to denote plurality, while readers generally interpret it to denote oneness, particularly since grammatically it is singular. The problem can be fixed by using it only as readers expect it and as the grammar requires it.
- include: It can mean consists of or contains as a subset. The problem can be fixed by writing what is meant.
- after, before, next, previous, etc.: Each of these indexical words has a fixed meaning but a variable reference. The problem can be fixed by careful use accompanied by verification of uniqueness of referent.
- minimum and maximum: The words minimum and maximum leave open whether the specified interval is open or closed, That is, *x* is the minimum can mean either no less than *x* or greater than *x*, and *x* is the maximum can mean either no greater than *x* or less than *x*. The problem can be fixed by writing what is meant.

[6] Cited after Jeff Gray, http://www.gray-area.org/Research/Ambig/.

- or: It can mean either one or the other, but not both, i.e., the exclusive OR of logic, or one or the other or both, i.e., the inclusive OR of logic. The problem can be fixed by writing what is meant, i.e., either ... or for the exclusive OR and either ... or ... or both for the inclusive OR.

Vague words should be avoided by precise quantifications. There is a large number of vague adjectives and adverbs: acceptable, accurate, appropriate, easy, efficient, essential, immediately, minimum, maximum, periodically, sufficient, user-friendly, etc. Also verbs and nouns can be vague; support, handle, process, reject, use, etc. and user. The following example of a vague requirement that appears precise at first glance was provided by Davis [17].

(E38) For up to 12 aircraft, the small display format shall be used. Otherwise, the large display format shall be used.

Assuming that small and large display formats are defined previously, the ambiguity lies in the phrase for up to 12. It can mean for up to and including 12 or for up to and excluding 12. The sentence should be rewritten to have the fuller term.

(E39) For up to and including 12 aircraft, the small display format shall be used. Otherwise, the large display format shall be used.

Uncertainty is expressed using words such as not limited to, etc., can, may, probably, possibly, usually, etc. These words should be avoided in favor of a complete list, a complete decision, an accurate probability estimate, etc. If a fact is unknown or uncertain at writing time, the need for further investigation must be stated explicitly, so that the reader knows of the uncertainty, its degree, and the intent to remove it.

5.2 Quantification

Quantification terms are natural language terms that are equivalent to the universal quantifier, ∀, and the existential quantifier, ∃, of mathematics. We consider first the universal quantifier equivalents and then the existential quantifier equivalents.

5.2.1 All, Each and Every

All, each, and every are universal quantifier equivalents in that each is used to describe properties that hold for all members of some set. In any sentence containing a universal quantifier, there is a danger that the sentence is not true, simply because very few universal statements about the world have no exceptions [9]. This handbook focuses on structural problems affecting the ability to convey intended meanings independently of whether or not the meaning is true.

Consider the sentence

(E40) All persons have a unique national insurance number.

We ignore here the issue of whether or not the sentence is true, that is, we assume that there is no one with no or more than one national insurance number. The problem with this sentence is that in the absence of domain knowledge, it is not clear whether

1. each person has his own unique national insurance number,

or

2. all persons share a common unique national insurance number.

In other words,

All persons have a unique national insurance number.

is ambiguous because it leaves the sayer's intention uncertain. If one writes,

(E41) Each person has a unique national insurance number.

the intent is clear and the ambiguity is avoided. Example E40 is a classical example of scope ambiguity; it is not clear which quantifier, all or a, takes precedence over the other. Mathematics shows the problem very clearly. The two meanings of this sentence are

(E42) $\exists! x \, \forall y$, x is the national insurance number of y

(E43) $\forall y \, \exists! x$, x is the national insurance number of y

Granted, in this case, a tiny bit of domain knowledge tells the reader that the intended meaning of Example E40 is that each person has his own unique national insurance number. However, replace national insurance number with a nonsense word, about which there is no domain knowledge, e.g., qwertyuiop. Then the meaning of

(E44) All persons have a unique qwertyuiop.

is not clear. To be clear, one should write one of,

(E45) Each person has a unique qwertyuiop.

(E46) All persons share a unique qwertyuiop.

For sentences with all that are ambiguous even with everyday domain knowledge consider,

(E47) All persons practice their religion.

(E48) All persons practice their religions.

The imprecision of Example E47 is that it is not clear whether the one religion

① is per person or
② is shared by all persons.

The imprecision of Example E48 is that it is not clear whether

③ a single person practices one religion that may be different from another one's only religion,
④ a single person practices more than one religion, or
⑤ all persons share the same, possibly several, religions.

The sentences below, given in the same order as the meanings above marked by circled-numbers, are clearer renditions of Examples E47 and E48.

(E49) Each person practices his religion.

(E50) All persons practice their common religion.

(E51) Each person practices his own religion.

(E52) Each person practices his religions.

(E53) All persons practice their common religions.

according to his intended meaning; the correspondence to an intended meaning is shown by a circled digit. In any case, each of Examples E47 and E48 should be avoided as sufficiently imprecise.

Thus, proper choice of words in an all, each, or every sentence is used to convey the cardinality of the relation between the subject and the object, that is whether the relation is 1–1, 1–*n*, *n*–1, or *n*–*m*.

(E54) Each practices his religion. is 1–1,

(E55) Each practices his religions. is 1–*n*,

(E56) All practice their religion. is *n*–1, and

(E57) All practice their religions. is *n*–*m*.

Observe that being precise about meaning of the relations and cardinality of the relations is easy when mathematical notation with its quantifiers and logical variables are used. The mathematical expressions for the sentences of Examples E54, E55, E56, and E57 are, respectively.

(E58) $\forall x\ (\exists! r\ (x \text{ practices } r))$

(E59) $\forall x\ (\exists! R\ (x \text{ practices } r) \supset r \in R)$

(E60) $\exists! r\ (\forall x\ (x \text{ practices } r))$

(E61) $\exists! R\ (\forall x\ (x \text{ practices } r) \supset r \in R)$

An example of an ambiguous requirement caused by this sloppiness is

(E62) All lights in the room are connected to a single switch.

Does it means that all lights in the room share a single switch? Does it mean that each light in the room has its own single switch? It officially says the first, but most people would think it means the second. The careful writer might be aware and write it meaning the second. The careful reader might be aware and wonder if the writer were equally aware.

In the discussion below, for any use of all, let *Dall* be the set of all objects quantified by the word all. For example, in the sentence,

(E63) All persons have a unique national insurance number.

Dall is the set of all persons of one nation. The grammatical basis for the syntactic problem with all is that all is plural, and when it is used in or as the subject of a sentence, the verb of the sentence must be plural. Consequently, the complement of the sentence, i.e., the object or predicate of the sentence, must also be plural. When the complement is singular, the grammatical implication is that the complement applies to the entire *Dall* and not to each member of *Dall*.

The simplest way to write a sentence describing a property of each member of *Dall* is to use each or every, which is grammatically singular. (Recall that everybody means every single body.) When it is used in or as the subject of a sentence, the verb of the sentence must be singular. Consequently, the complement of the sentence must also be singular. In this case, when the complement is singular, the grammatical implication is that the complement applies to each member of *Dall*.

The simplest way to be precise about intention is to use each when the intention is to talk about properties of each member of *Dall* and to use all when the intention is to talk about properties of the whole set *Dall*.

Unfortunately, at least in English, many people use every as plural. Moreover, these same people are sloppy about maintaining number and gender across the verb from an every-laden subject to the object or predicate. An example of an extremely annoying *incorrect* construction involving every treated partially as plural is[8]

(E64) Everybody has their national insurance number.

with the intended meaning of

(E65) Each has his national insurance number.

The utterer of the sentence has mixed number for Everybody. That is, he uses a singular verb, a singular object, but a plural possessive. When asked about the their, the typical utterer says that everybody is plural, but when challenged about the clearly singular verb has, he sees the error and says something to the effect that their is a gender-free politically correct possessive that does not sound as sexist as the grammatically correct but politically incorrect his or as clumsy as the grammatically and politically correct His or her.

Interestingly, the problem does not arise in one of the other languages, specifically in Portuguese Portuguese. In this language, the possessive must agree with the possessed, while in English, Brazilian Portuguese, French, German, and Hebrew, the possessive must agree with the possessor. Actually, in German, the situation is a bit more complicated. The base word of the possessive agrees with the possessor, but the endings agree with the possessed, as illustrated by the following variations of Each has his national insurance number (NIN).

(E66) Each has his NIN.
Jeder hat seinen Sozialversicherungsnummer.

(E67) She has her NIN.
Sie hat ihren Sozialversicherungsnummer.

(E68) He has his NIN.
Er hat seinen Sozialversicherungsnummer.

(E69) Each has his NINs.
Jeder hat seine Sozialversicherungsnummern.

(E70) She has her NINs.
Sie hat ihre Sozialversicherungsnummern.

(E71) He has his NINs.
Er hat seine Sozialversicherungsnummern.

The biggest problem with this incorrect construction is the resulting language error ambiguity when the incorrect plural becomes the subject for a subclause. Then we get the same imprecision as with Examples E47 and E48 about the correspondence between instances of the subject and instances of the object or predicate.

(E72) Everybody says that they practice their religion

(E73) Everybody says that they practice their religions.

Adding to the ambiguity is that each of these sentences may or may not be grammatically correct. It is grammatically correct if in some nearby previous sentence, a plural noun is introduced describing people about whom everybody wants to discuss religious practices. If no such previous sentence exists, then the sentences are grammatically incorrect.

[8]In some of the languages, the sentence is *so* incorrect that it is *never* said!

The correspondence problem of Examples E47 and E48 occurs whenever a plural noun referring to members of a set is used as the subject in a sentence. Consider the two structurally identical sentences:

(E74) Students enroll in six courses per term.

(E75) Students enroll in hundreds of courses per term.

Domain knowledge tells us that the first sentence is talking about each student while the second is talking about the whole set of students. Since the first sentence is talking about each student, it should be written in singular form, as one of

(E76) Each student enrolls in six courses per term.

(E77) A student enrolls in six courses per term.

The first is more for stating the fact that *every* student enrolls in six courses per term, and the second is more for specifying a rule that a student should enroll in six courses per term. Using a singular formulation for talking about properties of each or any student reserves the plural formulation for talking about properties of the collection of students.

These syntactic problems with plural universal quantifier equivalents and with plural sentences are not restricted to English. For example, all of the above examples using all and each can be duplicated with the same meanings in French with tous and chaque, in German with alles and jeder, in Hebrew with col and col ekhad, and in Portuguese with todo and cada, respectively.

Interestingly, mathematics has adopted a convention that makes intent very clear. In mathematics, the universal quantifier $\forall$, read as for all is singular as in,

(E78) $\forall\, x \in Int, x < x+1$
For all Integers x, x is less than $x+1$.

Mathematical notation makes it easier to be unambiguous. However, sadly the ambiguity problems described in this section are often found in the mathematical text that accompanies formal statements of definitions, theorems, and proofs. Fortunately, the formal mathematical statements are usually correct and serve to disambiguate the incorrect or imprecise natural language text.

With adjectives, e.g. in the sentences

(E79) All humans are mortal.

(E80) Each human is mortal.

the correspondence problem may seem not severe, since the same adjective works for both singular and plural and mortality is a property that holds equally well for the individual or for the whole set. However, in some languages other than English, e.g., in French, an adjective for a singular noun is different from the same adjective for a plural noun:

(E81) Tous les hommes sont mortels.

(E82) Chaque homme est mortel.

The word both is a variant of the word all for referring to a set that has precisely two elements. Like all, both is plural, and it requires a plural predicate when it is used as a subject in a sentence. For example, at a post office, one of the authors of this handbook had two envelopes with identical contents to be sent to the same region of the world. Thus, he expected that the postage on each to be the same. After the clerk had weighed both envelopes, she said:

(E83) Both cost $1.50.

He, the customer, gave her $2.00, expecting change. She said that this amount was not enough, that the total is $3.00. Whereupon, he replied that she meant:

(E84) Each costs $1.50; both cost $3.00.

This real-life everyday example illustrates the danger of using plural nouns as sentence subjects. It is ambiguous as to whether the predicate is about the whole set denoted by the plural subject or about each member of the set denoted by the plural subject. Using each with a singular subject makes it crystal clear what is meant. Even in the customer's reply, the plural construction would be ambiguous if it were not for the preceding definitive singular sentence that disambiguates the plural construction. This discussion applies to *any* plural noun used as a subject. It is difficult to tell whether the predicate applies to the subject set as a whole or to each element of the subject set.

5.2.2 A, All, Any, Each One, Some, and The as Quantifiers

Consider the following correct sentence, relative to its intent, and the variations of it, given below:

(E85) An office has a door connecting the office to a hallway.

Example E85 is a statement about all offices. That is An is a universal quantifier much the same way Each would be in the same position. The example is saying that each office has at least one door whose purpose is to connect the office to some hallway. This sentence has several indirect articles, An or a, and one direct article, the. Occasionally people confuse the articles and switch a direct for an indirect article and an indirect for a direct article. Unfortunately, in doing so, the meaning of the sentence changes to something not intended.

If the An is changed to The,

(E86) The office has a door connecting the office to a hallway.,

The office refers to a previously introduced, possibly unique, office, and the sentence is no longer a universal statement about all offices. Ironically, making the whole sentence plural and using the direct article,

(E87) The offices have doors connecting the offices to hallways.,

makes the sentence once again a universal statement about all offices. However, we do not know how many hallways each door connects an office to. Changing a hallway in Example E85 to the hallway,

(E88) An office has a door connecting the office to the hallway.,

has the sentence meaning that all offices share the same previously introduced hallway. The sentence

(E89) An office has the door connecting the office to a hallway.

is incorrect since the door should refer to a previously introduced door. If each office has its own door, then it is unlikely that this door would have been introduced before, since the purpose of this sentence is to introduce the

door. Another possibility is that the door refers to a previously introduced single door that is shared by all offices. However, domain knowledge tells us that this possibility does not exist. In some people's minds, the door says that there is only one door per room, while a door allows more than one door per room. Changing the door in Example E89 to its door recovers correctness while making it clear that there is but one door per room:

(E90) An office has its door connecting the office to a hallway.

If the office in Example E86 is changed to an office,

(E91) An office has a door connecting an office to a hallway.,

the sentence ends up talking about potentially a second office, and that some how one office has a door connecting a possibly other office to a hallway.

When a direct or indirect article is used incorrectly for the intended meaning, and an error is introduced, then the reader wonders what is meant. When no error is introduced, but another meaning results, the reader may understand the wrong meaning. Alternatively, the reader may notice the writer's propensity for errors and be left wondering if the sentence means what is says.

The rest of this section describes how to correctly use the and a in English to convey precisely what is intended. The rules about definite and indefinite articles are maddenly different in languages other than English, much to the consternation of anyone having to write a requirements specification or legal contract in a language other than his native language. For example, in French, Portuguese, and other Latin languages, the definite article is used also to precede a general noun. The English translation of such a use is usually unadorned with any article. For example, the French

(E92) L'humanité avance.

is written in English as

(E93) Humanity advances.

In English, the indefinite article a often serves as an existential quantifier or a mathematical let there be for introducing an arbitrary element of some class. Also any or some can serve as an existential quantifier. In this respect, each or all serves as a universal quantifier. Unfortunately for the non-native speaker of English, a is sometimes used as a universal quantifier, as in the sentence

(E94) **An** office has a door connecting the office to a hallway.

in which the bold faced An is a universal quantifier, but each of the other as is an existential quantifier. In this case the An introduces a general office and serves as an Each.

In English, the definite article the, a personal pronoun, or a possessive pronoun can serve to designate a previously introduced specific element or a previously introduced arbitrary element from a class to which an existential quantifier has been applied. The can also serve to designate a specific element of a class or a specific component of an element. As mentioned, in languages other than English, the definite article can serve also to introduce a general noun. This use of the definite article does not occur in English.

Often, there is confusion between a and the, and there are times that a is incorrectly used repeatedly to refer to the same arbitrarily introduced element, when the should be used for all but the first. The effect is that the writer has introduced another element, in principle different from all others that were introduced before. Thus, each of a and the can give rise to referential ambiguity. Occasionally, the indefinite personal pronoun one is subjected to the same error when it is used as a subject in one sentence and continues to be used as the subject of subsequent sentences. Sometimes, all is used incorrectly instead of any.

In

(E95) A boy brings his dog. The boy feeds the dog.
$\exists b$ (b is a boy $\wedge \exists d$ (d is a dog $\wedge$ d belongs to $b \wedge b$ feeds d)),

the A boy introduces a new arbitrary boy, and the The boy talks about the same boy. The following sentences, with Some in place of A, He in place of The boy, and it in place of the dog, express the same thought.

(E96) Some boy brings his dog. The boy feeds the dog.

(E97) A boy brings his dog. He feeds the dog.

(E98) A boy brings his dog. He feeds it.

A pronoun refers always to a previously existentially introduced noun or specific noun. Strictly speaking, a pronoun should refer to the most recent sentential subject. This rule prevents ambiguity when there are more than one noun to which the pronoun might refer. However, when the gender and number of the pronoun are sufficient to uniquely identify a previously introduced noun, it seems to be acceptable to have a pronoun refer to a noun that is not the most recent sentential subject. In the last example above, it is clear that the He refers to the A boy subject of the previous sentence while the it refers to the his dog object of the previous sentence.

The pair of sentences

(E99) The boy brings his dog. The boy feeds the dog.

is not meaningful if they do not follow an earlier introduction of a boy, either by name or by a or some.

Replacing the his by a in

(E100) A boy brings his dog. The boy feeds the dog.
$\exists b$ (b is a boy $\wedge \exists d$ (d is a dog $\wedge$ d belongs to $b \wedge b$ feeds d))

gives

(E101) A boy brings a dog. The boy feeds the dog.,
$\exists b$ (b is a boy $\wedge \exists d$ (d is a dog $\wedge$ b feeds d)),

which eliminates the logical connection between the boy and the dog, to make the dog completely arbitrary.

