

# Ambiguity in Natural Language Requirements Documents

**Daniel M. Berry, CSCS & SE Program  
University of Waterloo**

# Outline of Talk

- **Natural Language is Key in RE**
- **Overview of Dangerous Constructions**
- **Overview of Tools**
- **Definitions and Examples of Ambiguity**
- **Why is Ambiguity Bad in RSs?**
- **Avoiding or Detecting Ambiguity**
- **Lessons Learned about Ambiguity in RE**
- **AI for RE?**
- **Requirements for Tools**
- **Conclusions**

# Natural Language is Key in RE

**Natural Language (NL) Requirements Specifications (RSs):**

**Overwhelming majority of RSs are written in NLS.**

**Virtually every initial conception for a system is written in NL.**

**Virtually every RFP is written in NL.**

# 1996 Estimates

**Osborne and MacNish reported John McDermid's estimates [Osborne1996] that**

- **about 90% of SRSs are written in solely NL**
- **less than 10% of SRSs are written in NL plus formal language**
- **less than 1% of SRSs are written in solely formal language**

# 2003 Data

**Data at [eprints.biblio.unitn.it](http://eprints.biblio.unitn.it) [Mich2003, Mich2004] show that a majority of documents available for requirements analysis are from users, either directly or by interviews, and of these documents**

- **71.8% are written in common NL,**
- **15.9% are written in structured NL, and**
- **only 5.3% are written in formalized language.**

# But...

**We all know that...**

**NL is *so* ambiguous, and so inherently so!**

**No wonder RSs are such messes!**

**There is that old tradeoff: RSs written in**

- **NLs or**
- **Mathematics-Based (MB) Formal Languages (FLs)**

# NL:

- **inherently ambiguous**
- + **always someone who can write it**
- + **always more or less understood by all stakeholders,  
albeit somewhat differently by each**

# MB FL:

- + inherently unambiguous**
- not always someone who can write it**
- not understood by most stakeholders, although all that do understand it understand it the same**

# Focus of Research

**A lot of research in RE is directed at solving the problem of ambiguous RSs by**

- **convincing people to use MB FLs and**
- **addressing the negatives of FLs**

...

# Addressing Negatives of FLs

by making FLs generally more accessible by

- **coming up with simpler FLs, e.g.**
  - **Parnas's [Henninger1978] tabular form**
- **building tools that take the drudgery out of using FLs, e.g.,**
  - **smart editors**
  - **theorem provers**
  - **model checkers**
  - **code generators**

# Addressing Negatives, Cont'd

- **building tools that help people understand FL specifications, e.g.,**
  - **simulators**

# Stark Reality

**But, the reality is that there is no escaping NL RSs.**

**Michael Jackson [Jackson1995] reminds us that**

**Requirements engineering is where the informal meets the formal.**

# Stark Reality, Cont'd

**Therefore, NLS are inevitable, even if it is only for the initial conception.**

**(Unless the client is some really weird math-type nerd who thinks in first-order predicate calculus, and how many of these are there?)**

# Stark Reality, Cont'd

**Even if one moves immediately to FLs, the inherent ambiguity of the NL initial conception can strike as the transition is made.**

**What the formalizer understands of the conception may be different from what the conceiver meant.**

**The phenomenon of subconscious disambiguation strikes! [Gause2000]**

# Subconscious Disambiguation

**In subconscious disambiguation (SD), the reader of an ambiguous phrase is not even aware that there is an interpretation other than the one that came first to his or her mind.**

**The reader understands an interpretation and thinks that it is the only one.**

# SD, Cont'd

In fact, *here* is where it's most important to catch ambiguity: right up front when the requirements analyst (RA) is getting raw information, be it goals, business rules, or requirements, from the clients and users.

The RA must find *each* ambiguity and ask the clients and users what they mean with it!

# SD, Cont'd

**Don Gause and Jerry Weinberg's original formulation of SD [Gause1989] was at a fairly high level:**

Create a means for protecting a small group of humans from the hostile elements of their environment.

**If the client was thinking of an igloo, but the RA immediately thought of a space station, then SD has struck again.**

# SD, Cont'd

**We now realize that SD can strike at low-level details,**

**e.g., The spam filter only marks the e-mail it considers to be spam.**

**What does it *really* mean?**

**More on the problems with only later!**

# Subconscious Ambiguation

**Subconscious ambiguation (SA) is the flip side of SD.**

**In SA, the writer of an ambiguous phrase is not even aware that he or she has written a phrase that has an interpretation other than what he or she thought to create the phrase.**

# Semi-Formal Languages?

**What about semi-formal languages like UML notations?**

**Double whammy**

**Ambiguity can still strike when go from conception to model.**

***And***

**The model itself is not unambiguous.**

# Research in NLPs in RE

**Therefore, there is a group of researchers focusing on solving the problem of ambiguous RSs by trying to improve our writing, understanding, and processing of NLPs.**

**There are several approaches to avoiding the ambiguity of NLPs:**

# **Problem-Avoiding Approaches:**

- 1. Learn to write less ambiguously.**
- 2. Learn to detect ambiguity.**
- 3. Use a restricted NL which is inherently unambiguous.**

**The first two reduce the disadvantage of existing NLs.**

**The third restricts the NLs to disallow the disadvantages.**

# Processing Restricted NL RSs

**By restricting the NL (but then it is not so natural!), it becomes possible to ascribe precise semantics to the sentences.**

# Restricted NL RSs, Cont'd

However, the questions always arise:

- **How expressible is the language? Have we lost something valuable in the restrictions?**
- **How convenient is it for humans to write the language?**
- **Is writing in such a language really different from writing in MB FLs?**
- **Is it really unambiguous? Suppose the language has features enabling structural ambiguities.**

**More later!**

# Avoiding or Detecting Ambiguity and Other Problems in NL RSs

**Some are focusing on the actual writing of the NL RS by the human writers, with an aim that they produce more complete, consistent, and less ambiguous NL RSs.**

**Some are focusing on improving inspections of NL RSs to more reliably detect incompleteness, inconsistencies and ambiguities.**

# Avoiding or Detecting Overview

**Rupp & Rolf Götz [Rupp1997], Berry & Kamsties [Berry2000, Berry2005], and Berry, Kamsties, & Krieger [Berry2003] identify dangerous constructs to avoid.**

**Kovitz [Kovitz1998] and Dupré [Dupré1998] explain how to write requirements and technical material less ambiguously.**

**Kamsties [Kamsties2001a, Kamsties2001b] explains how to inspect RSs for ambiguities.**

# Avoiding or Detecting, Cont'd

**Osborne & MacNish [Osborne1996],  
Wilson, Rosenberg, & Hyatt [Wilson1996,  
Wilson1997],  
Mich, Garigliano, *et al* [Mich2000, Mich2001,  
Kiyavitskaya2007],  
Bucchiarone & Gnesi [Bucchiarone2005,  
Berry2006b],  
and Tjong [Tjong2006b, Tjong2006a,  
Tjong2007] have built or are building tools that  
find instances of potential ambiguities.**

# Dangerous Constructions

**Christine Rupp and Rolf Götz detail common problems in NL RSs that lead to ambiguities, incompletenesses, and inconsistencies [Rupp1997].**

