The Dangerous "All" in Specifications

Daniel M. Berry
Computer Science Department
University of Waterloo
Waterloo, Ontario N2L 3G1
Canada
dberry@csg.uwaterloo.ca

Erik Kamsties
Fraunhofer Institute for
Experimental Software Engineering
Kaiserslautern, D-67661
Germany
kamsties@iese.fhg.de

Abstract

Rupp and Götz observe that some, but not all, requirement specification sentences involving universal quantification, are dangerous because they are usually not true. Jackson and Zave provide a classification of requirement specification sentences into indicative and optative sentences. It is observed that the dangerous sentences involving universal quantifiers are all indicative.

Keywords: universal quantifier, indicative sentence, optative sentence, dangerous, true, false.

Dangerous Sentences

Christine Rupp and Rolf Götz, in "Sprachliche Methoden des Requirements Engineering" (Linguistic Methods in Requirements Engineering) caution specifiers of the dangers of using the words "never", "always", "none", "each", "all", and other universal quantifier equivalents in natural language specifications [5]. They point out that such a statement is sometimes dangerous because it may simply not be true and for a computer-based system to assume that it is true is courting disaster when an unanticipated input comes along and the system is not prepared to respond to it gracefully. For example, one might specify, "Each person has a unique national insurance (Social Security in the U.S.) number."* This statement is, to use the vernacular, mostly true and is thus logically false, since there are persons who for one reason or another have gotten more than one number. For a computer-based system dealing with national insurance to assume that each person has precisely one number is downright dangerous. The system must in fact deal with all sorts of anomalies, including,

1. that a given person has more than one number,
2. that a given person has never been assigned a number,
3. that a given person reports an invalid number, and
4. that a given person reports someone else's number.

There may be other anomalies that we have not listed here.

A similar case can be made for the danger many statements involving other universal quantifier words such as "never", "always", "none", and "all".

However, there are times in which such strong universally quantified statements are appropriate. For example, a robust procedure in a program should be able to handle all inputs, even if the mathematical function it implements is undefined for some inputs; in these undefined cases, the procedure should at least report that the input is illegal.

Indicative and Optative Moods

The question to ask is, "when are universally quantified statements dangerous and when are they not?" We believe that notions offered by Michael Jackson and Pamela Zave provide the distinction [3, 4]. Jackson and Zave talk about *descriptions* of *domains*, or *real worlds* and *requirements*, or *problems*. "The domain is the subject matter of the system's computations, and provides the context in which those computations have useful meaning or effect." [3] They consider a domain "as a topic for description in its own right, independently of any description that we may eventually make of the system to be constructed." Jackson and Zave divide sentences in a specification into two classes, those that describe the domain and those that describe requirements.

1. A sentence about the domain is grammatically in the *indicative* mood; it asserts truths about the domain.

* Most likely, one would say, "All persons have a unique national insurance number", but that is not correct for reasons beyond the scope of this note [1].

Proceedings of the Tenth International Workshop on Software Specification and Design (IWSSD'00)
0-7695-0884-7/00 $10.00 © 2000 IEEE

That is, it describes the world as it is, independent of any computation that may be placed in it.

2. A sentence about the requirements is grammatically in the *optative* mood; it describes what the computation being specified is required to bring about. That is, it describes the world as it will be after the computation is placed in it.

To be concrete, the sentence "Each person has a unique national insurance number." is an attempt to be an indicative statement, about the real world. Unfortunately it is incorrect, but it clearly does not depend on any computation that we might wish to impose on the real world. A correct indicative statement would be "Except for exceptions described elsewhere, each person has a unique national insurance number." The sentence "The national insurance system shall deal with each input that is claimed to be a national insurance number." is an example of an optative statement, about a system, a computation, to be built in the real world. With this distinction, it is clear when universally quantified statements are dangerous and when they are not.

Moods and Danger

A universally quantified indicative statement is dangerous because it probably is not true, and assuming that it is true leaves the program unable to deal with all possible inputs. Moreover, such statements lull the system designers into not investigating all possible contingencies. A requirement engineer who believes the customer's claim that "Each person has a unique national insurance number." is less likely to investigate all the possibilities and is less likely to discover the four exceptions to the rule that are mentioned above and with which the system must deal.