Replacing the The boy by A boy in Example E100 gives

(E102) A boy brings his dog. A boy feeds the dog.
$\exists b$ (b is a boy $\wedge \exists d$ (d is a dog $\wedge$ d belongs to $b \wedge \exists b'$ (b' is a boy $\wedge$ b' feeds d))).

The second A boy introduces yet another arbitrary boy. This second boy could be the same as the first or entirely different. To make it clear that the second boy is not the first, one could say

(E103) A boy brings his dog. Another boy feeds the dog.
$\exists b$ (b is a boy $\wedge$ $\exists d$ (d is a dog $\wedge$ d belongs to b $\wedge$ $\exists b'$ (b' is a boy $\wedge$ $b \neq b'$ $\wedge$ b' feeds d)))

Combining the introduction of the second dog and the introduction of the second boy yields

(E104) A boy brings his dog. A boy feeds a dog.
$\exists b$ (b is a boy $\wedge$ $\exists d$ (d is a dog $\wedge$ d belongs to b $\wedge$ $\exists b'$ (b' is a boy $\wedge$ $\exists d'$ (d' is a dog $\wedge$ b' feeds d'))))

Again, it is possible to make it clear that the newly introduced boy and dog are not the same as the first.

(E105) A boy brings his dog. Another boy feeds another dog.
$\exists b$ (b is a boy $\wedge$ $\exists d$ (d is a dog $\wedge$ d belongs to b $\wedge$ $\exists b'$ (b' is a boy $\wedge$ $b \neq b'$ $\wedge$
$\exists d'$ (d' is a dog $\wedge$ $d \neq d'$ $\wedge$ b' feeds d'))))

One introduces an arbitrary person and as such serves as an existential quantifier. A subsequent reference to the same person should use a pronoun or some other reference to the one such as The person.

(E106) One brings his dog. He feeds his dog.
$\exists b$ (b is a person $\wedge$ $\exists d$ (d is a dog $\wedge$ d belongs to b $\wedge$ b feeds d))

(E107) One brings his dog. The person feeds his dog.
$\exists b$ (b is a person $\wedge$ $\exists d$ (d is a dog $\wedge$ d belongs to b $\wedge$ b feeds d))

Unless the intent is to introduce additional arbitrary persons, it is incorrect to refer to the first one with additional ones.

(E108) One brings one's dog. One feeds one's dog.
$\exists b$ (b is a person $\wedge$ $\exists d$ (d is a dog $\wedge$ $\exists b'$ (b' is a person $\wedge$ $\exists d'$ (d' is a dog $\wedge$ b' feeds d'))))

In this case, it is impossible to say that the dogs are owned by the bringers because the owners are yet newly introduced arbitrary persons.

A very common confusion is whether to use a or the in talking about an object possessed by an existentially introduced other object. Consider the sentences.

(E109) Joe brings the dog of a boy.
$\exists b$ (b is a boy $\wedge$ $\exists! d$ (d is a dog $\wedge$ d belongs to b $\wedge$ Joe brings d))

(E110) Joe brings a dog of a boy.
$\exists b$ (b is a boy $\wedge$ $\exists d$ (d is a dog $\wedge$ d belongs to b $\wedge$ Joe brings d))

The boy is existentially introduced as an arbitrary boy, which could even be Joe himself, assuming that Joe is a boy. In a sense the dog is also existentially introduced, however, we do know one thing about the dog; it belongs to the existentially introduced boy. Therefore it is not a completely arbitrary dog. The difference between the two sentences are in the number of dogs the existentially introduced boy is asserted to own. In the former case, with the, the boy is asserted to own *one* dog and that one dog is brought. In the latter case, with a, the boy is asserted to own one or more dogs and any one of them is brought.

A, any, and some are essentially equivalent as means to existentially introduce an arbitrary object. Each of

(E111) Joe brings a dog.

(E112) Joe brings any dog.

(E113) Joe brings some dog.

means

(E114) $\exists d$ (d is a dog $\wedge$ Joe brings d)

Sometimes an existential quantifier is not apparent in the wording. The sentence

(E115) Joe brings his dog.

does introduce an arbitrary dog only to use predicates to narrow down the field to the dog that Joe owns.

(E116) $\exists d$ (d is a dog $\wedge$ Joe owns d $\wedge$ Joe brings d)

Changing the existentially introduced dog to plural dogs forces introduction of an arbitrary set of dogs of which Joe brings all.

(E117) Joe brings some dogs.
$\exists D$ (D is a set of dogs $\wedge$ $\forall d$ ($d \in D \supset$ Joe brings d))

Changing some to his forces addition of more predicates qualifying the members of the arbitrary set of dogs.

(E118) Joe brings his dogs.
$\exists D$ (D is a set of dogs $\wedge$ $D = \{d \mid d$ is a dog $\wedge$ Joe owns $d\}$ $\wedge$ $\forall d'$ ($d' \in D \supset$ Joe brings d'))

Once Joe is bringing his dogs, adding that he brings all of them does not really change the meaning.

(E119) Joe brings all of his dogs.
$\exists D$ (D is a set of dogs $\wedge$ $D = \{d \mid d$ is a dog $\wedge$ Joe owns $d\}$ $\wedge$ $\forall d'$ ($d' \in D \supset$ Joe brings d'))

However, saying that he brings some of them does change the meaning to leave open the possibility that only a subset of the dogs that he owns are brought.

(E120) Joe brings some of his dogs.
$\exists D$ (D is a set of dogs $\wedge$ $D = \{d \mid d$ is a dog $\wedge$ Joe owns $d\}$ $\wedge$ $\exists D'$ ($D' \subset D \wedge$
$\forall d'$ ($d' \in D' \supset$ Joe brings d')))

Removal of his eliminates all limits on the set of dogs that are brought. In this case, all dogs and each dog appear to be the same, even though the grammar requires one to be plural and the other to be singular.

(E121) Joe brings all dogs.
$\forall d$ (d is a dog $\supset$ Joe brings d)

(E122) Joe brings each dog.
$\forall d$ (d is a dog $\supset$ Joe brings d)

The sentence

(E123) Some dog is white.

means

(E124) $\exists d$ (d is a dog $\wedge$ d is white)

Also, making dog plural works:

(E125) Some dogs are white.
$\exists D$ (D is a set of dogs $\wedge$ $\forall d$ ($d \in D \supset d$ is white))

Each of all and each corresponds to the use of a universal quantifier. They are equivalent in meaning even though one is plural and the other is singular.

(E126) All dogs are white.
$\forall d$ (d is a dog $\supset$ d is white)

(E127) Each dog is white.
$\forall d$ (d is a dog $\supset$ d is white)

When a or any is used with a sentence subject, the use is problematic. Sometimes, each can be a universal quantifier, i.e.,

(E128) A dog is white.

and

(E129) Any dog is white.

mean the same as

(E130) Each dog is white.

Sometimes, one is using a or any in the sense of one or some and each of Examples E128 and E129 can mean the same as

(E131) Some dog is white.

Even if one is using the natural language quantifier equivalents correctly, sometimes it is not clearly what is meant, and one must resort to formulae or to additional wording. Michael Jackson [45] has a particularly effective example of a lexical or scope ambiguity to illustrate this point. The sentence

(E132) Everyone likes a holiday.

has at least four meanings, each of which is given in both natural language and predicate calculus.

(E133) Everyone likes every holiday; i.e., everyone likes a holiday from work.
$\forall x, y$ ((person(x) $\wedge$ holiday(y)) $\supset$ likes(x,y))

(E134) Everyone likes the same holiday; e.g., everyone likes New Year's Eve.
$\exists y$ (holiday(y) $\wedge$ ($\forall x$ (person(x) $\supset$ likes(x,y))))

(E135) Everyone likes at least one holiday; i.e., Christine likes Easter, David likes Passover,
and Fred likes New Year's Eve and Christmas.

$\forall x$ (person(x) $\supset \exists y$ (holiday(y) $\wedge$ likes(x,y)))

(E136) Everyone likes one and only one holiday; Christine likes only Easter and no other, and David likes only Passover and no other.
$\forall x$ (person(x) $\supset \exists! y$ (holiday(y) $\wedge$ likes(x,y)))

The ambiguity arises from the multiple meanings of the indefinite article a. It can serve as an existential quantifier, as a universal quantifier, as an abbreviation for one, or as a combination of more than one.

At some hotel in North America, one of the authors saw a sign above the door to the hotel's swimming pool that says,

(E137) Any electrical appliance is not allowed in the pool area.

It is clear that the intention is to say,

(E138) No electrical appliance is allowed in the pool area.
$\forall a$ (a is an appliance $\supset$ a is NOT allowed in the pool area)

But what is the meaning of what the actual sign says? Since not allowed is equivalent to forbidden, it is possible to get rid of the negative to make a sentence that is easier to analyze. The sentence of Example E137 is equivalent to any of

(E139) Any electrical appliance is forbidden in the pool area.

(E140) An electrical appliance is forbidden in the pool area.

(E141) An electrical appliance is not allowed in the pool area.

(E142) Some electrical appliance is forbidden in the pool area.

(E143) Some electrical appliance is not allowed in the pool area.,

which in turn is equivalent to

(E144) $\exists a$ (a is an appliance $\wedge$ a is forbidden in the pool area)

(E145) $\exists a$ (a is an appliance $\wedge$ a is NOT allowed in the pool area).

Because of the existential quantification over the appliances, what the sign means is quite a bit different from what is intended.

(E146) No electrical appliance is allowed in the pool area.
$\forall a$ (a is an appliance $\supset$ a is NOT allowed in the pool area)

What the sign says is only that one cannot bring in some appliances. That is, so long as at least one appliance is left out of the pool area, it is acceptable to bring in some appliance or appliances. Since there are millions, if not billions, of appliances that are nowhere near the pool area—they are in our homes—the actual sign is effectively already true without having to exclude anything from the pool area. Observe the progression:

(E147) It is not the case that all appliances are allowed in the pool area.

(E148) Not (all appliances are allowed in the pool area).

(E149) **not** ($\forall a$ (a is an appliance $\supset$ a is allowed in the pool area))

(E150) $\exists a$ (a is an appliance $\wedge$ a is NOT allowed in the pool area)

The intended sign, Example E138, is equivalent to the following positive sentences each making use of all or each.

(E151) All electrical appliances are not allowed in the pool area.

(E152) All electrical appliances are forbidden in the pool area.

(E153) Each electrical appliance is not allowed in the pool area.

(E154) Each electrical appliance is forbidden in the pool area.

(E155) $\forall a$ (a is an appliance $\supset$ a is NOT allowed in the pool area)

If one wants to use any, a, or some, it has to be in the context of it is not allowed to ....

(E158) It is not allowed to bring any electrical appliance into the pool area.

(E156) It is not allowed to bring an electrical appliance into the pool area.

(E157) It is not allowed to bring some electrical appliance into the pool area.

5.2.3 Many and Few

Many and few suffer the same problem as does all. Each of them is plural. The following sentences are correct, and neither gives any clue as to how many dogs each bringer brings.

(E159) Many bring their dogs.
Viele bringen ihren Hund mit.

(E160) Few bring their dogs.
Wenige bringen ihren Hund mit.

Unfortunately, at least in English, there are no singular words corresponding to many and to few as each corresponds to all. Consequently, there are only convoluted ways to talk about each element of the set denoted by many and few.

(E161) Each of many brings his dog.
Jeder von vielen bringt seinen Hund mit.

(E162) Each of few brings his dog.
Jeder von wenigen bringt seinen Hund mit.

(E163) Many bring their dogs. Each of them brings his dog.
Viele bringen ihren Hund mit. Jeder von ihnen bringt seinen Hund mit.

(E164) Few bring their dogs. Each of them brings his dog.
Wenige bringen ihren Hund mit. Jeder von ihnen bringt seinen Hund mit.

(E165) There are many, each of whom brings his dog.

(E166) There are few, each of whom brings his dog.

One can use other words that mean one of many or one of few such as the typical, the average, the occasional or the rare, but these are not as clearly talking about one of many or one of few.

(E167) The typical boy brings his dog.

The rare boy brings his dog.

In English, when many is used as an adjective for a noun, e.g.,

(E168) Many boys bring their dogs.,

there is a singular construction that works, i.e.,

(E169) Many a boy brings his dog.,

and allows being precise about the number of object items per subject item. The same construction should work for few, but even the English speaking authors of this handbook have not heard that construction said.

The problem with many and few is like that of all. Each of them is plural and when it is used as the whole subject or as the main adjective of the subject, the subject ends up being plural and requiring a plural verb.

In essence, each of many *X* and few *X* is an existential quantifier, saying that there exists a set of *X*s each of which has the property ascribed to the many *X*s or few *X*s, respectively. Moreover, by the use of many or few, as opposed to some or at least one, the utterer is implying that the size of the set is a large or small fraction, respectively, of the size of the domain of all *X*s.

As with all, the habit in English is to treat the predicate or object in a sentence whose subject contains many or few as if it were applied to each element of the set denoted by the subject. However, grammatically, the predicate or object is supposed to apply to the whole set.

Michael Jackson has a concrete, real-life example of this problem. He points out that many baggage carousels in the U.S. have signs that read, Many bags look alike., and asks what it means. He suggests that while a computer scientist or mathematician might say that it means Many pairs of bags look alike., a practical lay person would suggest that it means Many other people's bags look like yours., or more correctly, Many bags belonging to other people look like yours.

By the way, many non-native speakers of English confuse many and much and confuse few and little. Here is a little trick to help remember the meanings. The first of each is digital, and the second of each is analog. The first of each is used with integers, and the second is used with real numbers. The distinction is like between dollar and money, and in fact one must say many dollars and few dollars but much money and little money.

5.3 Only, Also, and Others

Only, also, and other similar words have similar problems, those of proper placement within a sentence. The meaning of the containing sentence depends strongly on the placement of the problematic word.

5.3.1 Only

A very common mistake in English writing and speaking is the misplaced only. To be correct, an only should be immediately preceding the word or phrase that it limits. For example, if it is desired to say that the only thing that the boy brings is his dog, one properly says

(E170) The boy brings only his dog.
O garoto traz somente o seu cão.
Le garçon n'apporte que son chien. or Le garçon apporte seulement son chien.
Der Junge bringt nur seinen Hund mit.
Hayeled mavi rak et hakelev shelo.

At least among native English speakers, most will put the only before the verb no matter what it modifies, saying, instead,

(E171) The boy only brings his dog.
O garoto somente traz o seu cão.
Le garçon seulement apporte son chien.
Der Junge nur bringt seinen Hund mit.
Hayeled rak mavi et hakelev shelo.

The meaning of this alternative sentence is that the only thing the boy does to his dog is bring it. This quite sad, because then the boy does not also feed, love, bathe, etc. his dog; he only brings his dog. Most native English speakers understand this sentence as it is probably meant, because what it means does not make much sense. However, there are sentences of this form in which what it really means is as meaningful as what it probably means, and the careful reader is left wondering what the writer means.

Interestingly, the German sentence of Example E170:4 is ambiguous, and it means either the English sentence of Example E170:1 or the English sentence of Example E171:1. In spoken German, which is meant is clarified by emphasis. In fact, the German sentence of Example E171:4 is never used.

The famous Winston cigarette jingle is also incorrect in that its intention does not agree with its literal meaning, saying

(E172) I only smoke Winstons.
Eu somente fumo Winstons.
Je seulement fume des Winstons.
Ich nur rauche Winstons.
Ani rak m''ashen Winstonim.,

which means that the only thing I do with Winstons is smoke them. That is probably a good thing, because it excludes my eating them, using them as birthday candles, etc. Actually, this sentence is not very informative, because the only thing most people, who smoke, do with any cigarette is smoke them. The correct formulation of the jingle is

(E173) I smoke only Winstons.
Eu fumo somente Winstons.
Je ne fume que des Winstons. or Je fume seulement des Winstons.
Ich rauche nur Winstons.
Ani m''ashen rak Winstonim.,

which says that the only thing I smoke is Winstons. (Just in case the reader is worried, the authors smoke nothing.)

In most cases, people know what is meant by a sentence with a misplaced only, mainly because the actual meaning of the incorrect sentence is nonsense, for example, as the one cited as the meaning of the incorrect Winstons jingle. However, sometimes an incorrectly placed only, given the intended meaning, yields a perfectly reasonable interpretation, thereby leaving the reader wondering what word is really limited by the only. Below is a list of sentences found, each of which can be read either way, i.e., both the popularly understood meaning and the correct meaning are reasonable interpretations. For each, both meanings are listed underneath it.

(E174) It only illustrates the concepts.
It only illustrates the concepts and does not define them.
It illustrates only the concepts and not reasons for them.

(E175) Defiance of Russia only deepens in Chechnya.
Defiance of Russia only deepens in Chechnya and does not stay the same.
Defiance of Russia deepens only in Chechnya and not in Azerbaijan.

(E176) I only nap after lunch.
The only time I nap is after lunch.
The only thing I do after lunch is nap.

Peter Neumann in his short, elegant essay on only, "Only His Only Grammarian Can Only Say Only What Only he Only Means",[9] [62] gives 15 variations of the sentence,

(E177) I said he thought secret users may write secret data.

by migrating only through different places in the sentence. Each sentence potentially has a different meaning.

(E178)
1. Only(,) I said he thought secret users may write secret data.
2. Only I said he thought secret users may write secret data.
3. I only said he thought secret users may write secret data.
4. I said only (that) he thought secret users may write secret data.
5. I said only he thought secret users may write secret data.
6. I said he only thought secret users may write secret data.
7. I said he thought only (that) secret users may write secret data.
8. I said he thought only secret users may write secret data.
9. I said he thought only secret users may write secret data.
10. I said he thought secret users only may write secret data.
11. I said he thought secret users only may write secret data.
12. I said he thought secret users may only write secret data.
13. I said he thought secret users may write only secret data.
14. I said he thought secret users may write secret data only.
15. I said he thought secret users may write secret data only.