- **Writers can use this list to avoid problems.**
- **Inspectors can use this list as a checklist in inspections.**
- **RAs can use this list to find questions to ask of clients and users.**

# Dangerous “All”

**Erik Kamsties and I have sharpened one of these items, the recommendation that universal quantifier equivalents, e.g., all, never, etc. are often wrong [Berry2000].**

# Dangerous Plural

**Erik Kamsties and I have gone further and have determined that all and plural, in general, are hopelessly ambiguous ... [Berry2005],**

**so much so that I have nearly stopped using plural subjects in my writing.**

# Guide for Writing NL RSs

**There is at least one book out there on the writing of NL RSs, by Benjamin Kovitz [Kovitz1998].**

# Guide for Technical Writing

**There is a very good guide for technical writing, by Lynn Dupré [Dupré1998]**

# One Bit of Bad Advice

**One bit of advice common to a lot of writing style guides and subscribed to by every writing teacher I had,**

**Vary your words; use synonyms to avoid boring repetition!**

**is *bad* advice for RSs writing.**

# Bad Advice, Cont'd

**It creates what I call unnecessary synonymy (US) that leaves the reader wondering if there is an intended technical difference between two terms that generally mean the same thing, e.g.**

**data item and data element and datum.**

**I'd rather be bored by a RS than left wondering if these terms mean different things.**

**This is sort of the opposite of ambiguity!**

# Contracts, Laws, and RSs

**Legal contracts, laws, and software RSs are similar in that**

- **both are written in NLS and**
- **both have to anticipate all possible contingencies, i.e., exceptions.**

**They share the same kind of ambiguity problems.**

# Contracts and RSs, Cont'd

**A team was formed of two lawyers and two requirement engineers.**

- **Mickey Krieger, a lawyer who is a mathematician and computer scientist;**
- **David Kay, a lawyer who is a software engineer; and**
- **Kamsties and I, two software engineers who are requirements engineers.**

**Kay dropped out.**

# Still Other Documents

**The same ambiguity problems are shared by all kinds of documents with the same requirement of precision in specifying what is done and of covering all possible contingencies, e.g.:**

- **business goals**
- **business rules**
- **standard procedures**

# Writing Unambiguously

**This lawyers–requirement-engineers team has written a handbook on writing unambiguous NL legal contracts and software RSs [Berry2003].**

# Writing Unambiguously, Cont'd

**The handbook focuses on ambiguities stemming from**

- **incorrect placement of only and words like it, and**
- **incorrect uses of**
  - **grammatical number**
  - **logical connectives**
  - **universal and existential quantifier equivalents, etc.**

# Surfacing Ambiguity

**Kamsties has studied ambiguity in NL RSs and has come up with experimentally validated methods to detect them in inspections [Kamsties2001a, Kamsties2001b].**

**More details about this work later, after the definitions and examples.**

# Ambiguity Detecting Tools for RSs

**Several researchers have built or proposed building tools for detecting ambiguous sentences in RSs.**

- **Miles Osborne and Craig MacNish [Osborne1996] have built a tool that displays parses of any sentence in a controlled NL, so that the user can spot multiple or surprising parses and thus detect and eliminate ambiguities.**

# Tools, Cont'd

- **William Wilson, Linda Rosenberg, and Lawrence Hyatt [Wilson1996, Wilson1997] have developed ARM, a NASA-validated tool that finds in any English RS instances of specific indicators of ambiguities.**

# Tools, Cont'd

- **Luisa Mich, Roberto Garigliano, *et al* [Mich2000, Mich2001, Kiyavitskaya2007] have built tools based on the LOLITA NLP system for detecting syntactic (in CS sense) ambiguities in RSs.**

**LOLITA uses a parser implemented with lazy functional programming [Frost2006].**

# Tools, Cont'd

- **Antonio Bucchiarone and Stefania Gnesi [Bucchiarone2005, Berry2006b] have built QuARS, a RS-quality-model based and case-study validated tool that finds in any English RS instances of specific indicators of ambiguities.**

# Tools, Cont'd

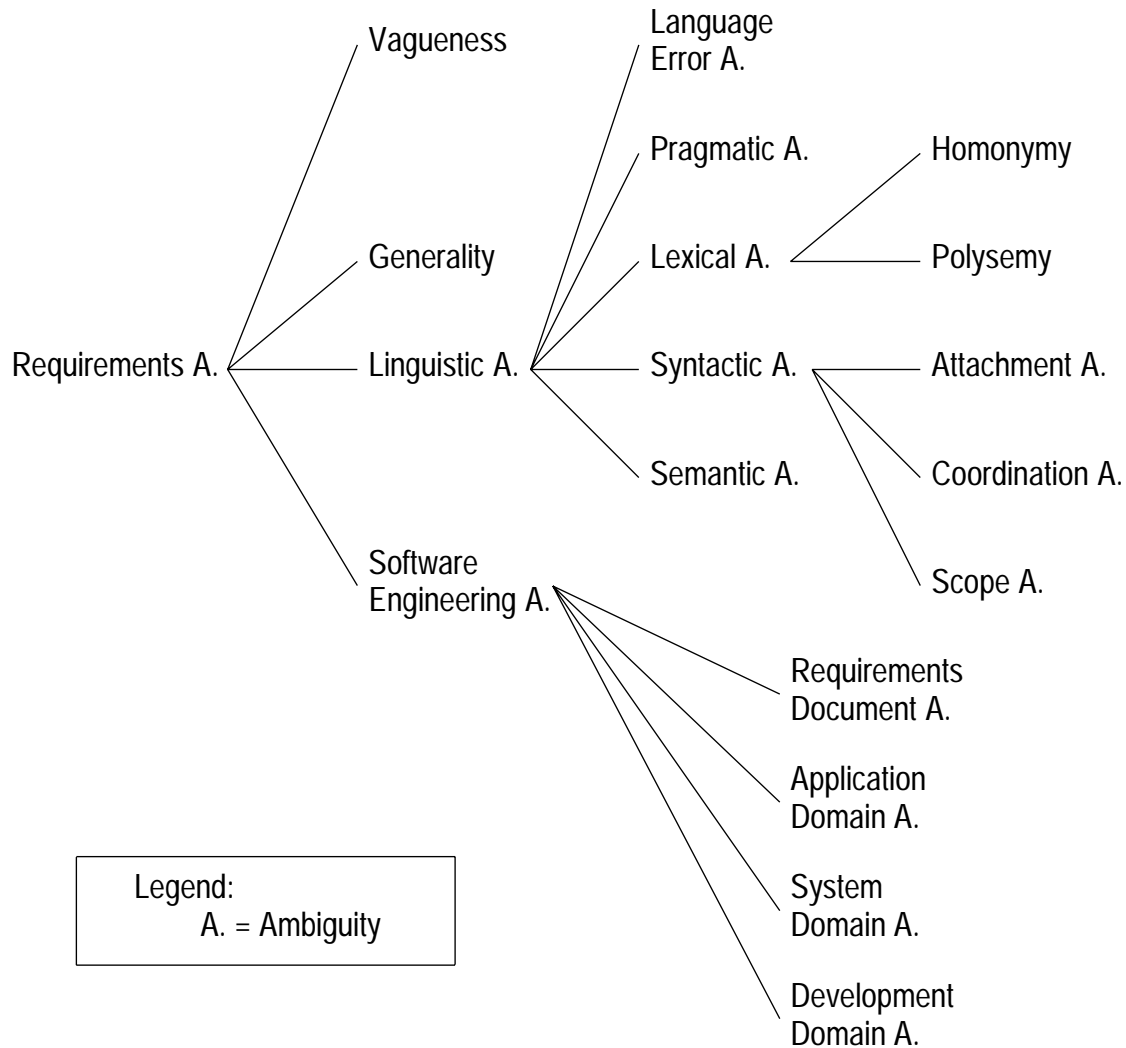
- **Sri Tjong [Tjong2006b, Tjong2006a, Tjong2007] is building a tool that finds in any English RS instances of specific indicators of ambiguities farmed from a corpus of industrial RSs.**

