

✤ a point of reference for product maintainers > must be updated when the product is updated ✤ baseline for change requests

→ A legal contract

 \succ must be possible to determine whether the specification was met

> must be updated whenever changes are negotiated

© 2001, Steve Easterbrook

Management

view?

Readers?

already allocated

resources

Primary: Spec author;

Secondary: Customer

estimate resource needs

and plan the development

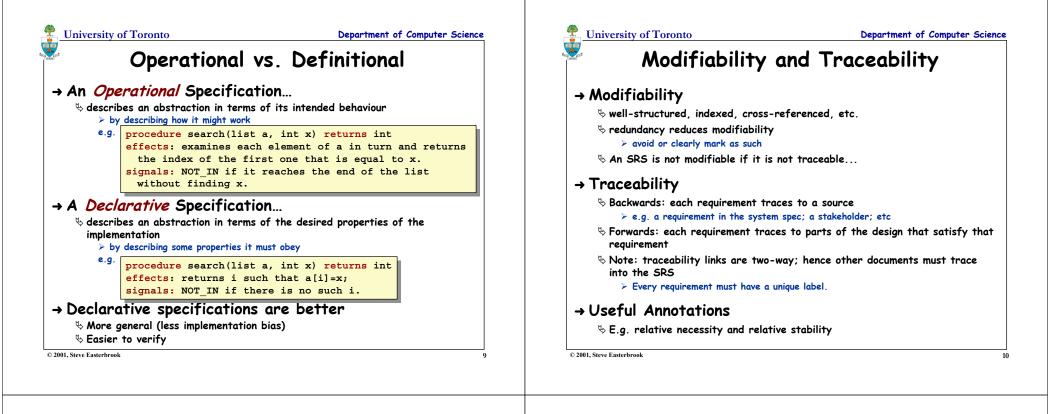
Primary: all programmers

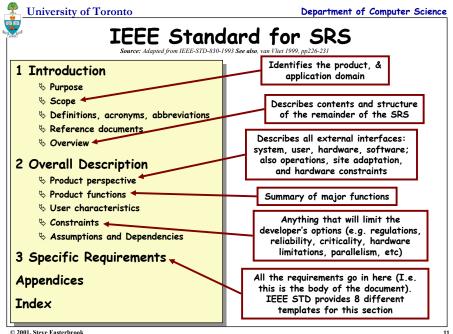
+ V&V team; Secondary:

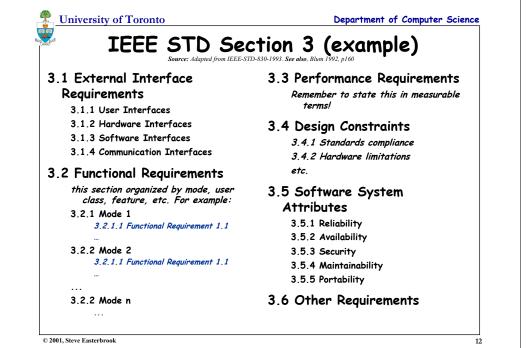
managers, customers

University of Toronto University of Toronto Department of Computer Science Department of Computer Science Desiderata for Specifications Source: Adapted from the IEEE-STD-830-1993. See also van Vliet 1999, pp225-226 Restrictiveness vs. Generality → Valid (or "correct") → Specificand Sets → Unambiguous ✤ expresses actual requirements le every statement can be read in A specification describes a set of acceptable behaviours exactly one way \rightarrow Complete The set of all implementations that meet a specification are its specificand b define confusing terms in a glossary ♦ Specifies all the things the system set must do → Verifiable ♦ There are always an infinite number of possible implementations ✤ ...and all the thinas it must not do! ✤ a process exists to test satisfaction This specification is ∜ E.g.∣ ✤ Responses to all classes of input procedure foo(x: int) returns y:int of each requirement trivial, but its ♦ Structural completeness, and no effects: x = y♦ "every requirement is specified specificand set is TBDs!! behaviorally" still infinite! → Restrictiveness: → Consistent → Understandable (Clear) ♦ doesn't contradict itself (i.e. is a specification should rule out any implementation that is unacceptable to its satisfiable) ♦ by non-computer specialists users ♦ Uses all terms consistently → Modifiable ♦ Note: timing and logic are especially \rightarrow Generality: prone to inconsistency & Carefully organized, with minimal a specification should be general enough so that few of the acceptable redundancy \rightarrow Necessary implementations are excluded. Traceable! ♦ doesn't contain anything that isn't > In particular the more desirable (e.g. elegant, efficient) implementations should `required' not be excluded. > Examine every condition in the spec and ask if it's really needed © 2001, Steve Easterbrook © 2001 Steve Easterbrook University of Toronto Department of Computer Science University of Toronto Department of Computer Science **Consistency and Completeness** Ambiguity

 \rightarrow Is this ambiguous? \rightarrow Consistency: "The system shall report to the operator all faults that originate in satisfied critical functions or that occur during execution of a critical sequence and for which there is no fault recovery response." inconsistency may depend on context: The text should be kept in lines of equal length specified by the user. Spaces should be inserted between words to keep the line lengths equal. \rightarrow Can test by trying to translate it: A line break should only occur at the end of a word Originate in critical functions F т F т F т ✤ In practice, inconsistency is hard to test for. Occur during critical segeunce F F Т т F F Т т \rightarrow Completeness F F F F т Т т No fault recovery response т ♦ internally complete: ? ? ? ? ? ? ? Report to operator? > all terms are defined > no TBDs Scomplete with respect to the requirements the if you get different answers from different people, then it is ambiguous. > i.e. describes all services needed by the users ♦ In practice, completeness is nearly impossible to achieve > aim for balance between generality and restrictiveness © 2001. Steve Easterbrook © 2001, Steve Easterbrook







University of Toronto

Department of Computer Science

13

References

van Vliet, H. "Software Engineering: Principles and Practice (2nd Edition)" Wiley, 1999.

Section 9.2 covers most of the material in this lecture, and gives a good introduction to the IEEE standards.

IEEE-STD-830-1993

Is the current IEEE standard that covers software specifications. It is available electronically through the IEEE electronic library (access via U of T library website for the campus-wide subscription)

Blum, B. "Software Engineering: A Holistic View". Oxford University Press, 1992

Provides some additional insights into how to write good specifications.

© 2001, Steve Easterbrook