Actually, Neumann gave six additional variations, distinguished from the original 15 by different emphases when speaking the sentences. Thus, here is a case of a sentence about secret data for which a misplaced only can mislead the reader. If any of the 15 sentences were in a specification, there is no telling what the reader might deduce. Here is a case in which the writer and the reader must be equally precise about meanings, and too few people are. The proof that most people are not that precise comes from the fact that most people understand the incorrect Winstons example in the incorrect, but intended way.

5.3.2 Also

Also suffers the same fate as only in that it is supposed to be put immediately preceding the word or phrase it modifies, while most people put it in one of two standard places regardless of what is also.

(E179) The boy brings also his dog.
O garoto traz também o seu cão.
Le garçon apporte aussi son chien.
Der Junge bringt auch seinen Hund mit.
Hayeled mavi gam et hakelev shelo.

conveys the meaning that in addition to the other things the boy brings, he brings his dog. Most people write either

[9] In our opinion, the title lacks one only, between Grammarian and Can.

(E180) Also the boy brings his dog.
Também o garoto traz o seu cão.
Aussi le garçon apporte son chien.
Auch der Junge bringt seinen Hund mit.
Gam hayeled mavi et hakelev shelo.

or

(E181) The boy also brings his dog.
O garoto também traz o seu cão.
Le garçon aussi apporte son chien.
Der Junge bringt auch seinen Hund mit.
Hayeled gam mavi et hakelev shelo.

intending the meaning of the sentence above. The real meaning of the first alternative is that in addition all the other people who bring their dogs, the boy brings his, and that of the second alternative is that in addition to all the other things the boy does, possibly to his dog, he brings his dog.

Here again, the German sentence of Example E179:4 is ambiguous, and it means either the English sentence of Example E179:1 or the English sentence of Example E180:1. Again, in spoken German, which is meant is clarified by emphasis, and the German sentence of Example E180:4 is never used.

5.3.3 Even

There are other words that have the same problem as only and also. These include almost, even, hardly, just, merely, nearly, and really. For example, consider the following sentences involving even. After each is given a clarification that makes the meaning clear.

(E182) Even I did not see him on Monday. No one, including I, saw him on Monday, and one would certainly expect that I would see him.

(E183) I did not even see him on Monday. I had no contact with him at all on Monday.

(E184) I did not see even him on Monday. I saw no one on Monday.

(E185) I did not see him even on Monday. I saw him on no day this week, even on Monday of all days.

5.4 Structural Ambiguity

Each ambiguity described in this section arises from uncertainty as to which nearby word or phrase a given word modifies. The given word may be an adjective, a pronoun, the word this, the word otherwise, the word not, either by itself or with because, or one of several conjunctions in a sentence.

5.4.1 Adjectives and Other Modifiers

In English, the phrase

(E186) English grammar teacher
(no direct equivalent in Portuguese)
(no direct equivalent in French)
(no direct equivalent in German)
moreh l'diqduq angli

is just one example of a common ambiguity involving nouns used as adjectives. The phrase can be read two ways:

1. English, i.e., British, teacher of grammar, not necessarily English grammar
2. teacher of English grammar, with the teacher's nationality left unspecified

This ambiguity is avoided in most other languages, because in them, nouns cannot be used directly as adjectives; they must undergo some changes to become full-fledged adjectives. Thus, in most of the other languages, the ambiguous phrase cannot be translated to an ambiguous phrase. Hebrew is an exception even though nouns cannot be used as adjectives. In certain situations in Hebrew, it is not clear which noun is modified by an adjective. If the teacher were a woman then one would indicate which word is modified by the adjective by chosing the correct gender ending.

(E187) morah l'diqduq angli

(E188) morah l'diqduq anglit

In the first case, the masculine diqduq (grammar) is modified by the masculine angli (English). In the second case, the feminine morah (teacher) is modified by the feminine anglit (English).

The only way to say this phrase in most of the other languages corresponds to the disambiguated forms in English:

(E189) English teacher of grammar
professor inglés da grammatica
professeur anglais de la grammaire
englischer Lehrer der Grammatik or englischer Grammatiklehrer
moreh angli l'diqduq

(E190) teacher of English grammar
professor da grammatica inglesa
professeur de la grammaire anglaise
Lehrer der englischen Grammatik or Lehrer der Grammatik von Englisch
moreh l'diqduq ba'anglit

In German, nouns are capitalized and other words are not, and nouns can be strung together to make a compound noun.

5.4.2 Pronoun References

Recall the following sentence, used as an example for clarifying problems with definite and indefinite articles.

(E191) An office has a door connecting the office to a hallway.

Many times, one will write instead

(E192) An office has a door connecting it to a hallway.

The question that needs to be asked about the it is "What is it?". The closest previous noun is door, but the subject of the sentence is office. It could be either the office or the door. In this case, domain knowledge indicates strongly that it is the office, for why bother telling the reader that the door, which is for connecting things to a hallway, connects itself to a hallway? However, it is not hard to invent a sentence in which domain knowledge does not help much to disambiguate the reference of a pronoun.

(E193) Bob said to Joe that he must leave.

Who is he that must leave, Bob, Joe, or someone else? The simplest way to avoid confusion is to replace the pronoun with an unclear reference by the noun that the sayer intends to be the pronoun's referent. Thus, Example E192 should be replaced by Example E191 and Example E193 should be replaced by one of:

(E194) Bob said to Joe that Bob must leave.

(E195) Bob said to Joe that Joe must leave.

(E196) Bob said to Joe that Dan must leave.

as the meaning may be.

5.4.3 This and Whole Ideas

An extremely popular usage error, in the sense of the number of occurrences, that causes ambiguity is the this that refers to a whole idea rather than to a specific noun, as it should. A stand-alone this can serve as a pronoun and, as with other pronouns such as it, he, and she, it should refer to a previously mentioned noun. However, many people use this as the first word of a sentence referring not to a previously mentioned noun, but to a whole idea captured by the previous sentences.

This handbook was carefully crafted so that except for inside quotations by other authors, there are no examples of this phenomenon.[10] The one example, in Section 3.2, is in a quoted definition of an ambiguous requirement by Harwell, Aslaksen, Hooks, Mengot, and Ptack [39]

(E197) A requirement must be unambiguous in the sense that different users (with similar backgrounds) would give the same interpretation to the requirement. **This** has two aspects. On the one hand there is the aspect of grammatical ambiguousness .... On the other hand there is the aspect of ambiguousness arising from a lack of detail allowing different interpretations.

What is the bold-faced This? By grammatical rules, it could be the subject of the previous sentence, A requirement. However, a requirement does not have the two mentioned aspects. So what can this be? This refers to the entire concept that different users, with similar backgrounds, would give the same interpretation to the requirement. Therefore, a better wording of the sentence This has two aspects. would be

(E198) This concept of multiple interpretation by different readers has two aspects.

The sentence must be about multiple interpretation, and not the lack of multiple interpretation, since the two aspects are about ambiguousness, and not lack of ambiguousness. However, then it is not even clear that there should be a This, because the previous sentence talked the *lack* of multiple interpretation. Therefore, a still better replacement for This has two aspects. is

(E199) Multiple interpretation by different readers has two aspects.

with no this.

We now look at some examples of text from this handbook to see what would happen if we had not followed our own rules about avoiding this that refers to whole concepts. In previous incarnation of Section 3.3.4, we said

(E200) In traditional semantics, the relation between a word or phrase and the object of the real world that the word or phrase describes is called a *reference*. An *anaphor* is an element of a sentence that depends for its reference on the reference of another element, possibly of a different

[10] Did you expect that we would have any? Nu?!? Of course not! One of the problems with writing a handbook like this one is that it is so easy to get hoisted by our own petard! You can bet your life that we tried our damndest to follow our own rules.

sentence. This **other element** is called the *antecedent* and must appear earlier in the same sentence or in a sentence before.

If we had left out the bold-faced text, it would be very difficult to determine which noun in the previous sentences this refers to. The possibilities are

1. the element that is the anaphor,
2. the first reference,
3. the second reference, and
4. the other element,

It would take a bit of figuring, including grammatical domain knowledge, to determine that it had to be the fourth possibility.

The longest example of a this that would refer to a whole concept is from a previous incarnation of Section 2.4

(E201) An indication of the difference in impact of changes to software and contracts is the special clause that is very common in contracts that specifies that if an arbitrary clause in the contract is invalidated, the rest of the contract still holds. The though of applying such an idea to software makes any programmer chuckle. If one part of a program fails, generally the whole program comes to a screeching halt, because each part of a program is connected to all other parts. In law, the hope is that this clause is workable, that the parties can find meaningful ways to apply the still valid clauses. However, in some cases, such a special clause carries the seeds of its own contradiction, as it is sometimes impossible to decompose the contract into an invalid part and a totally independent valid part. Often there are logical implications from one clause to another to the extent that the whole contract is invalidated by a single invalidated clause. This **removal problem** is not unlike the problem of throwing out or modifying an incorrect line of code without affecting the rest of the program.

Without the bold-faced removal problem that makes it clear that we are talking about the problem caused by removing a whole clause from contract, which is described in the entire paragraph up to the point of the This, the reader would be left wondering what This is.

In a previous incarnation of Section 4.1, we defined

(E202) A controlled language is a precisely defined subset of natural language for the use in specific environments. The objective of a controlled language is to increase the readability and understandability of any kind of technical documentation. This **improvement** is accomplished by reducing the inherent ambiguity of natural language through a restricted grammar and a fixed vocabulary.

Without the bold-faced improvement, it is not as clear what is accomplished by reducing the inherent ambiguity of natural language.

Finally, we see a case in which avoiding a this that refers to a whole concept, we avoid another ambiguity, that of an ambiguous anaphor. Consider the example given in Section 3.3.1 of a behavior-vs.-disposition ambiguity:

(E203) This is a fast car.

We observed that the sentence can denote the current behavior of a particular car or the general capability of each element of a class of cars, such as Ferrari, independent of the current behavior of any particular Ferrari. If the sentence had been written properly, the noun phrase following the This would disambiguate:

(E204) This car, in front of us, is a fast car.

(E205) This brand of car is a fast car.

5.4.4 Otherwise

Otherwise is a tricky animal. The purpose of otherwise is to specify what is or is to happen when a previously mentioned condition is not true, after what is or what happens when the condition is true has been specified. We call what is or is to happen when a condition is true the *normal response* to the condition and what is or what is to happen when the condition is false, i.e., otherwise, the *alternative response*. If there are more than one condition described prior to the otherwise, it is very easy to use otherwise in way that leaves the reader wondering to which condition the otherwise applies, that is, to which condition the alternative response is being given. In other words, the scope of the otherwise is ambiguous.

Consider the sentences stripped down to bare essentials:

(E206) If C1, A1. If C2, A2. If C3, A3. Otherwise, AO.

An occasional writer, after a series of conditions, uses otherwise in the sense of if all else fails, as providing an alternative response to the last condition under the assumption that each condition after the first is tested only if the previous condition is false. Another uses otherwise as providing the alternative response to the first condition in the series. Still another uses otherwise as providing the alternative response to some condition in the series, the one of which he just happens to be thinking. Of course, the poor reader has a hard time divining what the writer was thinking.

Interestingly, programming language designers faced the same problem when designing the conditional statement, **if ... then ... else ...**. In earlier languages, e.g., Algol 60 [61], the **else** was not required and there was no explicit end, e.g., **fi**, to the whole conditional, as **if ... then ... else ... fi** in Algol 68 [78]. Consequently, when one wrote in Algol 60

(E207) If C1 then if C2 then S2 else S3

the compiler and the reader did not know to which **if** the **else** belonged. In Algol 68, one has to use **fi** to close off any conditional, and therefore he would have to write one of

(E208) If C1 then if C2 then S2 fi else S3 fi

or

(E209) If C1 then if C2 then S2 else S3 fi fi

thus making perfectly clear to which **if** the **else** belongs.

While the problem of the association of **else**s with conditionals has been solved for programming languages by adding an additional mandatory keyword to mark the end of a conditional, natural language does not have that option. The problem is compounded by the fact that the clause beginning with otherwise and the alternative response is supposed to be in a sentence separate from that containing the if, the condition, and the normal response.

The solutions in the natural language case involve using external structural elements or parenthetical material to make it clear to which if an otherwise belongs.

Thus one would rewrite Example E206 as one of:

(E210) If C1, A1.
If C2, A2.
If C3, A3.
Otherwise, AO.

or

(E211) If C1, A1.
If C2, A2.
If C3, A3.
Otherwise, AO.

By the indentation structure, both capture the idea that testing C2 is the alternative response to condition C1 and that testing C3 is the alternative response to condition C2, which is really C2 $\wedge \neg$ C1. Example E210 captures the if-all-else-fails meaning, that AO is the alternative response to condition C3. Example E211 captures the idea that AO is the alternative response to condition C1.

This sort of structuring can be achieved also in normal linear text by using outline style numbering. However, nothing beats a two-dimensional layout for making these associations clear, especially to those with programming experience.

Another way to achieve proper association of otherwise is to state explicitly the condition under which the otherwise, alternative response is done. For any otherwise, the condition under which its alternative response is done is the logical negation of the condition to which it is associated. Thus one would rewrite Example E206 as one of:

(E212) If C1, A1. If C2, A2. If C3, A3. Otherwise, when $\neg$C1 and $\neg$C2 and $\neg$C3, AO.

or

(E213) If C1, A1. If C2, A2. If C3, A3. Otherwise, when $\neg$C1, AO.

Example E212 captures the if-all-else-fails meaning, that AO is the alternative response to condition C3, whose testing is the alternative response to condition $\neg$C1 and $\neg$C2. Example E213 captures the idea that AO is the alternative response to condition C1.

There are undoubtedly other ways by which the author can make his intent known.

5.4.5 Not

Not has a placement problem different from that of only. Except for a major exception, not negates the word that follows it. The exception is that negating the verb of a sentence normally negates the whole sentence. Thus,

(E214) He does not bring his dog.

means

(E215) It is not the case that he brings his dog.,

which should be parsed as

(E216) It is not the case that (he brings his dog).

Negating

(E217) He brings his dog.

means that something is happening that causes him not to be bringing his dog, for example,

(E218) He does not bring his dog; he brings his *cat*.

(E219) He does not bring his dog; he *feeds* his dog.

(E220) He does not bring his dog; *she* brings his dog.

From a simple negation, all one knows is that the negated sentence is not true. Additional information needs to be given to explain in how that sentences is not true.

If there are multiple verbs in a sentence, care must be taken as to the placement of the nots, up to one for each verb. There are three negations of the sentence

(E221) It is clear that he brings his dog.

They are

(E222) It is clear that he does not bring his dog.

(E223) It is not clear that he brings his dog.

(E224) It is not clear that he does not bring his dog.

These three sentences with additional clarifying information are

(E225) It is clear that he does not bring his dog. It is clear that something is happening that causes his not to be bringing his dog.

(E226) It is not clear that he brings his dog. It is only possible that he brings his dog.

(E227) It is not clear that he does not bring his dog. It is only possible that he does not bring his dog.

The two latter sentences say almost the same thing since it is uncertain as to what is actually happening. The difference is in the stated belief or hope of the utterer.

There are at least five ways to negate the command

(E228) Try to bring your dog

They together with clarifying additional information are

(E229) Try not to bring your dog. Try to bring your cat or try to leave your dog at home.

(E230) Try to not bring your dog. Try to feed your dog.

(E231) Do not try to bring your dog. Don't even attempt to bring your dog or force yourself to bring your dog.

(E232) Do not try not to bring your dog. Don't even attempt to bring your cat or to leave your dog home.

(E233) Do not try to not bring your dog. Don't even attempt to feed your dog.

5.4.6 Not and Because

A special case of scope ambiguity arises when because is used following a clause containing a negation such as not.

(E234) The witness said that the case was not brought before committee because of the incident the night before.

One way to read it is that the case was not brought before committee and the incident the night before is what caused the case not to be brought. However, is it definite that the case was not brought before committee at all? One cannot be sure. Another way to read the sentence is that the incident the night before did not cause the case not to be brought, but in fact, the case was brought before committee anyway.

To disambiguate, the sentence must be restructured to make the scope of the not clear.

(E235) The witness said that because of the incident the night before, the case was not brought before committee,

(E236) The witness said that the incident the night before did not prevent the case from being brought before committee.

This sort of sentence is often subconsciously disambiguated by both the sender and the receiver. That is, from either meaning the sender constructs the ambiguous sentence, and he says it, unaware of the other meaning. The receiver understands it one way, which possibly disagrees with the intent of the sender.

5.4.7 And and Or in the Same Sentence

In mathematics, there are precedence rules that govern the meaning in a sentence with more than one logical operator, and and or. The binary and binds tighter than the binary or, and associativity is to the left, and if a different precedence or associativity is desired, parentheses are used to make the desire clear. Natural language has no such precedence and associativity rules, and it is not considered good form to use parentheses in natural language text to indicate precedence and associativity. In the case of a sentence with only one kind of binary logical operator, there is no problem as in natural language associativity appears to be towards the beginning of the sentence.[11] As a consequence, one finds many ambiguous sentences with at least one each of and and or. In many restaurants, the menu says something similar to

(E237) With each entree, you get a vegetable and salad or soup.

The customer wonders which of the following is meant.

(E238) With each entree, you get (a vegetable and (salad or soup)).

(E239) With each entree, you get ((a vegetable and salad) or soup).