# Definitions and Examples

**Let's now focus on what *is* ambiguity in requirements specifications.**

**There are several sources of additional information [Berry2000, Berry2003, Berry2004, Berry2005].**

**First, a taxonomy of ambiguities in requirements specification**



# Definitions of Ambiguity

- **Dictionary**
- **Linguistic**
- **Software Engineering**

# Dictionary Definition

**The Merriam-Webster English Dictionary [Merriam-Webster1998] defines “ambiguity” as**

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

# Dictionary Definition, Cont'd

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.

# Lack of Ambiguity

**We need an easily pronounced term for “lack of ambiguity”, i.e., the property of being unambiguous.**

**I propose “uniguity” since “unambiguity” sounds yucky!**

**Also, “Uniguous” sounds better than “unambiguous”.**

# Multiguity

**Strictly speaking, the word that means “capable of being understood in two or more possible senses or ways” should be “multiguous”.**

**But, something which is multiguous is also ambiguous.**

# In a Defining Mood

In place of “etc.” = “et cetera”, we sometimes need a word that means “or others” to end an “or”-separated list.

I propose:

“velc.” = “vel cetera”, for inclusive-“or”-separated lists, and

“autc.” = “aut cetera”, for exclusive-“or”-separated lists.

# Linguistic Definitions

**Linguistic ambiguity is investigated in linguistics [Lyons1977] and in related fields, namely, computational linguistics [Hirst1987, Allen1995] and philosophy [Levinson1983, Walton1996].**

**I give examples of these shortly, but first, let's dispense of the third definition.**

# Software Engineering Definition

**There is no single comprehensive definition of ambiguity in SE literature.**

**Each gives some key aspects and misses others.**

**Most are operational.**

**I give only two here, but there are others [Schneider1992, Gause1989].**

# IEEE Definition

**The IEEE Recommended Practice for SRSs [IEEE1993] 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 this definition is that there is no such thing as an unambiguous specification.**

**There are useful [Parnas2002] specifications!**

# Davis's Definition

**Alan Davis [Davis1993] has suggested a test for ambiguity to serve as a definition:**

**“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.”**

# Davis's Definition, Cont'd

**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, which is unambiguous for most practical purposes.**

# Linguistic Ambiguities

- **Lexical Ambiguity**
- **Syntactic Ambiguity**
- **Semantic Ambiguity**
- **Pragmatic Ambiguity**
- **Generality & Vagueness**
- **Language Error (NEW!)**

# Lexical Ambiguity

***Lexical ambiguity* occurs when a word has several meanings.**

**There are several kinds:**

- **homonymy,**
- **polysemy, and**
- **systematic polysemy**

# Lexical Ambiguity, Cont'd

***Homonymy* occurs when two different words have the same written and phonetic representation, but unrelated meanings and different etymologies, e.g.,**

bank

***Polysemy* occurs when a word has several related meanings but one etymology, e.g.,**

drunk, green

# Lexical Ambiguity, Cont'd

***Systematic polysemy*** occurs when the reason for the polysemy is confusion between classes, e.g., between unit and type, e.g.,

I like this jacket.

and between process and product, e.g.,

I like writing.

# 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.**

**There are several kinds, including:**

- **attachment ambiguity and**
- **coordination ambiguity.**

# Syntactic Ambiguity, Cont'd

***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, e.g.,**

The police shot the rioters with guns.

# Syntactic Ambiguity, Cont'd

***Coordination ambiguity* occurs**

- **when more than one conjunction, and or or, is used in a sentence, e.g.,**

I saw Peter and Paul and Mary saw me.

- **or when one conjunction is used with a modifier,**

young man and woman.

# 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.**

**These include**

- 1. coordination ambiguity (see above)**
- 2. referential ambiguity (e.g, of pronouns, also is pragmatic ambiguity), and**
- 3. scope ambiguity, e.g.,**

**All linguists prefer a theory.**

# Pragmatic Ambiguity

***Pragmatic ambiguity* occurs when a sentence has several meanings in the context in which it is uttered, e.g.,**

Every student thinks she is a genius.

# Generality & Vagueness

**Cousin is general w.r.t. gender in English. So,  
Sue is visiting her cousin.**

**is general.**

**fast has no clear boundary between fast and  
not fast. So,**

**fast response time**

**is vague.**

# Language Error

**Our experience has identified another category of ambiguity, *language error*.**

**As with all other categories, language error may not be mutually exclusive of other categories.**

# Language Error, Cont'd

**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.**

# Examples

**The most common language errors are:**

- **only**
- **all and plural**
- **pronouns**

**They are certainly the most difficult to detect if you are not aware of the problem.**

**For a complete discussion of these errors and others, see our handbook [Berry2003]**

# **Dangerously Misplaced “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.**

**The typical native English speaker puts only always before the main verb of its sentence, no matter what is limited by only.**

# Misplaced “Only”, Cont’d

**E.g., if it is desired to say that the only e-mail that a spam filter delivers to the user is e-mail that the user wants, one properly says:**

**The spam filter delivers only the e-mail that the user wants.**

# Misplaced “Only”, Cont’d

**Many a native English speaker says instead:**

The spam filter only delivers the e-mail that the user wants.

**The meaning of this alternative sentence is that the only thing the spam filter does to the e-mail that the user wants is to deliver it; it does not forward, eat, modify, or anything else the e-mail that the user wants.**

# Misplaced “Only”, Cont’d

**It does *not* promise to deliver only the e-mail that the user wants.**

**This incorrect sentence is understood by most native English speakers as it is probably meant to be understood, because what the sentence really means does not make much sense.**

**While the error can be made in other natural languages, in practice, speakers of other languages make the mistake far less often.**

# Misplaced “Only”, Cont’d

**However, there are sentences of this form in which what the sentence really means is as meaningful as what it probably means, and the careful reader is left wondering what the writer really means.**

# Misplaced “Only”, Cont’d

**E.g.,**

It only illustrates the concepts.

**To most native English speakers, the sentence means:**

**It illustrates only the concepts and not reasons for them.**

# Misplaced “Only”, Cont’d

**But, it really means:**

**It only illustrates the concepts and does not define them.**

**The correct sentence for the first is:**

**It illustrates only the concepts.**

# Misplaced “Only”, Cont’d

**Another example:**

I only nap after lunch.

**To most native English speakers, the sentence means:**

**The only time I nap is after lunch.**

# Misplaced “Only”, Cont’d

**But, it really means:**

**The only thing I do after lunch is nap.**

**The correct sentence for the first meaning is:**

**I nap only after lunch.**

# Misplaced “Only”, Cont’d

**Another example:**

The spam filter only marks the e-mail it considers to be spam.