There are universally quantified indicative statements that are true, for example, "Each human is mortal." However, such statements are rare. In general, each universally quantified indicative statement has to be examined closely to search for exceptions or to ascertain that it is indeed true.

On the other hand, a universally quantified optative statement is reasonable and often desired. It is reasonable to require that the national insurance system deal with each input claiming to be a national insurance number. The system should be able to handle the four exceptions mentioned above as well as the normal case in which the number belongs to one and only one person. The system should also be able to handle any situation that has not been thought of and described in the specifications.

Example

The description of the case study for this workshop [2] provides a classic example of a dangerous universally quantified sentence. Its description of the Teleservices and Remote Medical Care System (TRMCS) is divided, as would be done by Jackson and Zave, into a Domain Theory section and a System Requirements section. The Domain Theory section is filled with indicative statements and the System Requirements section is filled with optative statements. Interestingly there is only one universally quantified sentence in the whole description and it is in the Domain Theory section. The sentence is indicative and should, thus, by our argument, be dangerous. The sentence is shown in bold face in its context, shown in regular face.

> The following two scenarios illustrate these new types of services. Both assume some prior events where a patient/user has seen a physician and has been approved to receive at-home assistance and that **the help center has prior medical information stored about each user registered with it.**

It assumes, at the very least, that each patient or user (1) has seen a physician and (2) has had his or her medical history gathered. It assumes moreover, that this medical history (3) is accurate, (4) is up to date, and (5) has been stored in the system's database. Additional thinking should show some other assumptions as well. The description says that the patient or user has only *seen* a physician and been approved for the service. It says nothing about the physician having gathered accurate medical history from the patient or user. The description also says that the help center has only prior medical history only stored for only each registered user. It says nothing about this medical history being taken, up to date, and accurate. The description says nothing about what to do if the information for a patient or user is out of date or inaccurate, and most importantly, what to do if it gets a call from a patient or user with no stored medical history. It also fails to consider what is to be done with a medical history of a non-registered patient or user. These issues must be considered in completing the specification.

To be fair, the description is not intended to be a complete requirements document and is intended to promote the kind of thinking implicit in the previous paragraph. However, the danger of the cited sentence is clear.

Proceedings of the Tenth International Workshop on Software Specification and Design (IWSSD'00)
0-7695-0884-7/00 $10.00 © 2000 IEEE

Conclusion

In conclusion, a specification consists of two kinds of sentences, indicative and optative. A universally quantified indicative statement is probably not true and should thus raise a red flag. It should be a signal to the requirement engineers to ask when it might not be true, to allow discovery of all the exceptions that must be handled. Having universally quantified optative statements is a laudable goal for all (note the universal quantifier in this essentially optative statement) computer-based systems, as it indicates the goal that each system handle both its normal cases and all possible exceptions and contingencies. A universally quantified optative statement should be yet another signal to the requirement engineers to search for other contingencies that the system should handle.

Acknowledgments

The authors thank the anonymous referees, Jo Atlee, Don Cowan, and Michael Jackson for their comments on earlier drafts of this paper.

References

[1] Berry, D.M., Kamsties, E., and Krieger, M.M., "From Contract Drafting to Software Specification: Linguistic Sources of Ambiguity," Technical Report, University of Waterloo, Waterloo, ON, Canada, 2000.

[2] Inverardi, P., Muccini, H., and Richardson, D., *Tenth International Workshop on Software Specification and Design (IWSSD-10) Case Study: The Teleservices and Remote Medical Care System (TRMCS)*, 2000, http://www.ics.uci.edu/iwssd/case_study.pdf.

[3] Jackson, M. and Zave, P., "Domain Descriptions," pp. 56–64 in *Proceedings of the International Symposium on Requirements Engineering*, IEEE Computer Society, 1993.

[4] Jackson, M., *Software Requirements & Specifications: A Lexicon of Practice, Principles, and Prejudices,* Addison-Wesley, London, 1995.

[5] Rupp, C. and Götz, R., "Sprachliche Methoden des Requirements Engineering (NLP)," in *PROFT # CONQUEST-1, First Conference on Quality Engineering in Software Technology*, Nürnberg, Germany, 25–26 September 1997.

Proceedings of the Tenth International Workshop on Software Specification and Design (IWSSD'00)
0-7695-0884-7/00 $10.00 © 2000 IEEE