The customer guessed that the intended meaning is the former, because where the restaurant is, in North America, salad and soup are both eaten before the main course and the vegetable is eaten with the main course. It makes sense to have a choice for the before dish and to have that choice *and* the side dish. However, this reasoning is culture relative, and there are places in which this reasoning cannot be used to resolve this sentence. In China, soup comes

[11] In a language written from left to right, associativity towards the beginning of the sentence amounts to left associativity. We had to say, "associativity towards the beginning of the sentence" because not all natural languages are written from left to right!

after the main course, and there is no salad at all. In Europe, salad comes after the main course, while soup comes before the main course. Nevertheless, the guessed meaning was confirmed by the waitress when the customer simply asked for a vegetable and a soup and a salad and the waitress said that he has to choose between the soup and the salad. An earlier attempt to get the waitress's help disambiguating failed in an interesting way. The customer asked the waitress to explain the choice, and she said, you get a vegetable *and* salad or soup. Her answer was confusing because contrary to the norm in mathematics, she emphasized not the tighter binding operator, but the more global operator.

It is interesting also that the sentence was written in the order it was. Based on the reasoning used to guess the intended meaning, the sentence should probably have been written as

(E240) With each entree, you get salad or soup and a vegetable.

with the intended parsing

(E241) With each entree, you get ((salad or soup) and a vegetable).

Perhaps the author felt that the *first* logical operator in the sentence should be the more global and was using position in the sentence to disambiguate.

Clearly, in an SRS or a contract, one should not depend on such reasoning to disambiguate a sentence. The usual approach to make the writer's intent clear is to use physical structure or enumeration to indicate precedence. The menu example could be written as

(E242) With each entree, you get
a vegetable
and
salad or soup.

or

(E243) With each entree, you get
a vegetable and salad
or
soup.

or as one of

(E244) With each entree, you get 1) a vegetable and 2) salad or soup.

(E245) With each entree, you get 1) a vegetable and 2) a) salad or b) soup.

(E246) With each entree, you get 1) a vegetable and salad or 2) soup.

(E247) With each entree, you get 1) a) a vegetable and b) salad or 2) soup.

5.5 Parallelism

Many people do not observe parallelism when it is required. For example, in phrases connected by a conjunction, each of the phrases should be of the same part of speech, e.g, all nouns, all verbs, etc. More than that, within the part of speech, each should be of the same grammatical structure, e.g., when they are all nouns, if one is a gerund, then all the others should also be gerunds; when they are all verbs, if they should all be in the same tense. The same should hold for each item in a bulleted or enumerated list.

5.5.1 Assumed Parallelism

Michael Jackson reports a sign above an escalator [45].

(E248) Shoes must be worn. Dogs must be carried.

The structural parallelism implies a semantic parallelism. That is, the two sentences are structurally identical, of the form

(E249) *X* must be *Y*.

Therefore, the reader might be compelled to believe that they are semantically identical with respect to the must be. However, they are quite different in terms of what the user of the escalator is compelled to do. Expressing the two sentences in conditional form yields

(E250) If you want to use the escalator, you must be wearing shoes.

(E251) If you want to use the escalator, and you have a dog with you, then you must carry the dog.

The semantic parallelism implied by the structural parallelism would suggest that to use the escalator either,

1. you must wear shoes only if you happen to have them with you, or
2. you must have a dog with you and you must carry it, and if you have no dog with you, you cannot use the escalator.

Our culture helps to disambiguate the sentences and recognize the correct meanings. That same culture helps us realize that shoes are not absolutely necessary, that any reasonably protective footwear such as sandals are sufficient. The same culture makes us realize also that if one has a cat with him, he must carry it too.

Michael Godfrey demonstrates the importance of culture to understand this sign by replacing the *X* and *Y* by three nonsense words and one rarely used word.

(E252) Slumpets must be slarpled. Frumplings must be defenestrated.

There is no way to see that semantic parallelism is not intended by the structural parallelism.

5.5.2 Than and Different From and Parallelism

When a comparison is done using than, it is essential that the two sides of the than be parallel in order to be able to determine unambiguously what are being compared. Consider yet another sentence that is often subconsciously disambiguated on both sides:

(E253) Cleveland is closer to Philadelphia than New York.

There are two readings of this sentence. Assuming that Cleveland is the one in Ohio, Philadelphia is the one in Pennsylvania, and New York is the one in New York State, under one reading, the sentence is true, and under the

other, it is false.

(E254) Yes, the distance between Cleveland and Philadelphia is smaller than the distance between Cleveland and New York.

(E255) No, the distance between Cleveland and Philadelphia is not smaller than the distance between New York and Philadelphia.

The problem with the original sentence is that the reader does not know the grammatical position of New York in the elided sentence that comes after the than. The elided sentence that comes after the than is of the form:

(E256) *X* is close to *Y*.

The writer, to make his intent clear, must give enough of the elided sentence to make it clear whether New York is lX or Y. Thus, the following sentences are disambiguations of the original sentence.

(E257) Cleveland is closer to Philadelphia than to New York.

(E258) Cleveland is closer to Philadelphia than Cleveland is close to New York.

(E259) Cleveland is closer to Philadelphia than New York is.

(E260) Cleveland is closer to Philadelphia than New York is close to Philadelphia.

The presence of to preceding New York in Example E257 forces the realization that New York is parallel to Philadelphia, and the presence of is following New York in Example E259 forces the realization that New York is parallel to Cleveland. The full form of each elided form probably sounds strange, and thus, people prefer to say the elided form.

Notice how parallelism is observed in the full forms; in each of these sentences, full clauses are on both sides of the than, and the parallelism is very pronounced.

Another trouble spot is different from. First of all, many people incorrectly use than as the preposition following different. The correct preposition is from. Here too, failure to observe parallelism makes it difficult to tell what are different and how they are different. Consider the deceptively innocent sentence:

(E261) The method used for coding is different from structuring.

Is the method for structuring different? Is structuring the method that is different? As with the than, more of the elided clause around structuring is needed to be able to tell what is different from what. If the method for structuring is different, the correct sentence is any of the following:

(E262) The method used for coding is different from that for structuring.

(E263) The method used for coding is different from that used for structuring.

(E264) The method used for coding is different from the method used for structuring.

In the first of these, that stands for the method used, and in the second of these, that stands for only the method. The for preceding the structuring is sufficient to force the realization that structuring is parallel to coding and not to method.

If on the other hand, structuring is the method that is different, the correct sentence is:

(E265) The method used for coding is different from structuring for coding.

The for coding following structuring forces the realization that structuring is parallel to method and not to coding.

5.6 Quoted Word vs. Denotation

Another source of ambiguity is confusion between a word used to mean itself and the same word used to mean its denotation. Most of the time, a word is used to mean its denotation. Certainly, every word in this current paragraph is used to mean its denotation. However, when it is necessary to talk about words *per se*, it is necessary to use words to mean themselves. To distinguish the two uses of a word, the use of a word to mean itself is normally quoted. This handbook also typesets such words in a sans-serif font, just to save from having to use quotation marks all over the place.

Thus, strictly speaking the sentence,

(E266) He said no.

is incorrect, because the word no is being used to mean the word "no" rather than the denotation of the word no. The sentence should be written,[12]

(E267) He said, "no".

In this particular case, the incorrect sentence is understood by all. What it really means is nonsense; so, there is no ambiguity.

However, suppose you have an *X* test in statistics that is supposed to be run on the data of an experiment to tell you whether or not conclusions drawn from the data are significant. This test, given the input data, reports one of two results, ''significant'' and ''not significant'', with the meanings that the conclusions drawn on the data are significant and not significant, respectively. Statisticians often write,

(E268) The *X* test result was not significant.

or

(E269) The *X* test result was significant.

These sentences are punctuated incorrectly if the intent is to report the result of the *X* test. They should be written

(E270) The *X* test result was "not significant".

or

(E271) The *X* test result was "significant".

[12]There are some who insist that the sentence ending period should be inside the quotation mark and the sentence should be written,

He said, "no."

We prefer what we wrote as more logical. In any case, this point is not the key issue at hand.

In this case, the incorrect sentences have sensical meanings. The first says that the *X* test result itself is not significant and the second says that the *X* test result itself is significant. To see the distinction, see that the result of the *X* test is basically a boolean value, "0" or "1". Therefore, the correct sentences could be rewritten as,

(E272) The *X* test result was "0".

and

(E273) The *X* test result was "1".

It would still be correct to say

(E274) The *X* test result was not significant.

or

(E275) The *X* test result was significant.

to discuss the *X* test itself. That is, it would be legitimate to say

(E276) The *X* test result of "0" was significant.

or

(E277) The *X* test result of "not significant" was significant.

In these last examples, the use of of is arbitrary. The last example could have been written in at least three other ways, with an apposition, with a compound predicate, and with a compound sentence:

(E278) The *X* test result, "not significant", was significant.

The *X* test result was "not significant" and significant.

The *X* test result was "not significant", and it was significant.

These sentences need to be seen, not heard, to be understood.

There is an apocryphal urban legend of a customer that calls Microsoft service to complain that he cannot find the "any" key. The system has told the user:

(E279) Press any key.,

but the customer's keyboard has no "any" key. Here again is an ambiguity created by failure to understand the role of quotation marks and what the lack of quotation marks means. Indeed, here is one case in which the customer was wrong. For the customer to have rightly been searching for the "any" key, the message from the system had to have been

(E280) Press the "any" key.

Singh and Singh [73] point out a more serious problem, a variant of the dangerous "all" problem [9], with the sentence of Example E279 given by the system. Specifically, it is not true. Pressing the "Caps Lock", the "Control", or the "Shift" key, among others, will not have the desired effect.

Think how confusing this handbook would be if words used as words were not distinguished by quotes or by being set in a sans-serif font. The authors admit that sometimes it was difficult to decide if a word was being used to mean itself or to mean its denotation.

5.7 Time Expressions

Specifications of times, both instants and durations, are particularly troublesome and involve issues of open and closed intervals and prepositions that do not translate well between languages. Among these prepositions, two in English, by and until, are very close but not identical in meaning, and non-native speakers of English have a habit of confusing them or using only one to mean either. More than once, we have been ordered by a non-native speaker of English:

(E281) Send the final copy of the paper until 1 September.

when we should have been ordered:

(E282) Send the final copy of the paper by 1 September

The sentence E281 with until instead of by is actually meaningless. It would have been meaningful, but with the opposite meaning, to have been told:

(E283) You do not have to send the final copy of the paper until 1 September.

E283 says that the earliest required date for sending the final copy of the paper is 1 September. While the paper could be sent earlier, it is not really needed until 1 September, and it could be sent on, for example, 1 October, after the deadline of E281. It is meaningful also to say:

(E284) You do not have to send the final copy of the paper by 1 September.

The meaning of E284 is close to that of E283 but is weaker, because E284 does not require that the final copy ever to be sent, while E283 insists, while not being very definitive about any deadline, that the final copy be sent eventually.

In all languages, the time prepositions are tricky. Each has a very particular meaning and denotes a particular duration or instant of time. Each preposition precedes a particular time duration or instant. The meaning of such a prepositional phrase depends on whether the object of the preposition is a duration or an instant of time. In particular, on is used exclusively with durations, at is used exclusively with instants, and they correspond to each other. The other prepositions are used with both durations and instants, and the meanings in both cases correspond. Additional variations arise by whether the duration of time denoted by the phrase is bounded above or below, and by whether the bound is open or closed.

This section clarifies the common time prepositions by showing the time durations or instants denoted by each. Its subsections consider variations of two commands,

(E285) Eat *when*.

and

(E286) Don't eat *when*. ,

with different time prepositions substituted for *when*. In each case, if the command is meaningful, a diagram is given, showing the duration or instant the eating should or may *be done*, even if the sentence is a command not to

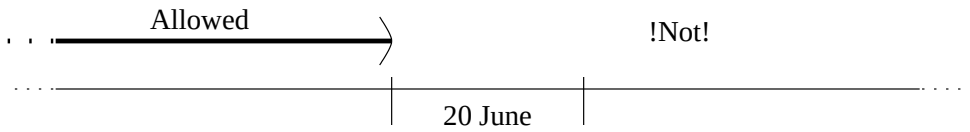
eat. In the sentences of the first subsection, the indirect object of the preposition is a time duration. In the sentences of the second subsection, the indirect object of the preposition is a time instant. In each diagram, "——)" represents an open interval of time, but "——]" represents a closed interval of time. Over the duration or instant in the diagram is the word "Required" or "Allowed" to indicate whether the eating is required or allowed in the duration or at the instant. Over the portion of time not included in the duration or instant the eating should or may be done, "!Not!" denotes that eating is not allowed, and "??" denotes that it is unspecified as to whether eating is allowed.

5.7.1 Durations

(E287) Eat until 20 June.
Eat up to 20 June.

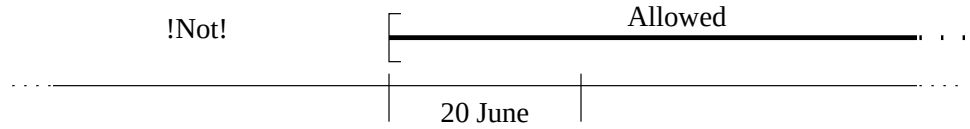
This command with until or up to has a meaning only because eating is an activity that can be done continually. If the command were something that can or should be done only once, then the command would not have meaning, as with Example E281. On and after the start of 20 June, it is unspecified as to whether eating is allowed.

(E288) Eat only until 20 June.
Eat only up to 20 June.



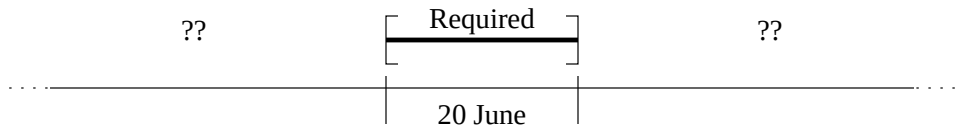
The addition of only to each command of Example E287 changes the command from a requirement to eat to a permission to eat, but in the same duration. Moreover, at and after the start of 20 June, eating is definitely prohibited.

(E289) Don't eat until 20 June.
Don't eat up to 20 June.
Don't eat only until 20 June.
Don't eat only up to 20 June.



It is required not to eat until the start of 20 June, and at and after that instant, eating is allowed. In each of these cases, adding only does not change the meaning.

(E290) Eat on 20 June.
Eat during 20 June.
Eat *duration* and including 20 June.

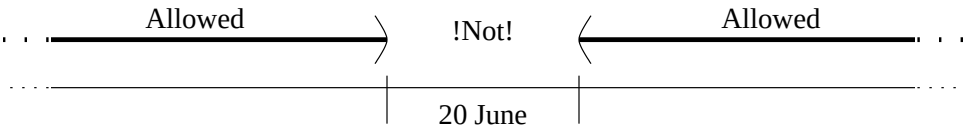


In the case of *duration* and including, the shown duration is united with that implied by the *duration*.

(E291) Eat only on 20 June.
Eat only during 20 June.
Eat only *duration* and including 20 June.

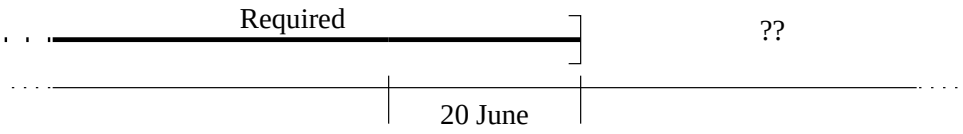
In the case of *duration* and including, the shown duration is united with that implied by the *duration*. The addition of the only to the commands of Example E290 changes each command from a requirement to eat during 20 June to a permission to eat during 20 June and makes it that eating is prohibited in times other than during 20 June.

(E292) Don't eat on 20 June.
Don't eat during 20 June.
Don't eat *duration* and including 20 June.
Don't eat only on 20 June.
Don't eat only during 20 June.
Don't eat only *duration* and including 20 June.

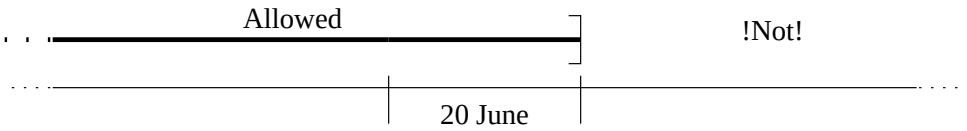


In the case of *duration* and including, the shown duration is intersected with that implied by the *duration*. The addition of only does not change the meaning of these negative commands.

(E293) Eat through 20 June.

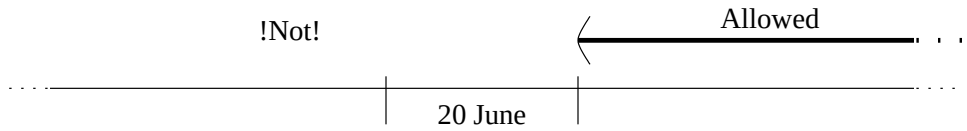


(E294) Eat only through 20 June.



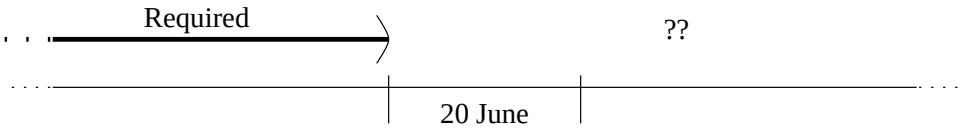
The addition of only to the command of example E293 forces prohibition of eating after the end of 20 June.

(E295) Don't eat through 20 June.
Don't eat only through 20 June.



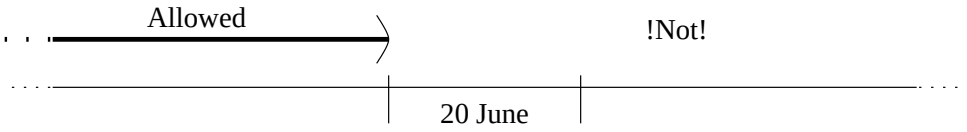
The addition of only does not change the meaning of the command.