**Each of (1) the correct meaning, (2) what most native English speakers think it means, and (3) yet another meaning is a reasonable utterance about spam filters:**

# Misplaced “Only”, Cont’d

- 1. The spam filter only marks, and does not delete, the e-mail it considers to be spam.**
- 2. The spam filter marks only the e-mail it considers to be spam.**
- 3. The spam filter only marks only the e-mail it considers to be spam.**

# Misplaced “Only”, Cont’d

**Peter Neumann, in his short elegant essay on only, “Only His Only Grammarian Can Only Say Only What Only he Only Means” [Neumann1984] gives 15 variations of the sentence,**

I said he thought secret users may write secret data.

**by migrating only through different places in the sentence and inserting commas and that.**

There’s an “only” missing in the title.

# Misplaced “Only”, Cont’d

**Each of the 15 variations has a different meaning, which can be relevant in the security domain.**

**e.g.,**

**Only, I said he thought secret users may write secret data.**

**Only I said he thought secret users may write secret data.**

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

I said only he thought secret users may write secret data.

I said he only thought secret users may write secret data.

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

I said he thought only secret users may write secret data.

I said he thought only secret users may write secret data.

I said he thought secret users only may write secret data.

I said he thought secret users only may write secret data.

I said he thought secret users may only write secret data.

I said he thought secret users may write only secret data.

I said he thought secret users may write secret data only.

# A California Proposition

Only marriage between a man and a woman is valid in California.

**instead of the intended:**

**The only marriage that is valid in California is that between a man and a woman.**

# California Proposition, Cont'd

**Hmmm, so are all relationships between a man and a woman other than marriage, e.g. friendship, dating, engagement, sex, separation, divorce, parental, grandparental, sibling, etc., not valid in California?**

# Misplaced “Only”, Cont’d

**There are other words that have the same problem as only. Among these words are almost, also, even, hardly, just, merely, mostly, nearly, and really.**

**In other words, it is common to misplace also also, not only only. (Thanks go Jo Atlee for the pun effects.)**

# Other Languages

**These syntactic problems with only are not restricted to English.**

# Other Languages, Cont'd

**Each of the above examples using only can be duplicated with the same meanings**

- **in French with seulement or ne ... que,**
- **in German with nur,**
- **in Hebrew with רק,**
- **in Italian with soltanto,**
- **in Portuguese with somente, and**
- **in Spanish with solamente,**

**respectively.**

# Syntactically Dangerous “All”

**Consider the sentence,**

All the lights in any room have a single on-off switch.

**The question to be asked is “How many switches does any room have, one or one per light?”**

# Dangerous “All”, Cont’d

**The problem with this sentence is that it is not clear whether**

- 1. each light in any room has its own single on-off switch that isn’t shared with any other light, or**
- 2. all lights in any room share a common single on-off switch.**

**The sentence is ambiguous.**

# Even a Third Meaning

**There is yet another, more obscure, meaning, that ...**

- 3. each light in any room has its own single on-off switch that may be shared with another light.**

# Dangerous “All”, Cont’d

**If one writes**

Each light in any room has a single on-off switch.

**or**

Each light in any room has its own on-off switch.

**then the first meaning is clearly intended.**

# Dangerous “All”, Cont’d

**If he writes**

All lights in any room share a single on-off switch.

**then the second meaning is clearly intended.**

# Dangerous “All”, Cont’d

## The ambiguous sentence

All the lights in any room have a single on-off switch.

**is a classic example of scope ambiguity; it is not clear which quantifier equivalent, all, for “ $\forall$ ”, or a, for “ $\exists!$ ” (there exists a unique), takes precedence over the other.**

# Dangerous “All”, Cont’d

**Mathematics shows the problem clearly. The two meanings are:**

1.  $\forall y \in$  the lights in a room,  $\exists! x$  such that  $x$  is the on-off switch of  $y$
2.  $\exists! x$  such that  $\forall y \in$  the lights in a room,  $x$  is the on-off switch of  $y$

# Surprise and Embarrassment!

**Erik and I had looked over the contents last 6 slides many times. I had explained the last 6 slides to others many times, and ...**

**we never noticed a lurking ambiguity.**

**Finally, when we published a paper [Berry2005] including this information, Bob McCloskey at U. of Scranton, PA, wrote us a letter [Berry2006a] pointing out that ...**

# Embarrassment, Cont'd

**(1)** Each light in any room has a single on-off switch.

**and**

**(2)** Each light in any room has its own on-off switch.

**are different!**

# Embarrassment, Cont'd

**Bob pointed out that:**

**(1) Each light in any room has a single on-off switch.**

**means only that each light has a single on-off switch and says nothing about how many lights each switch controls *and***

**allows the interpretation that all lights in any room share a single common on-off switch.**

# Embarrassment, Cont'd

**To mean**

**(2) Each light in any room has its own on-off switch.,**

**(1) can be changed to**

**Each light in any room has a single on-off switch in the room, and each on-off switch in any room controls a single light in the room.**

**(2) avoids the problem by using the word own.**

# Lessons Learned

## Hard lessons learned:

- **humility** 😊 **and**
- **ambiguities are hard to catch *even* when you are looking for them and you are attuned to them.**

# Dangerous “All”, Cont’d

**Many times the same ambiguity is hidden by domain knowledge.**

**E.g., consider ambiguous sentence (that is structurally similar to the “lights” sentence),**

**All persons have a unique national insurance number.**

**There is another, semantic danger in the sentence: it ain’t true,**

**Therefore, the software should not depend on it!**

# Dangerous “All”, Cont’d

**Domain knowledge tells the reader that the intended meaning of the sentence is that**

- **each person has his or her own unique national insurance number,**

**and not the ridiculous idea that**

- **all persons share a common unique national insurance number.**

# Dangerous “All”, Cont’d

**The second option is so ridiculous that most readers of the sentence would not even think that there *is* another option and that the sentence is ambiguous.**

# Syntactically Dangerous Plural

**Closely related to the syntactically dangerous all, is the syntactically dangerous plural.**

**The use of plural to describe a property of elements of a set or of sets makes it difficult to determine whether the property is that of each element or of the whole set.**

# Dangerous Plural, Cont'd

**Consider the two structurally identical sentences:**

Students enroll in six courses per term.

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.**

# Dangerous Plural, Cont'd

**Without this domain knowledge, there is nothing in either sentence to indicate whether enrollment in the stated number of courses per term is a property of each student or of the set of all students.**

# Dangerous Plural, Cont'd

**The first sentence is talking about each student; it should be written in singular form:**

**Each student enrolls in six courses per term.**

# Dangerous Plural, Cont'd

**Using a singular formulation for talking about properties of each or any student reserves the plural formulation:**

Students enroll in hundreds of courses per term.

**for talking about properties of the collection of students.**

# Dangerous Plural, Cont'd

**Alternatively, you could insist on singular even for sets, by introducing some set equivalent to hold the elements of the set:**

The student body enrolls in hundreds of courses per term.

# Dangerous Plural, Cont'd

**The same syntactic problem exists with other, non-universal quantifier equivalents, e.g., some, many, which are all plural.**

# Even in Math or Tech Writing

**Plural ambiguity is particularly problematic in mathematical or technical writing, although there are occasionally nearby formulae that disambiguate.**

**E.g., Systems contain subsystems.**

**Is containment one–one, one–many, many–one, or many–many?**

# Math or Tech Writing, Cont'd

**Domain knowledge tells us what makes sense in this case, ...**

**but if you are trying to learn new mathematics with a sentence like this, then what?**

# Formal Treatment of Plural

**We discovered that Uta Schwertel had published a Ph.D. thesis thoroughly researching the semantics of plural for the purpose of extending ACE to include plural [Schwertel2003].**

**Her example illustrating the inherent ambiguity of plural is:**

Three girls lift a table.

# Schwertel's Thesis

**The thesis explored the semantics of plural using a computational proof-theoretic approach.**

**ACE had outlawed plural because it was hopelessly ambiguous [Schwertel2000].**

# Schwertel's Thesis, Cont'd

**The thesis was a study to see if this restriction could be unambiguously relaxed.**

**She thinks, “Yes!”.**

**I think, “No!”.**

**We agreed to differ. 😊**

# **Guilt and My Writing**

**I began to feel guilty about using plural in my own writing.**

**Could I in good conscience write any sentence with an ambiguous use of plural?**

**These sentences amount to almost every sentence with plural.**

**Would I be able to sleep nights?**

# Guilt, Cont'd

**I finally decided to banish sentences with the plural ambiguity from my writing, using plural *only* when I am talking about a property of the set of objects denoted by a plural construct, or rarely, when ambiguity is totally innocuous [Chantree2005].**

# Banishing Plural

**Implementing this decision meant finding singular equivalents for many, some, few, etc.**

...

**e.g. in Many people like to eat their lunch with a cold drink.**

**or Many people like to eat their lunches with cold drinks.**

# Singular Substitutes

**Like each is a singular substitute for all.**

**For many, I use many a or the typical.**

**For some, I use the occasional.**

**For few, I use the rare.**

**e.g. Many a person likes to eat his or her lunch with a cold drink.**

**and *definitely not* Many a person likes to eat their lunch with a cold drink.**

# My Recent Writing

**For the past 3 years I have been careful in my writing to avoid ambiguous plural and other problems mentioned in these slides, especially misplaced onlys and alsos.**

# My Recent Writing, Cont'd

**Sometimes the wording is strange.**

**My coauthors notice it and complain, but eventually accept it.**

**An occasional reviewer complains that we have English errors all over or that we need to have an English editor clean up the paper.**

**The typical reviewer nevertheless says that the paper is well-written and clear, even when occasionally rejecting it!**

# My Recent Writing, Cont'd

**I have had trouble with the typical copy editor who rewrites sentences into plural and moves the onlys and alsos to their normal sounding places, *even in an article about the plural problem or about ambiguities, including that of misplaced onlys and alsos.***

# Other Languages

**These syntactic problems with plural universal quantifier equivalents and with plural sentences are not restricted to English.**

# Other Languages, Cont'd

**Each of the above examples using all or each can be duplicated with the same meanings**

- **in French with tous or chaque,**
- **in German with alles or jeder,**
- **in Hebrew with col or col ekhad,**
- **in Italian with tutti or ogni, and**
- **in Portuguese and Spanish with todo or cada,**

**respectively.**

# Dangerous Plural, Cont'd

**Mathematics has adopted a convention that makes intent very clear.**

**In mathematics, the universal quantifier  $\forall$ , read as for all is singular as in,**

$$\forall x \in \text{Int}, x < x+1$$

**For all Integers  $x$ ,  $x$  is less than  $x+1$**

# Dangerous Pronouns

**With each pronoun, to which noun it refers is problematic, e.g.,**

Every student thinks she is a genius.

**One must be careful in writing to make sure that the referent of a pronoun is what is intended.**

**When one is writing text, she has no difficulty understanding a pronoun's referent.**

# **Dangerous Pronouns, Cont'd**

**However, the poor reader must often guess.**

**The grammatical rules say that the referent of a pronoun must be the previous noun or the non-pronoun sentence subject.**

**This rule alone is ambiguous.**

**Moreover, sometimes the writer does not follow this rule in her thinking.**

# Dangerous Pronouns, Cont'd

**The best defense is to use nouns instead of pronouns, but that can sound funny.**

**Another good defense is to introduce formal names:**

Consider the switch  $s1$ . ...  $s1$  is turned off.

# Dangerous “This”

**The most insidious problem is that of This. (I capitalize it because it usually comes at the beginning of a sentence.)**

**The writer says, e.g.:**

**This prevents security breaches.**

# Dangerous “This”, Cont’d

**To what does This refer?**

**to the previous noun?**

**to the previous sentence subject?**

**to the idea of the previous sentence?**

**to the idea of the previous  $n$  sentences?**

**to the idea of the current paragraph?**

**to the idea of the previous paragraph?**

# Dangerous “This”, Cont’d

**I have seen all these possibilities, and ...**

**I have seen situations in which more than one of these makes sense.**

# Dangerous “This”, Cont’d

**The defense:**

**Always follow this by a noun that restricts the referent**

**e.g,**

**This encoding scheme prevents security breaches.**

# Other Languages

**These problems with This are not restricted to English.**

**Certainly, each language other than English has pronouns, which can have uncertain referents.**

# Other Languages, Cont'd

**Moreover, each of the above examples using This can be duplicated with the same meanings**

- **in French with Ceci,**
- **in German with Dieses,**
- **in Hebrew with Zehu,**
- **in Italian with Ciò,**
- **in Portuguese with Isto, and**
- **in Spanish with Esto,**

**respectively.**

# Ambiguity, Details

- **Why is Ambiguity Bad in RSs?**
- **Why Ambiguity is Tough to Find?**
- **Not All Ambiguity is Linguistic**
- **Poor Writing Causes Ambiguities**
- **Lexicons and Unnecessary Synonymy**
- **Avoiding or Detecting Ambiguities with Inspection Drivers**

# Why is Ambiguity Bad in RSs?

**Ambiguity in requirements specification is dangerous because stakeholders (customers, users, developers) can disagree on the meaning of a requirements specification without being aware of the disagreement.**

**This disagreement can result in disastrous failures as the software is not prepared to deal with expected situations or deals with them in unexpected ways.**

# Why Ambiguity Bad?, Cont'd

**On the other hand, ambiguity can be quite acceptable provided that contextual information is available, because humans are naturally skilled in resolving ambiguities, sometimes without even knowing it.**

**Often disambiguation is subconscious and tacit, as the reader of an ambiguous phrase is not even aware that there is an interpretation other than the one that came first to his or her mind.**

# Why Ambiguity is Tough to Find?

**Many ambiguities are not noticed because of SD mentioned earlier.**

**The reader understands an interpretation and thinks that it is the only one.**

**This understanding is why an inspection checklist with one ambiguity-relevant item, “Is document ambiguous?” is ludicrous.**