(E296) Eat by 20 June.



The difference between Example E296 and Example E287 is that the former could mean that the eating is continual in the duration while the latter could mean that the eating happens once in the duration.

(E297) Eat only by 20 June.

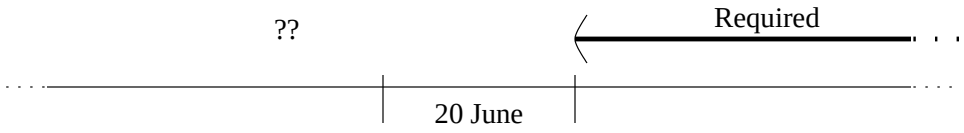


The addition of only to the command of Example E296 changes the command to eat to only allowing eating and forcing a prohibition against eating on or after the beginning of 20 June.

(E298) Don't eat by 20 June..
Don't eat only by 20 June.

For Example E298, there is no diagram because its commands have no real meaning.

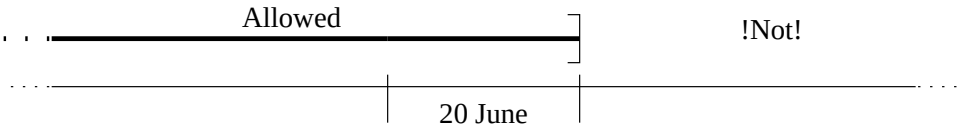
(E299) Eat after 20 June.



(E300) Eat only after 20 June.

The addition of only to the command of Example E299 changes the command to eat to only allowing eating and forces a prohibition against eating before or at the end of 20 June.

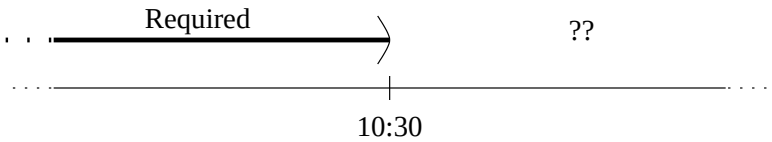
(E301) Don't eat after 20 June.
Don't eat only after 20 June.



The addition of only does not change the meaning of the command.

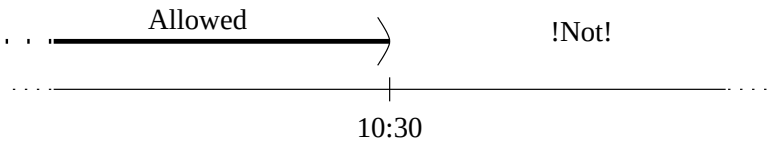
5.7.2 Instants

(E302) Eat until 10:30.
Eat up to 10:30



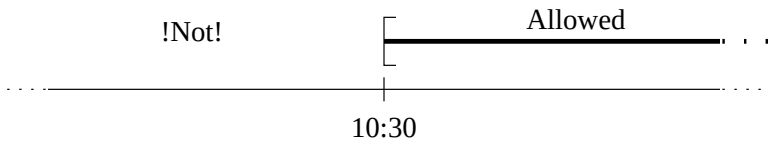
This command with until or up to has a meaning only because eating is an activity that can be done continually. If the command were something that can or should be done only once, then the command would not have meaning, as with Example E281. On and after 10:30, it is unspecified as to whether eating is allowed.

(E303) Eat only until 10:30.
Eat only up to 10:30.



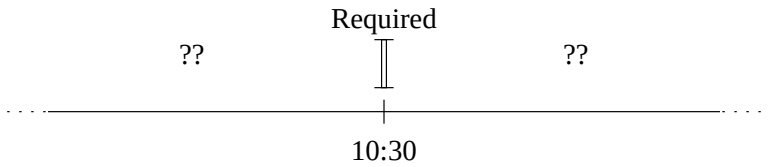
The addition of only to each command of Example E302 changes the command from a requirement to eat to a permission to eat, but in the same duration. Moreover, at and after 10:30, eating is definitely prohibited.

(E304) Don't eat until 10:30.
Don't eat up to 10:30.
Don't eat only until 10:30.
Don't eat only up to 10:30.



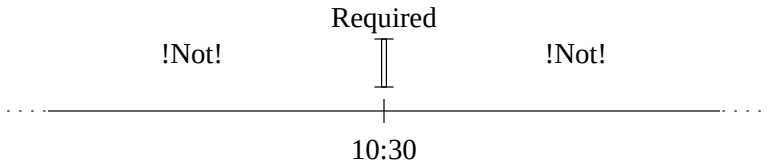
It is required not to eat until 10:30, and at and after that instant, eating is allowed. In each of these cases, adding only does not change the meaning.

(E305) Eat at 10:30.
Eat *duration* and including 10:30.



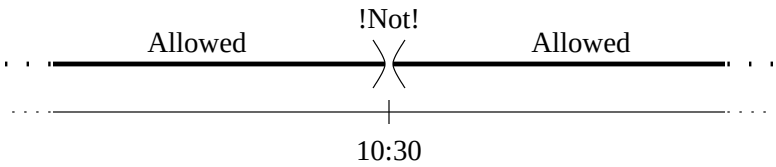
In the case of *duration* and including, the shown duration is united with that implied by the *duration*.

(E306) Eat only at 10:30.
Eat only *duration* and including 10:30.



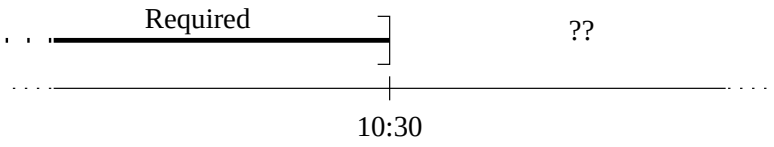
In the case of *duration* and including, the shown duration is united with that implied by the *duration*. The addition of the only to the commands of Example E305 changes each command from a requirement to eat at 10:30 to a permission to eat at 10:30 and makes it that eating is prohibited in times other than at 10:30.

(E307) Don't eat at 10:30.
Don't eat *duration* and including 10:30
Don't eat only at 10:30.
Don't eat only *duration* and including 10:30

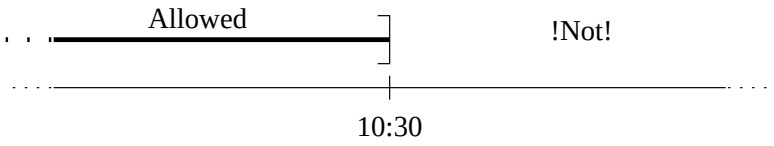


In the case of *duration* and including, the shown duration is intersected with that implied by the *duration*. The addition of only does not change the meaning of these negative commands.

(E308) Eat through 10:30.

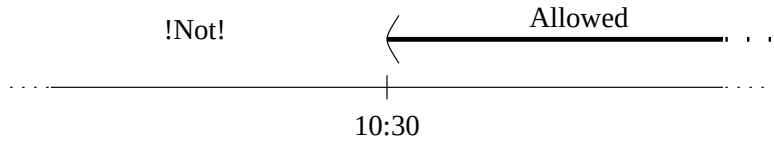


(E309) Eat only through 10:30.



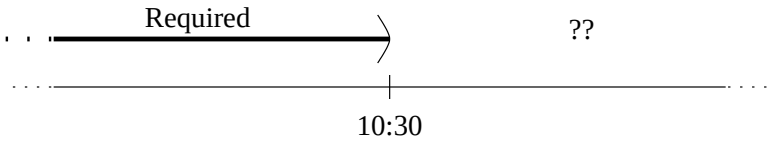
The addition of only to the command of Example E308 forces prohibition of eating after 10:30.

(E310) Don't eat through 10:30.
Don't eat only through 10:30.



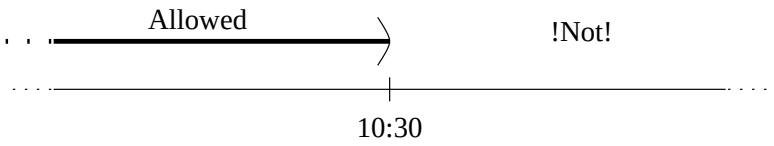
The addition of only does not change the meaning of the command.

(E311) Eat by 10:30.



The difference between Example E311 and Example E302 is that the former could mean that the eating is continual in the duration while the latter could mean that the eating happens once in the duration.

(E312) Eat only by 10:30.

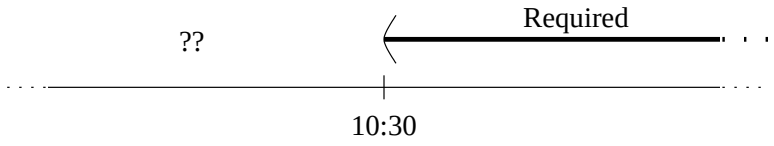


The addition of only to the command of Example E311 changes the command to eat to only allowing eating and forces a prohibition against eating at or after 10:30.

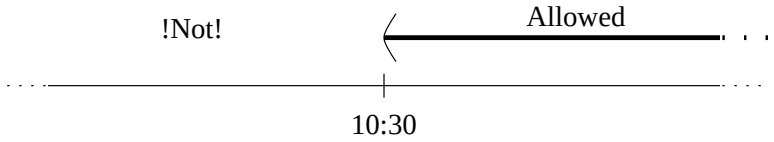
(E313) Don't eat by 10:30.
Don't eat only by 10:30.

For Example E313, there is no diagram because its commands have no real meaning.

(E314) Eat after 10:30.

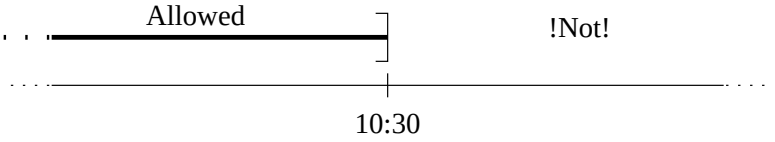


(E315) Eat only after 10:30.



The addition of only to the command of Example E314 changes the command to eat to only allowing eating and forcing a prohibition against eating before or at 10:30.

(E316) Don't eat after 10:30.
Don't eat only after 10:30.



The addition of only does not change the meaning of the command.

Note in Sections 5.7.1 and 5.7.2 that:

1. A positive command says that the eating is required in the duration and leaves unspecified whether eating is allowed not in the duration.
2. A negative command says that the eating is prohibited in the duration and only allows eating not in the duration.
3. Adding only to a positive command says that eating is only allowed, not required in the duration.
4. Adding only to a negative command does not change the meaning of the command.
5. In the duration, if eating is required, then not in the duration, it is unspecified whether eating is allowed.
6. Not in the duration, if eating is prohibited, then in the duration, eating is only allowed, not required.
7. The logical negation of eating is prohibited is eating is not prohibited, which is eating is allowed, but not required.

8. The logical negation of eating is required is eating is not required, which is it is unspecified whether eating is allowed.

5.7.3 By and Until

In English by and until are different. In practice, native speakers of English do not confuse them. However, as mentioned in the beginning of Section 5.7, many non-native speakers of English confuse them, both as writers and as readers, probably because in some other languages, the two words are translated as the same word. In some of these other languages, the direct translation has the meaning of only one of them and the other must be said in another way. In some of these other languages, the common translation of by and until is itself ambiguous, and the precise meanings of the English terms must be expressed with more involved terminology.

Consider four related sentences in English:

(E317) The money will be ready by 20 June.

(E318) The money will not be ready by 20 June.

(E319) The money will be ready until 20 June.

(E320) The money will not be ready until 20 June.

Explanations for them in other words are, respectively:

(E321) The money is probably not ready until 20 June, but on 20 June, and thereafter, it will be ready.

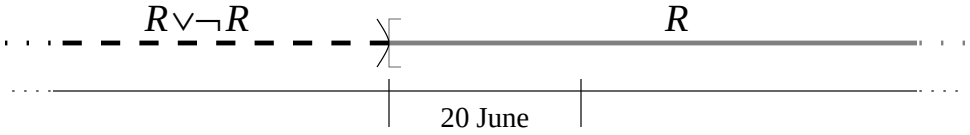
(E322) The money is not ready now and will not be ready any time before 20 June, but it will probably be ready on 20 June and thereafter.

(E323) The money is probably not ready any more on and after 20 June, but it is ready now.

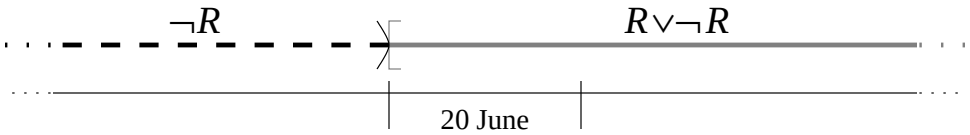
(E324) The money is not ready now and will not be ready any time before 20 June, but it will be ready on 20 June and thereafter.

The meanings can be shown graphically by putting spans covered by logical predicates over portions of a time line. The meaning of "probably X" in logic is at best "$X \vee \neg X$", because it is possible that X is false.

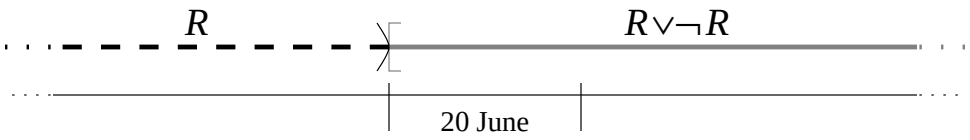
For E321:



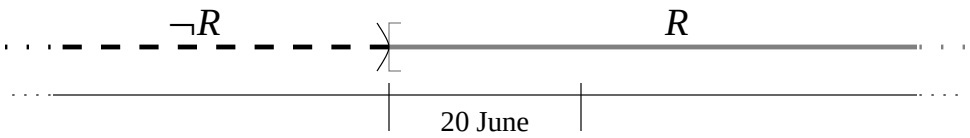
For E322:



For E323:



For E324:



Consider now the translation of these sentences into other languages:

1. These sentences are perfect candidates for expression with temporal logic. Hence, the last translation given below of each of these is a temporal logic expression, in which *R* means The money is ready now, and *20J* means It is now 20 June.
2. In Portuguese, by and until are translated as em and até, respectively.
3. In French, until is translated as jusqu'à. However, there is no direct translation for by; the meaning of a sentence using it has be be written indirectly. It is assumed that if the money is ready on 20 June, it stays ready thereafter.
4. In German, bis means both by and until; thus it is ambiguous. The speaker makes his intent clear by using instead of bis
   1. ab to establish a lower bound as needed for by, or
   2. bis einschließlich to establish an upper bound as needed for until.

   Thus, while it is acceptable to use the ambiguous bis, it is preferrable, for disambiguation, to use the more precise terms.
5. In Hebrew, by and until are translated as b' and 'ad, respectively.

(E325) The money will be ready by 20 June.
O dinheiro estará disponível em 20 de Junho.
L'argent sera disponible le 20 Juin.
Das Geld wird ab 20. Juni bereitgestellt.
Hakesef yehiyeh mukhan b'20 l'yuni.
$20J \supset \Box R$

(E326) The money will not be ready by 20 June.
O dinheiro não estará disponível em 20 de Junho.
L'argent ne sera pas disponible avant le 20 Juin.
Das Geld wird nicht ab 20. Juni bereitgestellt.
Hakesef lo yehiyeh mukhan b'20 l'yuni.
$\neg RU20J$

(E327) The money will be ready until 20 June.
O dinheiro estará disponível até 20 de Junho.
L'argent sera disponible jusq'au 20 Juin.
Das Geld wird bis einschließlich 20. Juni bereitgestellt.
Hakesef yehiyeh mukhan ad 20 l'yuni.
$RU20J$

(E328) The money will not be ready until 20 June.
O dinheiro não estará disponível até 20 de Junho.
L'argent ne sera pas disponible jusq'au 20 Juin.

Das Geld wird nicht bis einschließlich 20. Juni bereitgestellt.
Hakesef lo yehiyeh mukhan ad 20 l'yuni.
$(\neg R U 20J) \wedge (20J \supset \Box R)$

From the temporal logic statements, it is clear that the following relationships hold:

1. On or after 20 June,
   (The money will be ready by 20 June) ≡ (The money will not be ready until 20 June)
   and
   (The money will not be ready by 20 June) ≡ (The money will be ready until 20 June)
   i.e., on or after 20 June,
   $(20J \supset \Box R) \equiv ((\neg R U 20J) \wedge (20J \supset \Box R))$
   and
   $(\neg R U 20J) \equiv (R U 20J)$
2. Up to and not including 20 June,
   ((The money will not be ready until 20 June) ⊃ (The money will be ready by 20 June)) ∧ ((The money will be ready by 20 June) ⊅ (The money will not be ready until 20 June))
   and
   (The money will be ready until 20 June) ≡ ¬(The money will not be ready by 20 June)
   i.e., up to and not including 20 June,
   $(((\neg R U 20J) \wedge (20J \supset \Box R)) \supset (20J \supset \Box R)) \wedge ((20J \supset \Box R) \not\supset ((\neg R U 20J) \wedge (20J \supset \Box R)))$
   and
   $(R U 20J) \equiv \neg(\neg R U 20J)$
3. Always,
   (The money will not be ready until 20 June) ≡ ((The money will not be ready by 20 June) ∧ (The money will be ready by 20 June))
   i.e., always,
   $((\neg R U 20J) \wedge (20J \supset \Box R)) \equiv ((\neg R U 20J) \wedge (20J \supset \Box R))$

5.8 Ambiguity Even in Formalism

Even in highly mathematical text, ambiguities can be found. Berry has two experiences in which misinterpretating important natural language sentences in the statement of a theorem and in a definition delayed his understanding of the theorem and the definition. Both involve type mismatches, one in which an element is used to denote a set containing the element and another in which an object of totally the wrong type is used.