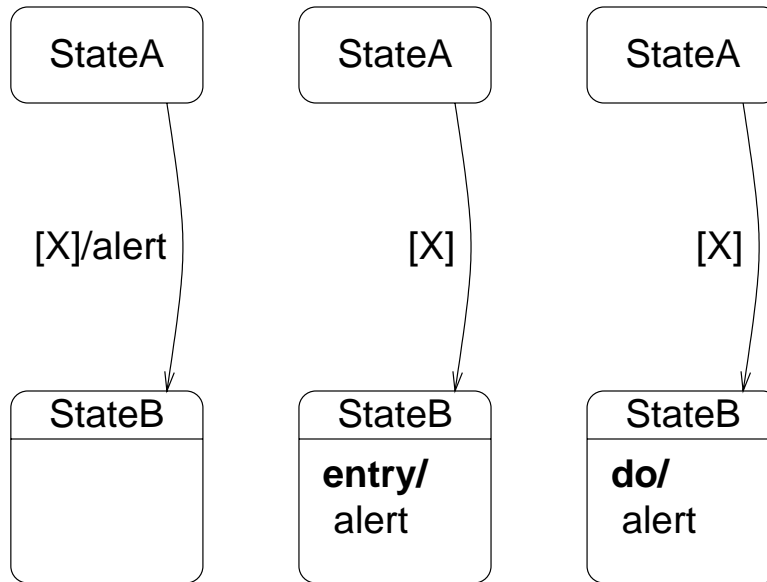
# Not All Ambiguity is Linguistic

**Some ambiguities are due specifically to computing and do not appear ambiguous from the linguistic point of view, e.g.**

**If X happens then raise an alert.**

**is clear to the reader (assuming that X is clear), but gives rise to three different UML State Machine models, each with slightly different semantics.**

# Not All Ambiguity, Cont'd



**At implementation and run times, which one is meant can be critical.**

# Poor Writing Causes Ambiguities

**People write appallingly poorly and ambiguously when it comes to words like**

- **only**
- **all**
- **each**
- **not | no | none**
- **pronouns**
- **plural**

**We have seen some examples.**

# Poor Writing, Cont'd

**These people include even lawyers and requirements engineers who should know better, because their documents have a heavy requirement for lack of ambiguity!**

**Even more appalling is the fact that neither law degree nor SE degree programs teach people how to write clearly.**

# Poor Writing, Cont'd

**Each program focuses on using its field's technology to solve problems to which poor writing is a major contributor.**

**It's like teaching SE without teaching RE, because it's in poor RE that about 80% of the errors are introduced.**

# Knowing How to Write Well →

**Knowing how to write well is important for avoiding ambiguities as well as for every other benefit of good writing.**

**Moreover, knowing where the pitfalls are makes it easier to *detect* ambiguities in both polished SRSs and in initial client-supplied documents about a system the client desires.**

# Knowing, Cont'd

**The most important documents in which to detect ambiguities are these initial client-supplied documents about a system the client desires.**

**Each potential ambiguity should give rise to a question asked of the client.**

# Unnecessary Synonymy

**Unnecessary Synonymy is the use of several different terms to mean the same concept, data or process, within one RS.**

**The synonymy is *unnecessary* because use of one term suffices.**

# Lexicons and US

**Julio Leite *et al* remind us that it is necessary to build a lexicon of the domain-specific terms used in any RS [Leite2000, Cysneiros2001].**

**This lexicon not only helps to avoid US, but also supports the elicitation of requirements and the construction of scenarios.**

# Lexicons and US, Cont'd

**In Leite's Language Extended Lexicon (LEL), the terms are defined in terms of each other.**

**Even though the LEL's definition graph is cyclic, the LEL serves the purpose of avoiding US in domain-specific terms.**

# Avoiding other US

**LA Scheinholtz addresses the problem of unnecessary synonymy (US) [Scheinholtz2006].**

**We tend to be pretty good about avoiding US in domain-specific terms by the use of acronyms, glossaries, and lexicons, but we are positively sloppy when it comes to non-domain-specific terms.**

# Avoiding other US, Cont'd

**Scheinoltz describes an iterative process for identifying US among non-domain terms in a collection of RSs, a procedure which quickly drills down to one stakeholder-agreed-upon term for each concept, creating a lexicon of terms and phrases to be used throughout RSs.**

# Avoiding or Detecting Ambiguity

**We are reducing the disadvantages of existing NLs by**

- **helping the specification writer to avoid ambiguities, and**
- **helping the specification reviewer to detect ambiguities.**

# Surfacing Ambiguities

**Erik Kamsties recently completed his PhD dissertation, *Surfacing Ambiguity in Natural Language Requirements* [Kamsties2001b].**

**Kamsties observes that ambiguity is more complex than is often recognized in RE literature.**

# Surfacing Ambiguities, Cont'd

**The emphasis of his work is on reducing ambiguity during modeling and analysis of the requirements, instead of only trying to identify them later only after the requirements are specified.**

# Surfacing Ambiguities, Cont'd

**Kamsties suggests techniques, based on industrially proven inspection and modeling techniques, that enable a requirements engineer to spot dangerous ambiguities, i.e., ones that are misinterpreted.**

- 1. Checklists**
- 2. Modeling**
- 3. Scenario-Based Reading**

# Checklists

**The core of the checklist approach is a comprehensive set of definitions of both kinds of ambiguity that occur in NL RSs:**

- **linguistic (syntactic and semantic) ambiguity and**
- **conceptual (requirements and computing relevant) ambiguity.**

# Checklists, Cont'd

**The definition of linguistic ambiguity was borrowed from linguistics; nu?.**

**These definitions of ambiguity can be used as a checklist for inspections of NL RSs.**

# Modeling

**He developed a procedure for deriving ambiguity-surfacing guidelines from an arbitrary modeling technique to be applied during application of the technique.**

# Modeling, Cont'd

**He used this procedure to develop ambiguity surfacing guidelines that are tailored to the modeling techniques,**

- **for SCR (Software Cost Reduction) and**
- **for UML (Unified Modeling Language)**

# Modeling, Cont'd

**The guidelines are normally applied during the modeling of requirements.**

**These guidelines are adaptable for use also in inspections.**

# Scenario-Based Reading

**The scenario-based reading techniques, based on ideas of Victor Basili [Basili1997]**

**provides the inspector with an operational scenario (of something to do to the specification) that requires him or her to create an abstraction of the requirements and then to answer questions based on an analysis of the abstraction, e.g.**

# Scenario-Based Reading, Cont'd

- **abstraction = test cases**

**question = “Do you have all the information needed to develop a test case?”**

**and**

- **abstraction = matrix of interacting requirements**

**question = “Is there only one relation between requirements A and B?”**

# Experimental Validation

**He experimentally validated and compared the effectiveness of the techniques in controlled experiments at Fraunhofer IESE with SE students and professionals on problems of industrial size.**

# Experimental Validation, Cont'd

**These techniques led to**

- **reduced numbers of ambiguities misinterpreted during requirements modeling and**
- **higher numbers of detected ambiguities in inspections.**

**over previous unfocused techniques.**

# Experimental Validation, Cont'd

Moreover, he was able to rank the techniques according to effectiveness.