5.8.1 Element Denoting Whole Set

A context-free grammar (CFG), *G*, is said to be *LR*(*k*) if *G*'s language can be recognized by a left-to-right, bottom-up, parse involving a lookahead of not more than *k* characters. An *LR*(0) grammar requires no lookahead at all, and an *LR*(1) grammar requires a lookahead of not more than one character in some situations and no lookahead at all in all other situations. The *LR grammars* is the set of all CFGs *G* for which there is some $i \geq 0$ such that *G* is *LR*(*i*). That is

$$LR = \{G \mid G \text{ is a CFG} \wedge \exists i \geq 0\ (L \text{ is } LR(i))\}$$

Unfortunately, in the literature back in the early 1970s [42], when Berry was studying formal languages and automata theory,

(E329) *LR*(*k*) grammars

was used to denote both the entire class of *LR* grammars as well as the grammars that happen to be *LR*(*k*) for a

given *k* or for a *k* established as a bound variable. The literature ended up having to distinguish the different cases by saying

(E330) *LR*(*k*) for any *k*

for the first case and

(E331) *LR*(*k*) for the given *k*

or, for example,

(E332) *LR*(5)

for the second case and

(E333) ∃*k* ... *LR*(*k*)

or

(E334) Let *k* be ... *LR*(*k*)

for the third case.

This unfortunate choice of notation is confusing. When Berry read a discussion about a particular CFG *G* and then saw the claim,

(E335) *G* is not *LR*(*k*),

it was not clear whether (1) *G* was not *LR*(*k*) for the last *k* mentioned in the discussion or (2) *G* was not *LR*. The claim

(E336) *G* is not *LR*(*k*) for any *k*

is a little better, but is still confusing because the scope of the not was not completely clear. It would have been far less confusing, and in fact totally clear, if the claim had read

(E337) *G* is not *LR*,

or

(E338) There is no *k*≥0 for which *G* is *LR*(*k*).

5.8.2 Entirely the Wrong Type

The second example is the traditional definition of a *continuation* in denotational semantics of programming languages. It is supposed to be a

(E339) the function computed by the rest of the computation, following the current statement, *S*, from (the state of program variables after the execution of *S*) to (whatever is considered the answer of the computation).

That is, at any point in a computation, one can do the current statement *S* and then apply *S*'s continuation to the program variables after *S*'s execution to get the final answer. Unfortunately, at the time Berry was trying to learn denotational semantics for the first time, the literature, e.g. [35], gave as the definition of continuation

(E340) the function computed by the rest of the program, following the current statement, *S*, from (the state of program variables after *S*) to (whatever is considered the answer of the program).

The rest of the program is entirely a different animal from the rest of the computation. The rest of the program is a syntactic object, a program fragment, which is always finite. The rest of the computation is a semantic object, and it may not even be finite, that is, if the computation is nonterminating. In the rare case that the program has no loops, the rest of the program might very well correspond very tightly with the rest of the computation. However, in any normal situation with loops, the rest of the computation may wander through parts of the program that are not included in the literal rest of the program. This misstatement of the definition was confusing enough that it took Berry several years to recover from it.

6 Examples

This section gives examples of actual ambiguities found in several requirements specifications and in several legal documents.

6.1 Requirements

The examples are from various requirements specifications in the literature, including for the lighting system of a building, for a safety feature of a nuclear power plant, for the instruction manual of a toy, and for instructions for using a web site.

6.1.1 From a Requirements Case Study, the Light Control System

This subsection shows ambiguous sentences from the light control system (LCS) requirements specification prepared as a case study for a Dagstuhl Seminar held in June, 1999 on the subject, "Requirements Capture/Documentation/Validation". In each case, a problematic sentence is shown from the document, and it is followed by a better formulation based on the rules discussed above together with a brief statement of why the change was made. In each case, the author of the document has confirmed that the reformulation captured her intent. (For reasons not entirely clear, the identity of the document's author was shielded from us.)

1. Implicit All with *n*–*n* meaning

(E341) Sections are divided into offices(O), computer labs(CL), hardware labs(HL), peripheral rooms(P), meeting rooms(M), and hallways(H).

(E342) Each section is divided into some hallways (H) and rooms, each of which may be an office (O), a computer lab (CL), a hardware lab (HL), a peripheral room (P), or a meeting room (M).

The change was made to be more precise about what is true for each section.

2. All used as Any

(E343) Ceiling light groups in all rooms can only be turned on or off in groups.

(E344) The luminaires in a ceiling light group in any room are turned on or off only as a group.

The change was made to avoid the possible conclusion that a group may span several rooms.

3. All used as Each

(E345) In all rooms, each ceiling light group is controlled by one or more push-buttons, that toggle the light if switched to the other position.

(E346) In each room, each ceiling light group is controlled by one or more push buttons, each of which toggles the light group if switched to the other position.

The change was made to be more precise aobut what is true for each room and for each push button.

4. Implicit All used as Each

(E347) In the hallways, several push-buttons can toggle the ceiling light group on and off. All push-buttons are connected in parallel.

(E348) In each hallway, several push buttons can toggle the hallway's ceiling light group on and off. All and only the push buttons for one ceiling light group are connected in parallel.

The change was made to be more precise about what is true for each hallway, that a light group belongs to one hallway, and about how the push button of each group behaves.

5. A used as Each

(E349) An office (shown in Figure 2) has one door (d1) to the hallway and can have doors to the adjacent rooms (d2, d3).

(E350) Each office (shown in Figure 2) has one door (d1) leading to the hallway and can have up to two doors (d2, d3) leading to its adjacent rooms.

The change was made to avoid confusion over whether an is universal or existential and to make more precise the relationship of an office to its adjacent rooms.

6. Singular relative clause for All-induced plural noun

(E351) Each office is equipped with two ceiling light groups (window and wall), that can be dimmed individually with dimmer-actuators lle1 (window) and lle2 (wall)

(E352) Each office is equipped with two ceiling light groups (window and wall), each of which can be dimmed with its own dimmer-actuator (lle1 and lle2, respectively).

The change was made to make more precise what is true of each ceiling light group in an office.

7. Singular relative clause for All-induced plural noun

(E353) Each office is equipped with three status lines (sll1..3) that show the status of the three light sources.

(E354) Each office is equipped with three status lines (sll1, sl2, and sl3), each of which shows the status of one light source in the office.

The change was made to make more precise what is true of each status line.

8. Implicit All with *n*–*n* meaning

(E355) Staircases connect several floors.

(E356) Each staircase connects several floors.

The change was made to make more precise the relationship between staircases and floors.

9. Missing Each

(E357) At the floor level, a staircase is equipped with XXX.

(E358) At the landing of each staircase at each floor, the staircase is equipped with XXX.

The change was made to make more precise the relationship between a staircase and all of its XXXs.

10. Missing Each

(E359) A computer lab has one door (d1) to the hallway and can have doors to the adjacent rooms (d2, d3).

(E360) Each computer lab has one door (d1) leading to the hallway and can have other doors leading to adjacent rooms (d2, d3).

The change was made to avoid possible confusion over whether A is existential or universal.

11. Singular relative clause for All-induced plural noun

(E361) Each computer lab is equipped with two ceiling light groups (window and wall) that can be dimmed individually with dimmer-actuators lle1 (window) and lle2 (wall)

(E362) Each computer lab is equipped with two ceiling light groups, one for the window and one for the wall, each of which can be dimmed with its own dimmer-actuator, lle1 and lle2, respectively.

The change was made to be more precise about what is true of each light group.

12. Singular relative clause for All-induced plural noun

(E363) Each computer lab is equipped with two status lines (sll1, sll2) that show the status of the light sources.

(E364) Each computer lab is equipped with two status lines (sll1 and sl2) each of which shows the status of one light source in the computer lab.

The change was made to be more precise about what is true of each status line.

6.1.2 A Small Requirements Document

The ESFAS (Engineered Safety Feature Actuation System) requirements were originally introduced by Courtois and Parnas [15]. The ESFAS, as used in Belgian and Canadian PWR type nuclear power plants, prevents or mitigates damage to the core and coolant system when a fault such as loss of coolant occurs. The ESFAS receives data from some pressurizer's pressure sensors and determines whether an actuation signal called safety injection must be sent to the safety feature components that cope with pressure accidents. The human ESFAS operator has two push buttons to block the safety injection signal and to reset the blockage.

We have simplified the ESFAS requirements slightly. Rather than the original three independent sensors and a voting mechanism, we assume a single sensor that reliably measures the pressurizer's pressure. The ESFAS requirements are:

R1 The system monitors the pressure and sends the safety injection signal when the pressurizer's pressure falls below a 'low' threshold.

R2 The human operator can override system actions by turning on a 'Block' button and resets the manual block by pushing on a 'Reset' button.

R3 A manual block is permitted if and only if the pressure is below a 'permit' threshold.

R4 The manual block must be automatically reset by the system.

R5 A manual block is effective if and only if it is executed before the safety injection signal is sent.

R6 The 'Reset' button has higher priority than the 'Block' button.

The following problems are observed in the requirements; we give resolutions in the cases in which it is clear what the resolution should be:

- *Polysemy*: In R1, the phrase sends the safety injection signal can be interpreted as an action or a continuous activity. In the latter case, R1, R2 and R4 are also incomplete. That is, information is missing about what happens if (R1) the pressure rises above the 'low' threshold when sending the signal and if (R2, R4) a manual block is reset either by the operator or automatically. It is necessary to add an otherwise.
- *Potentially ambiguous reference*: In R2, the phrase overrides system actions could be ambiguous if the set of requirements were bigger. Currently, there is no system action other than sends the safety injection signal. It should have been written as: The human operator can override the safety injection signal by ....
- *Unstated assumption leading to ambiguity*: R2 has two meanings depending on the relation of the two thresholds 'low' and 'permit', i.e., 'permit' > 'low' or 'permit' ≤ 'low'. The first meaning is that the safety injection signal can be blocked when 'low' < water pressure < 'permit'. The second meaning is that the safety injection signal can never be blocked. One should take care not to ignore the second interpretation just because it conflicts with other requirements. Requirements do occasionally conflict, and a resolution has to be made explicitly rather than implicitly as a result of subconscious disambiguation. An additional requirement is needed stating the assumption that 'permit' > 'low'.
- *Ambiguous reference*: In R3, there is an ambiguity concerning the phrase manual block. When the pressure is not below a 'permit' threshold, either the event 'Block' button pressed is ignored or the state manual block is exited. Note that neither interpretation excludes the other. A fix would specify precisely what happens when the pressure is not below a 'permit' threshold.
- *Generality*: In R4, the phrase automatically reset by the system allows for a continuum of interpretations, and a fix would pin down precisely when the reset happens.
- *Polysemy*: In R5, the phrase A manual block is effective is polysemous. The blockage can be interpreted as a volatile state that is reset when the signal is sent or a persistent state that continues until the events described in R2 and R4 occur. A fix would explain how long the blockage lasts.

6.1.3 Selected Cases from an Experiment on Recognizing Ambiguity

A very effective, but expensive, way to analyze the level of ambiguity in a requirements document is to give it to several engineers, to ask for an interpretation, and to compare these interpretations afterwards. We did this in an experiment with a total of 58 computer science students. In the first part of the experiment, the students received a textual requirements document; their task was to model the requirements completely with several modeling techniques such as UML, Statemate, SDL, SCR, etc. In the second part of the experiment, a different set of students received a detailed checklist for ambiguity; the task was to inspect the requirements document. We discuss the most interesting ambiguities that we found during both parts of the experiment.

The employed requirements document describes the desired behavior of a consumer electronics product, the Tamagotchi toy. The toy simulates a virtual pet living in a plastic egg called Tamagotchi. The quality of the development of a pet depends on the care invested by the owner of the pet. The owner must feed the virtual pet, play with it, and monitor its health. Assuming sufficient care and attention, the Tamagotchi goes through several development stages. The plastic egg has a small LCD display with which to monitor the cyber chicken's development. To interact with the Tamagotchi, three buttons are placed underneath the display. A buzzer is integrated into the shell and is used for sounding alarms. All of the problems discussed below were written into the specification naturally; none were constructed to make the study more interesting. Some requirements were derived from a book describing the Tamagotchi [1], others were reverse engineered from the original toy, and some were invented.

1. In the requirements document, and in this section as well, the word Tamagotchi is used both as the name of the whole toy, i.e., the electro-mechanical device, as well as the cyber chicken simulated by the toy. This is not a problem for most of the requirements, because the way the word is used clarifies its meaning, as shown in the following requirement.

   (E365) The Tamagotchi dies when it is not fed for one day.

   Obviously, only the cyber chicken can die, i.e., that

   (E366) The Tamagotchi cyber chicken dies when it is not fed for one day.

   However, in some circumstances, both interpretations make perfectly sense such as in the following requirement.

   (E367) When the user inserts the paper strip, the Tamagotchi is set to its defaults.

   The requirement can mean that the whole toy or just the cyber chicken can be set to its defaults. Thus, it is necessary to write either

   (E368) When the user inserts the paper strip, the Tamagotchi toy is set to its defaults.

   or

   (E369) When the user inserts the paper strip, the Tamagotchi cyber chicken is set to its defaults.

   Polysemies are a larger problem in requirements documents than are homonyms. The meanings of a polysemy are related, thus, more detailed contextual information is necessary to disambiguate it. Polysemies of a term can occur despite the existence of a glossary defining the term, because the meanings of a term shifted during the project or because a term was not defined at all or precisely in the first place.

2. The following requirement is an example of a systematic polysemy.

   (E370) When the user presses the L- and R-button simultaneously, the alarm is turned off.

   The phrase turned off can refer to an alarm that is currently sounded by the system or to the general ability of the system to raise alarms. The requirement should be written as one of:

   (E371) When the user presses the L- and R-button simultaneously, the currently sounded alarm, if any, is turned off.

   (E372) When the user presses the L- and R-button simultaneously, the ability to sound an alarm is turned off.

   Commonly used words and phrases, such as turn off, are usually not described in glossaries, because they seem to be unambiguous.

3. The following requirement is an example of referential ambiguity:

(E373) When the user pulls the paper strip, the cyber chicken is born. ... After its first night, the cyber chicken becomes a Marutchi. ... **After 2 days**, the cyber chicken becomes a Tamatchi (friendly teen).

The phrase After 2 days refers to a contextually specified event; two days after this event, the cyber chicken changes its state. Since there are more than one event described in the requirement and in the requirements document that can be referenced by the phrase After 2 days, the requirement is ambiguous. The solution is to write one of:

(E374) Two days after birth, the cyber chicken becomes a Tamatchi (friendly teen).

(E375) Two more days later, the cyber chicken becomes a Tamatchi (friendly teen).

4. The following requirement is another example of an ambiguous reference; first contextual information and then the requirement of interest are provided.

(E376) The user can play with the cyber chicken. A game is started by choosing the menu item "Play" in the main menu. The display shows a playing chicken. A game has five rounds and can be won or lost. One game follows another; a game can be aborted by the R-button. ... The user must **play** at least twice a day to develop the Tamatchi character of the cyber chicken.

Ignore the fact that the idea of the game is not clear from these sentences. A typical discourse ambiguity occurs in the last sentence. The verb play refers to the process of playing with the cyber chicken. This process can take various courses, a game can be won or lost, a game can be aborted immediately after being started, etc. Therefore, the meaning of play is ambiguous. The requirement should be written as:

(E377) The user must chose the "Play" menue item at least twice a day to develop the Tamatchi character of the cyber chicken.

5. The following requirement provides an example of a conceptual ambiguity:

(E378) In the Marutchi state, the Tamagotchi has a lifespan of 15 to 22 days.

The requirement can be interpreted (1) as describing the minimum and maximum of the lifespan, 15 and 22 days, and leaving undefined the initial value or (2) as describing the initial value of the lifespan, which is a random value in the interval 15 to 22 days, and leaving undefined the minimum and maximum. Two possible unambiguous statements of the requirement would be:

(E379) In the Marutchi state, the Tamagotchi has a lifespan of between 15 and 22 days inclusive.

(E379) In the Marutchi state, the Tamagotchi has a lifespan that is initially some random number of days between 15 and 22 inclusive.

6.1.4 From Directions for Submitting a Proposal

The following examples were found in the directions for submitting a proposal to NSERC, the Canadian scientific research funding agent.

1. Singular object in a plural sentence.

(E381) All NSERC applicants have been or will be given a personal identification number (PIN).

This requirement needs to be changed to one of:

(E382) All NSERC applicants have been or will be given personal identification numbers (PINs).

(E383) Each NSERC applicant has been or will be given a personal identification number (PIN).

2. Singular object in a plural sentence.

(E384) For group applications, list co-applicants, their personal identification number, their organization, and the time (in hours per month) they will devote to the proposed research or to the use of the equipment or facility.

This requirement needs to be changed to one of:

(E385) For group applications, list co-applicants, their personal identification numbers, their organizations, and the time (in hours per month) they will devote to the proposed research or to the use of the equipment or facility.

(E386) For group applications, list co-applicants, and for each, his or her personal identification number, his or her organization, and the time (in hours per month) he or she will devote to the proposed research or to the use of the equipment or facility.

In each of these cases, the second correction is better since it makes clear that for each person there is a unique PIN, her own organization, and her own amount of time that will be devoted to the work.

6.2 Legal Documents

The legal examples are from a software license agreement, the statement of a law, clauses of constitutions, and from instructions for filling out income tax forms.

6.2.1 From A License Agreement

In the Adobe Systems Incorporated End-User License Agreement, there are three sentences involving only:

(E387) Trademarks can only be used to identify printed output produced by the Software...

This Agreement may only be be modified by a writing signed by an authorized officer of Adobe,...

... the ... Software and ... Documentation are being licensed to U.S. Government end users (A) only as Commercial Items and (B) with only those rights as are granted to all other end users,....

Only the third sentence has all of its onlys in the correct places. The probable intents of the first two sentences are:

(E388) Trademarks can be used only to identify printed output produced by the Software...