1. **Scenario-Based Reading**
2. **Modeling**
3. **Checklists**

# Lessons Learned about Ambiguity in RE

- **Restricted NLS**
- **AI for RE?**
- **Too Much Automation is Bad for RE**
- **RS Text vs. Ordinary Text**
- **Tools for Dealing with Ambiguity**

# Restricted NLS

**There are several restricted NLS out there for writing RSs:**

**Comer's early effort [Comer1983]**

**TELL [Enomoto1984b]**

**Attempto Controlled English (ACE)  
[Fuchs1999]**

# James Comer's Early Work

**James Comer described an interactive system converting descriptions of data structures given in a restricted subset of English into algebraic axiomatic data type specifications [Comer1983].**

**Comer's work was followed by other efforts, e.g., ...**

# TELL

**Hajime Enomoto *et al* developed TELL.**

**TELL translates specifications written in restricted NL, NSL, into formulae of modal logic [Enomoto1984b].**

**TELL is used to specify a network protocol that is then verified to be the alternating bit protocol [Enomoto1984a]**

# Attempto

**Norbert Fuchs *et al* have developed Attempto, a restricted English and a tool for understanding it [Fuchs1999].**

**It has been used for a variety of medium-sized examples and as the input language for a theorem prover.**

# Restricted NLs, Cont'd

**Assume that a restricted NL is expressive, usable, and readable enough.**

**The question remains:**

**Is it truly unambiguous?**

# Restricted NLS, Cont'd

**It probably avoids lexical and syntactic ambiguities.**

**However, if it uses only, all, plural, and pronouns, it can still have ambiguities, of the kind I have shown in these slides.**

# ACE and Ambiguity

**As a matter of fact, ...**

**ACE disallows plural.**

**ACE has no only.**

**ACE has very strict rules about pronoun reference which occasionally determines an unnatural referent.**

**ACE has very strict rules about the precedence of conjunctions and the use of commas to control these precedences.**

# Unrestricted NL in Analysis and Design

**Russ Abbott described the use of unrestricted natural language as part of an object-oriented requirements analysis and program design method [Abbott1983].**

**This approach became a key element of Grady Booch's approach to object-oriented program design [Booch1991].**

# Extracting Information from Grammatical Structure

**Motoshi Saeki, Hisayuki Horai, *et al* consider what semantics they can learn from the syntax structures of unrestricted NL RSs [Saeki1987].**

# Grammatical Structure, Cont'd

**Miles Osborne and Craig MacNish**

**[Osborne1996] have built a tool that displays parses of any sentence in a controlled NL, so that the user can spot multiple or surprising parses and thus detect and eliminate ambiguities.**

# Grammatical Structure, Cont'd

**Colette Rolland and Christophe Proix explored the relationship between grammatical cases of words in NL RSs and ER models and use what they learned to build a tool, OICSI [Rolland1992].**

**These and other similar approaches are summarized in a survey [Denger2001].**

# Model Building from Unrestricted NL RSs

**As mentioned above, Rolland and Croix built a tool that builds ER models from the grammatical case tagging given to nouns in a NL RS [Rolland1992]**

**Julio Leite and Ana Paula Franco considered a technique for deriving a conceptual model from a lexicon, which is in turn derived from NL RSs [Leite1993].**

# Model Building, Cont'd

**Yasunori Ishihara *et al* developed a tool that translates NL specifications of network protocols into algebraic specifications [Ishihara1993].**

**Stephane Somé *et al* developed a tool for elicitation of scenarios written in NL and then to derive complete RSs from them [Somé1996].**

# Gervasi's Work

**Most recently, Vincenzo Gervasi has developed CIRCE, a complete environment for interactive parsing and understanding NL RSs and for deriving a number of models of the specified system [Gervasi2000a, Ambriola2000].**

**He and Bashar Nuseibeh have applied CIRCE to an industrial strength example, i.e., the International Space Station's Node Control System [Gervasi2000b].**

# Popescu's Approach

**Daniel Popescu's idea is that the difference between a model generated automatically from a NL SRS and what the client expects is a sign of ambiguity that should be explored with the client.**

**He built a tool called Dowser for generating an object-oriented analysis model from a NL SRS [Popescu2006, Popescu2007].**

# Popescu's Approach, Cont'd

**However, the input language is not unrestricted NL.**

**The user must rewrite a SRS into a moderately restricted NL.**

**One could argue that this rewriting itself is beneficial to the understanding of the SRS by the rewriter.**

# Everyone Wants Tools

**One thing about computer scientists:**

**Given any problem in SE or RE, we want to build a tool that solves the problem or at least helps solve it!**

**Given any process in SE or RE, we want to build a tool that does the process or at least helps carry it out!**

**And the ambiguity-in-RE area is no different!**

**But, is it such a good idea?**

# AI for RE?

**Kevin Ryan wondered if AI approaches to NL understanding had any chance of working for RE [Ryan1993].**

**His conclusion was basically, “No!”.**

# Kevin Ryan's Observations on AI in RE

**Back in 1993, Kevin Ryan put the role of NLP in RE into perspective [Ryan1993].**

**He noted the importance of NL to RE, that indeed RE is firmly based in NL, as spoken and written by humans.**

**He observed:**

# Kevin Ryan's Observations, Cont'd

**The history of natural language processing (NLP), in relation to the specification of systems and programs has been bedevilled with many unrealistic suppositions and presumptions. Given the critical and expensive nature of the current approaches to requirements engineering (RE) the prospect of a support system that would automatically understand a user's needs is, naturally, very appealing. Numerous research projects have proposed to derive and validate system requirements knowledge by means of a natural or "near natural" conversation with the prospective client [e.g. *citations*].**

# Kevin Ryan's Observations, Cont'd

**This facility, it is fondly believed, is both feasible and desirable and would make specification of systems both easier and more accurate. Unfortunately this belief is incorrect on both counts. In this brief paper, I wish to assert that natural language processing does not now, nor will it in the foreseeable [sic] future, provide a level of understanding that could be relied upon, and even if it could, it is highly questionable that the resulting system would be of great use in requirements engineering.**

# Kevin Ryan's Observations, Cont'd

**I would add that such a system would not even be desirable, because it would take the thinking requirements analyst out of the loop, making it less likely that he or she would notice serious omissions and questionable, albeit logically okay, requirements.**

**While full understanding is out of the realm of possibility or desires, there are ways that NLP tools can help the practicing, thinking requirements analyst.**

# Kevin Ryan's Observations, Cont'd

**Such tools can help scan, search, browse, and tag early textual descriptions of the requirements to assist in the analysis and social processes that are necessary to produce full and accurate requirements specification.**

**Such tools can help maintain fully traced requirement specifications throughout the full lifecycle of the required system.**

# AI for RE, Cont'd

**Artificial intelligence approaches to language understanding do not work well enough for RE work.**

**The mistakes are annoying.**

**Too much good stuff is missed.**

# AI for RE, Cont'd

**We prefer naturally dumb clericality (פקידותיות) (picky-duty-ut) to artificial intelligence.**

**Real stupidity is preferable to artificial intelligence if the artificial intelligence can lose information as a result of it's not ever being perfect.**

# **Too Much Automation is Bad for RE**

**I regard with suspicion any tool that is so automatic that it takes the human requirement engineer out of the loop.**

**The most powerful RE tool ever is the good ol' human brain.**

**Tools that require human intervention bring the human into the loop and promote thinking about the results.**

# Too Much Automation, Cont'd

**It is in this thinking that omissions and questionable, albeit logically okay, requirements are noticed.**

**Tools that run by themselves to produce requirements take the human out of the loop and make it less likely that a human will think about the results.**