This Agreement may be be modified by only a writing signed by an authorized officer of Adobe,...

The Microsoft Office 2000 Standard Academic Edition License has two sentences involving only:

(E389) You may also store or install a copy of the SOFTWARE PRODUCT on a storage device, ... used only to RUN the SOFTWARE PRODUCT on your other COMPUTERS over an internal network; ....

You may not reverse engineer, decompile, or disassemble the SOFTWARE PRODUCT, except and only to the extent that such activity is expressly permitted by applicable notwithstanding this limitation.

Interestingly, both sentences use only correctly. However, the first sentence has its also in the wrong place; that sentence should be written:

(E390) You may store or install a copy of the SOFTWARE PRODUCT also on a storage device, ... used only to RUN the SOFTWARE PRODUCT on your other COMPUTERS over an internal network; ....

The Full One Year Warranty that comes with a Hoover vacuum cleaner has a sentence using only in a wrong place.

(E391) Warranty service can only be obtained by presenting the appliance to one of the following authorized warranty service outlets....

The sentence should have said:

(E392) Warranty service can be obtained only by presenting the appliance to one of the following authorized warranty service outlets....

6.2.2 From Statement of a Law

The WWW site of the State of California's Legislative Analyst's Office,

http://www.lao.ca.gov/initiatives/2000/22_03_2000.html,

describes Proposition 22, titled "Limit on Marriages", by one sentence.

(E393) This measure provides that only marriage between a man and a woman is valid or recognized in California.

The only is in the wrong place, and the word or is probably wrong too. The reason we say only "probably" is that the meaning of the or in conjunction with the misplaced only is simply not clear. However, normally, the or would be wrong simply because the law would be satisfied if only one of validity and recognition held. On the other hand, the author *did* keep to singular!

The proposition, as written, says:

(E394) Only marriage between a man and a woman is valid in California,
or
only marriage between a man and a woman is recognized in California.

Only one of the two clauses needs to be true, and both say that the only relationship that is valid between a man and a woman is marriage. Consequently, there are no friendship, no dating, no engagement, no sex, no separation, no

divorce, no nothing but marriage between men and women in California. Interpreted literally, it is making friendship, dating, engagement, sex, separation, and divorce between men and women illegal in California.

The intent of the proposition is to make same-gender marriage illegal in California. A correct statement of what is intended is:

(E395) The only kind of marriage that is valid and recognized in California is that between a man and a woman.''

Perhaps this would be enough to get the law derived from the proposition overturned in the state's Supreme Court.

When we showed this analysis to Steve Easterbrook. He argued that in practice, there was no real problem with this proposition.

> This is an interesting one. The working definition of ambiguity that we developed for analysing the Space Station documents was that if we gave the specification to lots of different people (presumably representing the different audiences for the document), would they interpret it in different ways.
>
> The interesting thing about proposition 22 is that although it's hopelessly poorly phrased, I don't imagine I could find anyone who does not immediately understand the intended meaning of the proposition. Even those that pick apart the grammar have no difficulty understanding the intended meaning.
>
> Hence, as far as I'm concerned, this one passes my test: there is no ambiguity. My point is that ambiguity is determined much more by context than from the actual wording.

We had to agree with him that everyone would understand the intent of the proposition, and that most would not even know that it was incorrectly worded. However, we pointed out other factors.

1. It is possible that something like this would be enough to get the law declared invalid.
2. In the context of an SRS, if a reader were faced with one of these incorrect, but generally understood sentences *and* the sentence as written had another meaning *and* the reader were careful, the reader would begin to wonder which is intended, especially when there are both correctly used and incorrectly used sentences in the same document.
3. In this day of people moving around, many a document is written by someone whose native language is not the language of the document. This fact creates a real dilemma on the reader. The reader must guess whether or not the writer unfamiliar with the general usage and whether or not the writer unfamiliar with the grammar. The author Berry knows that his English is better than that of most, native or non-native. However, his Hebrew is strange. His grammar is more correct than that most Israelis because he learned Hebrew via the grammar, but his word choices are abominable. For example, Berry correctly says the Hebrew equivalent of David or Moshe is coming. Most Israelis say the equivalent of David or Moshe are coming and are prepared to argue that it is correct. So, how should one interpret his Hebrew?

6.2.2 From Clauses of Constitutions

The Eighth Amendment to the U.S. Constitution specifies that,

(E396) Excessive bail shall not be required, nor excessive fines imposed, nor cruel and unusual punishments inflicted.

It precludes punishments that are cruel and unusual. In contrast, Section 17 of Article 1 of the Constitution of the State of California in the U.S. specifies that,

(E397) Cruel or unusual punishment may not be inflicted or excessive fines imposed.

Thus, the State of California's constitution precludes any punishment that is cruel or unusual.

Several lawyers have explained that the U.S. Constitution, thus, permits the death penalty. For a punishment to be disallowed in the U.S, it must be both cruel *and* unusual. Even if one believes that the death penalty is cruel, it is by no means unusual. Therefore, it is not disallowed. Conversely, California is obliged to disallow the death penalty since the death penalty is cruel and thus is at least one of cruel *or* unusual.

Whether or not this explanation was the basis for the court decisions that make the death penalty legal at the U.S. level and illegal in the State of California, the legal point is well taken. The writers of a contract must be careful about which of and and or they use. If they are not, they may find some readers of the contract interpreting provisions differently from what is intended.

The situation is made more complex because in the presence of a negative, i.e., not, the meanings of and and or in a sense flip. Recall that in logic, $\neg (A \wedge B) \equiv (\neg A) \vee (\neg B)$ and $\neg (A \vee B) \equiv (\neg A) \wedge (\neg B)$. If one is not careful, she can easily write a sentence that means exactly the opposite of what is intended.

6.2.4 From the U.S. Internal Revenue Service

In the Acrobat version of *Your Federal Income Tax*, For Individuals, Publication 17 for the year 2000, published by the U.S. Internal Revenue Service [3], we searched for the word all, independent of case. Of the 43 uses of all through the end of Chapter 1, not counting the ALL that appeared in the phone number 1-888-ALL-TAXX, none had any of the problems we have noted for all. A random check of some of the occurrences of all in the rest of the document showed no problems. The most common use of all is to specify that something cannot be done unless *all* conditions listed below are met.

There were three occurrences of everyone in the entire document, each of which modifies only a singular noun. There were 25 occurrences of their through the end of Chapter 6, and each modifies a plural noun. One of the two examples that came close to being problematic is:

(E398) A married couple filing a joint return can contribute up to $2,000 each to their IRAs, even if one spouse had little or no income.

It might have been written a little more clearly as:

(E399) Each spouse of a married couple filing a joint return can contribute up to $2,000 to his or her IRA, even if one spouse had little or no income.

The other example that came close to be being problematic is:

(E400) You may also obtain the convenience fee by calling the service provider's automated customer service telephone number or visiting their respective web site.

The problem is that in U.S. English an organization is normally considered singular even though it is made up of many members. In British English, a multi-member organization is considered plural. Thus the example is correct in British English. In U.S. English, it would have been better to write:

(E401) You may also obtain the convenience fee by calling the service provider's automated customer service telephone number or visiting its web site.

There are 27 occurrences of only through the end of Chapter 1. Surprisingly, only three of them are placed wrongly. The other 24 of them are place correctly, including a few that might generally have been placed incorrectly. The three incorrectly placed onlys are:

(E402) Previously the child only had to meet (2) and (3) to be an eligible foster child.

(E403) Generally, you will only need to report the sale of your home if your gain is more than $250,000 ($500,000 if married filing a joint return).

(E404) On Form 1040EZ, you can only use the tax table to figure your tax.

The meanings of these incorrect sentences are nonsense. Thus, nearly all will understand them correctly. Corrected versions of these sentences are:

(E405) Previously the child had to meet only (2) and (3) to be an eligible foster child.

(E406) Generally, you will need to report the sale of your home only if your gain is more than $250,000 ($500,000 if married filing a joint return).

(E407) On Form 1040EZ, you can use only the tax table to figure your tax.

The following are some examples of the many sentences with correctly placed onlys:

(E408) ... you pay only the tax you owe and no more.

(E409) This authorization applies only to the individual whose signature appears in the in the ''Paid Preparer's Use Only'' section of your return.

(E410) Your adjustments to income are for only the following items.

(E411) You claim only the following credits.

(E412) ... she would receive only the difference between her regular salary and the salary of a substitute teacher hired by the school board.

That these are written correctly is surprising since in normal writing these would probably have been written as:

(E413) ... you only pay the tax you owe and no more.

(E414) This authorization only applies to the individual whose signature appears in the in the ''Paid Preparer's Use Only'' section of your return.

(E415) Your adjustments to income only are for the following items.

(E416) You only claim the following credits.

(E417) ... she only would receive the difference between her regular salary and the salary of a substitute teacher hired by the school board.

Perhaps each of these examples were done correctly, because the more common, incorrect form of each could be misinterpreted as something contrary to the tax law. The very first usage of only in Chapter 2,

(E418) A marriage means only a legal union between a man and a woman as husband and wife.,

is interesting in view of Example E393 of Section 6.2.2. The IRS tax law captures correctly the restriction that Proposition 22 of the State of California incorrectly specifies.

Thus, it appears that the people who write the U.S. income tax laws and guide books are generally careful about writing precisely. Of course, they may have learned from bad experiences with many a taxpayer successfully and legally playing each incorrectly specified provision as a loophole.

7 Writing Guides

This section lists and describes other sources on

- writing in general,
- writing technical documents in general,
- writing requirements specifications,
- writing legal documents, and
- avoiding and finding ambiguities.

In these documents is a lot of good advice on writing that is of relevance to the writer of requirements specifications or legal documents, advice that can improve the overall quality of the documents beyond just avoiding ambiguities. A document that does not appear in this section does not appear because either we do not know about it or we believe that it is not very helpful. If you, as a reader, feel that a given document has been left out, please inform the author Berry at the e-mail address given on the title page.

7.1 General Writing

Below is a list of general writing guidelines:

1. *A Dictionary of Modern English Usage* [24], by H.W. Fowler: The second edition of this book, which is already in its third edition, is a favorite of many, and it covers many of the issues covered in this handbook; it even has a section on ambiguity as a phenomenon that many consider a gem.

2. *A Handbook for Scholars* [77], by Mary-Claire van Leunen: This book focuses on mainly word choice issues, such as may vs. can. The words involved are not considered ambiguous. However, when the wrong choice is made, the containing sentence says other than what is intended.

3. *Elements of Style* [76], by William Strunk, Jr. and E.B. White: This is an old favorite writing guide that discusses the general problem of matching number and gender between subjects and verbs and between pronouns and their antecedents. It reminds authors that each and every are singular. It observes the problem with one as an existential quantifier, but recommends the opposite of what we do, that pronouns referring to it be one and not he.

4. *The Elements of Grammar* [72], by Margaret Shertzer: This book, is intended as a companion to *The Elements of Style* by Strunk and White, covers grammatical issues not covered by Strunk and White, some of which figure in ambiguities.

5. *Harbrace College Handbook for Canadian Writers* [41], by John C. Hodges, Mary E. Whitten, Judy Brown, and Jane Flick: This book and its U.S. cousin are old favorite writing guides. It covers a variety of issues including the only problem, the general problem of matching number and gender between subjects and verbs and between pronouns and their antecedents, and the fact that each and every are singular.

6. *Bugs in Writing* [19], by Lyn Dupré: Of all the writing guides we have seen, this one covers more of the issues discussed in this handbook than any other writing guide, general or specific to requirements specification or legal documents. This writing guide was written by an editor for Addison-Wesley who has edited many computer science books and has noticed many problems in the highly technical writing of the area.

7.2 Technical Writing

Below is a list of guidelines for technical writing:

1. *Mathematical Writing* [51], by Donald Knuth, Tracy Larrabee, and Paul Roberts: This book is a series of notes from lectures by the authors for a course on mathematical writing offered at Stanford University. It aims to improved the precision and clarity of mathematical writing, by, for example, insisting that formulae be woven into full, properly constructed sentences. One point it argues for is proper placement of onlys.

2. *Writing for Computer Science* [81], by Justin Zobel: This book covers the proper use of all and some as quantifier equivalents and using specific technical terms precisely.

7.3 Requirements Specification Writing

Below is a list of guidelines for writing requirements specifications:

1. *Guide to Specification Writing* [63], by John Oriel: This book, directed at U.S. Government Engineers writing specifications for engineering work for the U.S. Government warns that ambiguity is construed in favor of the contractor rather than the Government.

   > The rule is that the contractor interprets the specifications, as long as the interpretation is reasonable. The Government is responsible for furnishing sufficiently clear and complete language to evoke the intended understanding in the reader, and the Government is also responsible for any expenses that may be incurred of the contractor does not interpret the specifications as intended.

   Oriel reminds the specification writer of the harsh truth about the cost of fixing errors.

   > When an error is made, the cost of correcting it is greatest when it was made early in the process and not discovered until subsequent work has been done. The earlier in the process that the error was made, the more work has to be done over. This is why we must pay very careful attention to project planning and specification writing: they are the foundation for all the rest of the work.

   The guide book covers many, but not all, issues covered herein, and covers them in the context of specification writing.

2. *Practical Software Requirements: A Manual of Content and Style* [52], by Ben Kovitz: This book describes in detail how to write good requirements and specifications starting from a problem, as opposed to a solution. About one third of the book deals with important stylistic issues, which do not intersect with the ambiguity issues discussed herein.

3. *Handbook of Walkthroughs, Inspections, and Technical Reviews: Evaluating Programs, Projects, and Products* [25], by Daniel P. Freedman and Gerald M. Weinberg is nominally about inspections. However, one of the things that inspectors look for when inspecting requirements specifications is ambiguities. The book has a good checklist of possible ambiguities.

7.4 Legal Document Writing

Below is a list of guidelines for writing legal documents:

1. *The Language of the Law* [57], by David Mellinkoff: The book's preface is instructive.

   > The law is a profession of words. Yet in a vast legal literature the portion devoted to the language of the law is a single grain of sand at the bottom of a great sea.... Still, at this writing [in 1963], the subject of "language" is absent from most law indexes and only in capsule form in the rest.

   This book deals with the history of legal language and then discusses a large variety of phrases of dubious meaning that appear in many legal documents, including, and/or. Mellinkoff's suggestion for *A* and/or *B* is simply *A* or *B* or both.

   The books that follow seem to rise to the challenge raised by Mellinkoff in 1963.

2. *The Elements of Legal Style* [28], by Bryan Garner: This book applies general writing advice to legal writing and has a brief section on ferreting out ambiguities. Interestingly, its discussion of the only problem, begins with an exception, which is apparently intentional.

   > **only.** Though we have old exceptions (God only knows now many), in editorial English the best placement of *only* is directly before the words to be limited by it. The more words separating *only* from its correct position, the more awkward the sentence and the greater the probability of ambiguity.

3. *The Lawyer's Guide to Writing Well* [34], by Tom Goldstein and Jethro Lieberman: This book focuses on general writing and points to the only problem as a source of confusion.

4. *Effective Legal Writing for Law Students and Lawyers* [11], by Gertrude Block: This book covers many, but not all, of the ambiguity issues identified herein.

5. *Expert Legal Writing* [53], by Terri Leclercq and Thomas Phillips: This book discusses the and/or and only problems as impediments to clear legal writing.

7.5 Avoiding and Finding Ambiguities

Below is a list of guidelines for avoiding and finding ambiguities:

1. *Collection of Ambiguous or Inconsistent/Incomplete Statements* [37], by Jeff Gray: This web page, at http://www.gray-area.org/Research/Ambig, notes that ambiguity is hard to avoid, that it can be the source of humor, but is a serious problem in requirements specifications. It has several lists of mostly humorous, but sometimes tragic, ambiguous, inconsistent, and incomplete statements, some of which come from specifications and legal documents.

8 Conclusions

This handbook attempts to identify common ambiguities that make many SW requirements specifications and many legal contracts difficult to interpret correctly. This handbook describes commonalities between software requirements specifications and contracts, both in the way that they are developed for and with clients and in the their necessity to be precise and to anticipate all possible contingencies. The handbook discusses a variety of definitions of ambiguity, including linguistic, legal, and software engineering. It introduces a new source of ambiguity, namely language errors committed by the writer, the reader, or both. The handbook describes general techniques for avoiding and detecting ambiguities. The handbook gives a lengthy discussion with examples of these language-error ambiguities. It then shows examples of many of these in various requirements and legal documents. The handbook concludes with pointers to writing style guides that help improve general, requirements, and legal writing; some of these guides mention some but not all the ambiguities described in this handbook.

The definitions and examples of ambiguity provided in Section 3 can be used by an attorney or RE practitioner to raise her awareness of the various facets of ambiguity. In our experience, everyone is aware of the potential for ambiguity, but few appreciate how much there can be, and are simply not aware of how ambiguous writing in law, including contracts, or software engineering, including requirements specifications, can be. Once ambiguity is recognized as a problem, then the examples of Sections 3, 5, and 6, the techniques of Section 4, and the guides listed in Section 7 can be used to avoid and detect ambiguous contracts and specifications.

The writer can use this material to help avoid introducing ambiguities into the contracts or specifications she is writing. The inspector can use this material to help detect ambiguities in the contracts specifications she is inspecting. The writer should build a check list of her recurring ambiguities, found repeatedly by inspectors, in order to be able to reduce their incidence in future contracts or specifications she writes. The inspector should build a check list of recurring ambiguities found in the specifications of each writer whose work she inspects, in order to improve her ambiguity detection effectiveness.

Even with all the systematic techniques and modeling, avoiding and detecting ambiguities is at best an art. Fundamentally and ultimately, an ambiguity is anything that causes different people to understand differently; unfortunately, the set of people that happen to examine a document may just not have a person that detects an understanding that demonstrates an ambiguity. Therefore, it will be necessary to improve our skills at avoiding and detecting ambiguities. We need to teach lawyers and requirements engineers to write unambiguously. We need to teach lawyers and requirements engineers to spot hidden ambiguities.

Note that this handbook was written very carefully, with every effort paid to following our own advice and avoiding ambiguities. Indeed, the reader may have noted some unusual word order, especially with any word whose position in its containing sentence affects its meaning, e.g., *also* and *only*. A careful look at instances of these words will show that we are indeed following our own advice. However, despite our best efforts, it is possible that ambiguities remain. This point was driven home to us when we noted an ambiguity in a sentence as we were producing the final version. We fixed that problem, but just as with bugs, there is no last ambiguity!

The reader is encouraged to send additional examples of ambiguities, of types discussed in this handbook or of other types, to the author Berry at dberry@uwaterloo.ca.

Finally, the authors hope that they will see fewer ambiguities in the legal documents and requirements specifications that they examine in the future.

Acknowledgments

We thank Yael Berry, Karin Breitman, Robin Cohen, Yann d'Halluin, Leah Goldin, Eli Kantorowitz, Estelle Lavoie, Julio Leite, Maria Nelson, Torsten Nelson, Isabel Ramos, Colette Rolland, and Ursula Thoene for their help with the sentences in languages other than English. We thank Mina Askari for her help finding errors. We thank Cathy Kay, a lawyer, for her explanations of how she avoids some of these ambiguity problems in contracts and other documents that she writes. The authors thank Michael Godfrey, Axel van Lamsweerde, and Roel Wieringa for pointing the authors to relevant literature. We thank Reinhard Gotzhein for verifying with the author of a document that we understood the author correctly. Finally, we thank Steve Easterbrook, David Kay, and Martin Feather for discussions about some of the issues.

D.M. Berry's work was supported in parts by RENOIR, Requirements Engineering Network Of International cooperating Research groups, a European Union/ESPRIT network of excellence, project number: 20.800 (1996-2000), a University of Waterloo Startup Grant, and by NSERC grant NSERC-RGPIN227055-00.

Bibliography

[1] *Das Original Tamagotchi Buch*, Tamagotchi & Bandai (1997).

[2] "Merriam-Webster's Collegiate Dictionary", Tenth Edition, Merriam-Webster, Springfield, MA (1998), http://www.m-w.com/dictionary.htm.

[3] "Your Federal Income Tax, For Individuals, Publication 17", U.S. Internal Revenue Service, Washington, DC (2000), http://www.irs.gov/.

[4] Allen, J., *Natural Language Understanding* , Second Edition, Addison-Wesley, Reading, MA (1995).

[5] Bach, K., "Ambiguity", in *Routledge Encyclopedia of Philosophy*, ed. E. Craig and L. Floridi, Routledge, London, UK (1998).

[6] Basili, V., Caldiera, G., Lanubile, F., and Shull, F., "Studies in Reading Techniques", pp. 59–65 in *Proceedings of the 21st Annual Software Engineering Workshop*, Software Engineering Laboratory Series, SEL-96-002, Goddard Space Flight Center, Greenbelt, Maryland, USA (December 1996).

[7] Ben Achour, C., "Guiding Scenario Authoring", pp. 152–171 in *Proceedings of the Eighth European-Japanese Conference on Information Modeling and Knowledge Bases*, IOS Press, Vamala, Finland (25–29 May 1998).

[8] Berry, D.M., "The Importance of Ignorance in Requirements Engineering", *Journal of Systems and Software* **28**(2), pp. 179–184 (February 1995).

[9] Berry, D.M. and Kamsties, E., "The Dangerous 'All' in Specifications", pp. 191–194 in *Proceedings of 10th International Workshop on Software Specification & Design, IWSSD-10*, IEEE CS Press (2000).

[10] Black, H.C., *Black's Law Dictionary* , Revised Fourth Edition, West Publishing, St. Paul, MN (1968).

[11] Block, G., *Effective Legal Writing for Law Students and Lawyers* , Fourth Edition, Foundation, Westbury, NY (1992).

[12] Boehm, B.W., *Software Engineering Economics*, Prentice-Hall, Englewood Cliffs, NJ (1981).

[13] Bostrom, R.P., "Succesful Application of Communication Techniques to Improve the Systems Development Process", *Information & Management* **16**, pp. 279–295 (1989).

[14] Buley, E.R., Moore, L.J., and Owess, M.F., "B5 (SRS/IRS) Specification Guidelines", Technical Report M88-57, ESD-TR-88-337, MITRE, Bedford, MA, USA (December 1988).

[15] Courtois, P.-J. and Parnas, D.L., "Documentation for Safety Critical Software", pp. 315–323 in *Proceedings of the Fifteenth International Conference on Software Engineering*, Baltimore, MD (May 1993).

[16] Cruse, D.A., *Lexical Semantics* , Cambridge Textbooks in Linguistics, Cambridge University Press, Cambridge, UK (1986).

[17] Davis, A., *Software Requirements: Objects, Functions, and States*, Prentice Hall, Englewood Cliffs, NJ (1993).

[18] DeMillo, R.A., Lipton, R.J., and Perlis, A., "Social Processes and Proofs of Theorems and Programs", *Communications of the ACM* **22**(5), pp. 271–280 (1979).
[19] Dupré, L., *Bugs in Writing: A Guide to Debugging Your Prose*, Addison-Wesley, Reading, MA (1998).
[20] Easterbrook, S. and Nuseibeh, B., "Managing Inconsistencies in an Evolving Specification", pp. 48–55 in *Proceedings of the Second IEEE International Symposium on Requirements Engineering*, IEEE Computer Society Press, York, UK (March 1995).
[21] Easterbrook, S. and Nuseibeh, B., "Using ViewPoints for Inconsistency Management", *Software Engineering Journal* **11**(1), pp. 31–43 (January 1996).
[22] Easterbrook, S.M. and Callahan, J.R., "Formal Methods for Verification and Validation of Partial Specifications: A Case Study", *Journal Systems and Software* **40**(3), pp. 199–210 (March 1998).
[23] Fabbrini, F., Fusani, M., Gnesi, G., and Lami, G., "An Automatic Quality Evaluation for Natural Language Requirements", in *Proceedings of the Seventh International Workshop on RE: Foundation for Software Quality (REFSQ'2001)*, Interlaken, Switzerland (4–5 June 2001).
[24] Fowler, H.W., *A Dictionary of Modern English Usage* , Second Edition, Oxford University, Oxford, UK (1965).
[25] Freedman, D.P. and Weinberg, G.M., *Handbook of Walkthroughs, Inspections, and Technical Reviews: Evaluating Programs, Projects, and Products*, Dorset House, New York, NY (1990).
[26] Fuchs, N.E. and Schwitter, R., "Attempto Controlled English (ACE)", in *Proceedings of the First International Workshop on Controlled Language Applications (CLAW'96)*, Belgium (March 1996).
[27] Fuchs, N.E., Schwertel, U., and Schwitter, R., "Attempto Controlled English (ACE) Language Manual Version 3.0", Technical Report Nr. 99.03, Institut für Informatik der Universität Zürich, Zürich, Switzerland (1999).
[28] Garner, B.A., *The Elements of Legal Style*, Oxford University, New York, NY (1991).
[29] Gause, D.C. and Weinberg, G.M., *Exploring Requirements: Quality Before Design*, Dorset House, New York, NY (1989).
[30] Gause, D.C., "User DRIVEN Design—The Luxury that has Become a Necessity, A Workshop in Full Life-Cycle Requirements Management", ICRE 2000 Tutorial T7, Schaumberg, IL (23 June 2000).
[31] Gervasi, V., "Environment Support for Requirements Writing and Analysis", Ph.D. Dissertation, TD-3/00, Dipartamento di Informatica, Università di Pisa, Pisa, Italy (2000).
[32] Gervasi, V. and Nuseibeh, B., "Lightweight Validation of Natural Language Requirements", pp. 140–148 in *Proceedings of Fourth IEEE International Conference on Requirements Engineering (ICRE'2000)*, IEEE Computer Society Press, Schaumburg, IL (19–23 June 2000).
[33] Goguen, J.A., "Requirements Engineering as the Reconciliation of Technical and Social Issues", pp. 165–199 in *Requirements Engineering: Social and Technical Issues*, ed. J.A. Goguen and M. Jirotka, Academic Press, London, UK (1994).
[34] Goldstein, T. and Lieberman, J.K., *The Lawyer's Guide to Writing Well*, University of California, Berkeley, CA (1991).
[35] Gordon, M.J.C., *The Denotational Description of Programming Languages: An Introduction*, Springer-Verlag, Berlin (1979).
[36] Gotel, O.C.Z. and Finkelstein, A.C.W., "An Analysis of the Requirements Traceability Problem", pp. 94–101 in *Proceedings of the IEEE International Conference on Requirements Engineering*, IEEE Computer Society Press, Colorado Springs, CO (1994).
[37] Gray, J., "Collection of Ambiguous or Inconsistent/Incomplete Statements", Vanderbilt University, Nashville, TN (2000),
http://www.gray-area.org/Research/Ambig/.
[38] Hadad, G., Kaplan, G., Oliveros, A., and Leite, J.C.S.P., "Integración de Escenarios con el Léxico Extendido del Lenguaje en la Elicitación de Requerimientos: Aplicación a un Caso Real", *Revista de Informática Teórica e Aplicada (RITA)* **6**(1), pp. 77–104 (2000).
[39] Harwell, R., Aslaksen, E., Hooks, I., Mengot, R., and Ptack, K., "What is a Requirement?", pp. 17–24 in *Proceedings of the Third Annual International Symposium*, National Council of Systems Engineers (NCOSE) (1993).

[40] Hirst, G., *Semantic Interpretation and the Resolution of Ambiguity* , Studies in Natural Language Processing, Cambridge University Press, Cambridge, UK (1987).

[41] Hodges, J.C., Whitten, M.E., Brown, J., and Flick, J., *Harbrace College Handbook for Canadian Writers* , Fourth Edition, Harcourt Brace & Company, Canada, Toronto (1994).

[42] Hopcroft, J.E. and Ullman, J.D., *Formal Languages and Their Relation to Automata*, Addison Wesley, Reading, MA (1969).

[43] IEEE, *IEEE Recommended Practice for Software Requirements Specifications* , ANSI/IEEE Standard 830-1993, Institute of Electrical and Electronics Engineering, New York, NY (1993).

[44] Ishihara, Y., Seki, H., and Kasami, T., "A Translation Method from Natural Language Specifications into Formal Specifications Using Contextual Dependencies", pp. 232–239 in *Proceedings of the IEEE International Symposium on Requirements Engineering*, IEEE Computer Society Press, San Diego, CA (January 1993).

[45] Jackson, M., *Software Requirements & Specifications: A Lexicon of Practice, Principles, and Prejudices*, Addison-Wesley, London, UK (1995).

[46] Jackson, M.A., "The Role of Architecture in Requirements Engineering", pp. 241 in *Proceedings of the IEEE International Conference on Requirements Engineering*, IEEE Computer Society Press, Colorado Springs, CO (April 18–22 1994).

[47] Kamsties, E., Berry, D.M., and Paech, B., "Detecting Ambiguities in Requirements Documents Using Inspections", pp. 68–80 in *Proceedings of the First Workshop on Inspection in Software Engineering (WISE'01)*, ed. M. Lawford and D. L. Parnas, Software Quality Research Lab at McMaster University in Canada, Paris, France (23 July 2001).

[48] Kamsties, E., von Knethen, A., Philipps, J., and Schaetz, B., "An Empirical Investigation of the Defect Detection Capabilities of Requirements Specification Techniques", in *Proceedings of the Sixth CAiSE/IFIP8.1 International Workshop on Evaluation of Modelling Methods in Systems Analysis and Design (EMMSAD'01)*, Interlaken, CH (June 2001).

[49] Kamsties, E., "Surfacing Ambiguity in Natural Language Requirements", Ph.D. Dissertation, Fachbereich Informatik, Universität Kaiserslautern, Germany, also Volume 5 of Ph.D. Theses in Experimental Software Engineering, Fraunhofer IRB Verlag, Stuttgart, Germany (2001).

[50] Kaplan, G., Hadad, G., Oliveros, A., and Leite, J.C.S.P., "Construcción de Escenarios a partir del Léxico Extendido del Lenguaje", pp. 65–77 in *So'97FT'97 Simposio en Tecnológia de Software, 26 Jornadas Argentinas de Informática y Investigación Operativa - SADIO.*, Buenos Aires, Argentina (1997).

[51] Knuth, D.E., Larrabee, T.L., and Roberts, P.M., *Mathematical Writing* , MAA Notes, No. 14, Mathematical Association of America, Washington, D.C. (1989).

[52] Kovitz, B.L., *Practical Software Requirements: A Manual of Content and Style*, Manning, Greenwich, CT (1998).

[53] Leclercq, T. and Phillips, T.R., *Expert Legal Writing*, University of Texas, Austin, TX (1995).

[54] Leite, J.C.S.P. and Franco, A.P.M., "A Strategy for Conceptual Model Acquisition", pp. 243–246 in *Proceedings of the IEEE International Symposium on Requirements Engineering*, IEEE Computer Society Press, San Diego, CA (January 1993).

[55] Levinson, S., *Pragmatics*, Cambridge University Press, Cambridge, UK (1983).

[56] Lyons, J., *Semantics I and II*, Cambridge University Press, Cambridge, UK (1977).

[57] Mellinkoff, D., *The Language of the Law*, Little, Brown, and Co., Boston (1963).

[58] Mich, L. and Garigliano, R., "Ambiguity Measures in Requirements Engineering", in *Proceedings of the International Conference on Software Theory and Practice (ICS2000)*, Sixteenth IFIP World Computer Conference, Beijing, China (21–24 August 2000).

[59] Mich, L., Franch, M., and Novi Inverardi, P., "Requirements Analysis using Linguistic Tools: Results of an On-line Survey", to appear in *Requirements Engineering Journal* 2003 or 2004, Technical Report 66, Department of Computer and Management Sciences, Unversità di Trento, Trento, Italy (2003), http://eprints.biblio.unitn.it/view/department/informaticas.html.

[60] Mullery, G., "The Perfect Requirements Myth", *Requirements Engineering Journal* **1**(2), pp. 132–134 (1996).

[61] Naur, P., "Revised Report on the Algorithmic Language, ALGOL 60", *Communications of the ACM* **6**(1) (January 1963).

[62] Neumann, P.G., "Only His Only Grammarian Can Only Say Only What Only he Only Means", *ACM SIGSOFT Software Engineering Notes* **9**(1), pp. 6 (January 1986).

[63] Oriel, J., "Guide for Specification Writing for U.S. Government Engineers", Naval Air Warfare Center Training Systems Division (NAWCTSD), Orlando, FL (1999), http://www.ntsc.navy.mil/Resources/Library/Acqguide/spec.htm.

[64] Parnas, D.L., Asmis, G.J.K., and Madey, J., "Assessment of Safety-Critical Software in Nuclear Power Plants", *Nuclear Safety* **32**(2), pp. 189–198 (April–June 1991).

[65] Parnas, D.L., *personal communication via electronic mail*, November 2002.

[66] Reubenstein, H.B. and Waters, R.C., "The Requirements Apprentice: Automated Assistance for Requirements Acquisition", *IEEE Transactions on Software Engineering* **17**(3), pp. 226–240 (March 1991).

[67] Robertson, S. and Robertson, J., *Mastering the Requirements Process*, Addison-Wesley, Harlow, England (1999).

[68] Rolland, C. and Proix, C., "A Natural Language Approach for Requirements Engineering", pp. 257–277 in *Proceedings of Conference on Advanced Information Systems Engineering, CAiSE 1992*, Manchester, UK (12–15 May 1992).

[69] Rupp, C. and Goetz, R., "Linguistic Methods of Requirements-Engineering (NLP)", in *Proceedings of the European Software Process Improvement Conference (EuroSPI)*, Denmark (November 2000).

[70] Ryser, J., Berner, S., and Glinz, M., "SCENT Scenario Authoring Guidelines", Technical Report, University of Zuerich, Zuerich, Switzerland (1998).

[71] Schneider, G.M., Martin, J., and Tsai, W.T., "An Experimental Study of Fault Detection in User Requirements Documents", *ACM Transactions on Software Engineering and Methodology* **1**(2), pp. 188–204 (April 1992).

[72] Shertzer, M.D., *The Elements of Grammar*, Macmillan, New York, NY (1986).

[73] Singh, M.P. and Singh, M., "Deconstructing the 'Any' Key", *Communications of the ACM* **43**(4), pp. 107–108 (April 2000).

[74] Sommerville, I. and Sawyer, P., *Requirements Engineering, A Good Practice Guide*, John Wiley & Sons, Chichester, UK (1997).

[75] Sorensen, R., "Sharp Boundaries for Blobs", *Philosophical Studies* **91**(3), pp. 275–295 (September 1998).

[76] Strunk, W. and White, E.B., *The Elements of Style* , Third Edition, Macmillan, New York, NY (1979).

[77] van Leunen, M.-C., *A Handbook for Scholars*, Knopf, New York, NY (1978).

[78] van Wijngaarden, A., Mailloux, B.J., Peck, J.E.L., Koster, C.H.A, Sintzoff, M., Lindsey, C.H., Meertens, L.G.L.T., and Fisker, R.G., "Revised Report on the Algorithmic Language Algol 68", *Acta Informatica* **5**(1–3), pp. 1–236 (1975).

[79] Walton, D., *Fallacies Arising from Ambiguity* , Applied Logic Series, Kluwer Academic, Dordrecht, NL (1996).

[80] Wing, J.M., "A Study of 12 Specifications of the Library Problem", *IEEE Software* **5**(4), pp. 66–76 (July 1988).

[81] Zobel, J., *Writing for Computer Science*, Springer, Singapore (1998).