**This lack of thinking is very, very dangerous.**

# RS Text vs. Ordinary Text

**The NL used in RSs is quite different from the NL used in ordinary, garden variety or even highly literary text.**

**RS text is highly explicit, over a restricted domain, using a reduced vocabulary, with well-defined terms.**

**The other kind of text is much more elliptical, over arbitrary domains, using an unlimited vocabulary, with fuzzily defined terms.**

# Tools

- **grep vs. Eyes**
- **Requirements for Tools**
- **Tools for Ambiguity Identification**

# grep vs. Eyes

***grep* is a lot better at finding all instances of a word or a pattern than is the human eye.**

# Requirements for Tools

**The total amount of information to deal with for any real problem is HUGE and repetititive.**

**We desire assistance in extracting useful information from this mass of information.**

# Requirements for Tools, Cont'd

**We would like the extracted information to be**

- **summarizing (got less stuff),**
- **meaningful (precision) (got only good stuff), and**
- **covering (recall) (got all good stuff).**

**From 500 pages, we want 5 pages containing *all* and *only* the meaningful information in the 500 pages.**

# Requirements for Tools, Cont'd

**We prefer less summarization and occasional meaningless stuff than to lose any meaningful stuff, because in any case, a human will have to read the output and at that time can filter out the meaningless stuff.**

**I.e.,**

- **100% recall,**
- **not too bad precision, and**
- **good summarization.**

# Imprecision a Boon?

**Actually, a little nonburdensome imprecision may help keep the human RA engaged, ...**

**especially if the instances of imprecision are funny examples of the stupidity of computers and algorithms! 😊**

# Information Retrieval (IR)

**Particularly indexing, keyword identification, and thesaurus generation**

**Mostly batch processing on large collections of documents, e.g., for Library of Congress**

**The work goes way, way back, but is summarized by [Salton1989, Frakes1992, Srinivasan1992].**

# Differences between IR and RE

**Both IR and RE deal with extracting small pieces of information from large texts.**

**However, there are key differences between IR and RE that impact the requirements for tools used for their extraction processes.**

# Differences, Cont'd

**IR tends to work with continually growing large collections of unchanging texts.**

**RE works with small collections of rapidly changing texts.**

**Thus, rapid, off-line, batch processing is essential for IR.**

**Thus, slower, interactive, on-line processing is acceptable for RE.**

# Differences, Cont'd

**The tools for IR cannot make mistakes. They must be summarizing, meaningful (precise), and covering (recalling) without any human intervention.**

**The tools for RE can afford to be less than meaningful (precise) because a human being is watching the output and can filter meaningless stuff, provided that there is not too much of it.**

# Differences, Cont'd

**In RE, you may err on the side of meaninglessness (imprecision) provided that you lose no coverage (recall).**

**In RE, it doesn't matter if it takes 3 hours to extract desired information; 3 hours is meaningless in the lifecycle; you can take the client out for lunch during the 3 hours.**

# Differences, Cont'd

**These differences affect requirements and implementation choices for tools for IR and RE.**

**We cannot use standard IR tools for extraction in RE without rethinking their applicability.**

# Ambiguity Identification Tools

**Some are trying to use language understanding tools to find ambiguities [Mich2000, Mich2001, Kiyavitskaya2007].**

**These tools certainly do find ambiguities; in fact, ambiguities in NLPs are the tools' biggest enemies.**

# Identification Tools, Cont'd

**They certainly will find all**

- **lexical ambiguities, by use of dictionaries**
- **syntactic ambiguities, by attempting to parse**

**But will these be useful?**

**I am not so sure.**

# Identification Tools, Cont'd

**In one case, the tool found dozens of ambiguities that were NOT ambiguities to me [Kiyavitskaya2007].**

**My understanding of the context disambiguated *all* that it did find.**

**However, what's worse, the tool did not find *any* of the only, all, plural, and referential ambiguities.**

# Identification Tools, Cont'd

**So maybe dumb tools [Wilson1996, Wilson1997, Bucchiarone2005, Tjong2006b, Tjong2006a] that find all occurrences of only, all, and other problematic words might be more helpful?**

**Can we build a tool that finds possible referents of pronouns accurately enough to be helpful?**

# Main Tool Requirements

**Suppose the ambiguity  $A$  is one for which I *must* find every instance, to be sure that there are no lurking problems.**

**Suppose the ambiguity  $A$  is one for which I expect that I could find every instance manually, except for the fact, that I get tired or sloppy and overlook instances.**

# Then, I Want!

**Then, I want a tool that searches instances of  $A$  to have 100% recall, even at the the expense of imprecision and poor summarization.**

**If I *even feel* that the tool is not finding every instance of  $A$  in a document  $D$ , then I cannot trust the tool, and I have to read the whole of  $D$  myself.**

# I want, Cont'd

**I don't even want to *see* the output of the tool before I have looked at *D* myself, because I am afraid that the output will lull me into complacency.**

# But I Am Not Perfect

**Of course, I am not perfect myself, so maybe a less-than-perfect-recall tool would be useful to complement me, but not to replace me.**

**My lack of perfection is in fatigue and complacency, not in recall or precision.**

# 100% Recall by Design

**Testing a tool's recall is problematic because you need perfect output against which to compare the tool's output.**

**So, I mean 100% recall by *design* rather than by testing.**

**For example, suppose *A* is “misplaced only”.**

# Parser-Based Tool

**One might say:**

**Hmm..., an only that is not immediately preceding the main verb of a sentence is probably not misplaced; since the only was put in a non-conventional place, it is probably in the right place.**

# Parser-Based Tool, Cont'd

**So, maybe I should build a smart tool, *TP*, based on a parser that tags each word with its part of speech so that it can check whether any *only* it finds is immediately preceding the main verb of the containing sentence.**

# grep-Based Tool

**No! No parser and no tagger has 100% recall.  
So, any *TP* would have less than 100% recall.**

**Instead, build a dumb tool, *TG*, based on *grep* that simply displays each sentence (any string of words ending in a . ; ; ? or !) that contains only.**

# Smart vs. Dumb

***TG* has 100% recall, ...**

**but it is less precise and less summarizing than *TP*; in this case, lack of summarization is *caused* by lack of precision.**

# Smart vs. Dumb, Cont'd

**On the other hand, the imprecision is manageable:**

- **I can easily spot the OK sentences.**
- **Probably, the OK sentences are statistically rare, because the typical English writer misplaces his or her onlys; so there will not be much imprecision.**

# Missing Semantic Ambiguity

**All this syntax-based work misses semantic ambiguity (using “syntax” and “semantic” in the CS sense).**

**Daniel Popescu’s work addresses semantic ambiguities [Popescu2006].**

# Conclusions

**NL is unavoidable in RSs, even if only at the very beginning when you are talking with the client.**

**SA strikes in writing.**

**SD strikes in reading.**

# Conclusions, Cont'd

**Ambiguity abounds in places you never even thought of, e.g., in only, in all, and in plural.**

**Any tool must have 100% recall and good summarization at the expense of some imprecision.**

# Most Important Lesson

**In reality, we are never going to prevent ambiguity.**

**So we must learn to spot it, not only in polished RSs, ...**

**but also, and especially, *in goals, business rules, and INITIAL RSs, in whose reading SD first strikes!***

**AND ASK THE CLIENT WHAT HE OR SHE MEANS!**