Writing Better Requirements

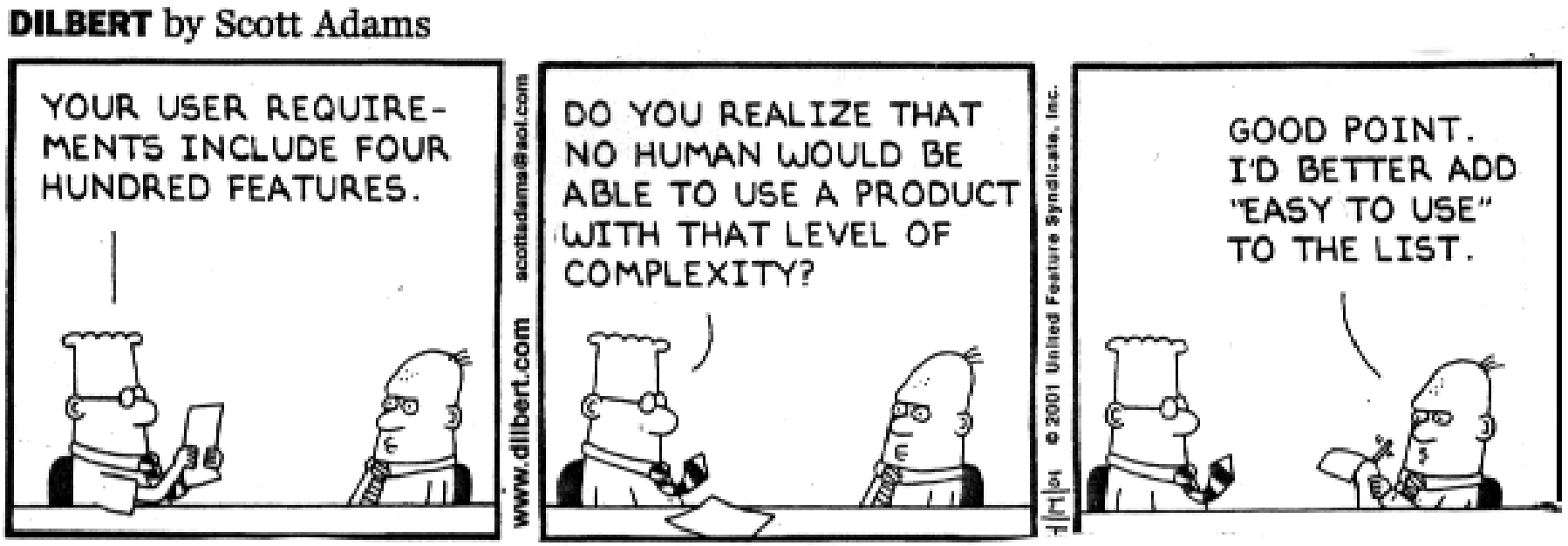
Gregor v. Bochmann, University of Ottawa

Based on Powerpoint slides by Gunter Mussbacher
with material from:
Ian Zimmerman (Telelogic, 2001),
Ivy Hooks (Compliance Automation, 1998)

Table of Contents

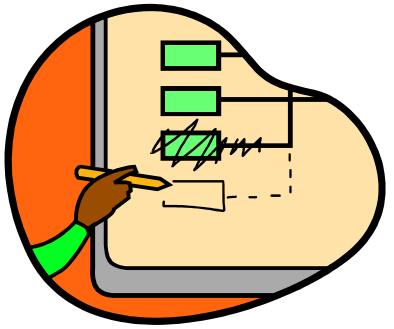
- Martha can't write requirements because…
- Anatomy of a Good / Bad User Requirement
- Standard for Writing a Requirement
- Writing Pitfalls to Avoid
- A Few Simple Tests…
- The greatest challenge to any thinker is stating the problem in a way that will allow a solution.[1]

[1] Bertrand Russell, 1872-1970


DILBERT by Scott Adams
YOUR USER REQUIREMENTS INCLUDE FOUR HUNDRED FEATURES.
DO YOU REALIZE THAT NO HUMAN WOULD BE ABLE TO USE A PRODUCT WITH THAT LEVEL OF COMPLEXITY?
GOOD POINT. I'D BETTER ADD "EASY TO USE" TO THE LIST.
www.dilbert.com scottadams@aol.com
© 2001 United Feature Syndicate, Inc.

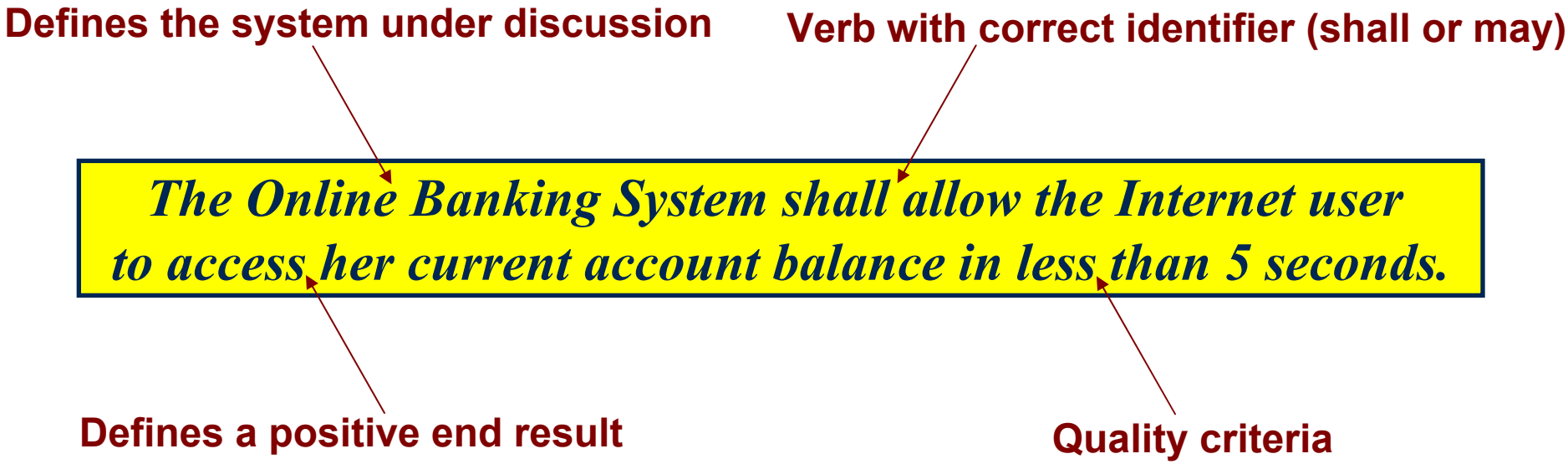
Martha can't write requirements because…

- She doesn't know what to do!
  - She was not taught at school
  - She doesn't know how to write
  - She doesn't understand the process
  - She doesn't have the necessary data
  - She doesn't know what she wants
- She doesn't understand why!
  - She doesn't understand the impact / changes
  - She thinks this is "just a document"
- She'd rather do something else!
  - She'd rather design – she sees no reward
  - She doesn't have enough time
  - She thinks the review process will catch the errors



Source: Compliance Automation, Inc., 1998

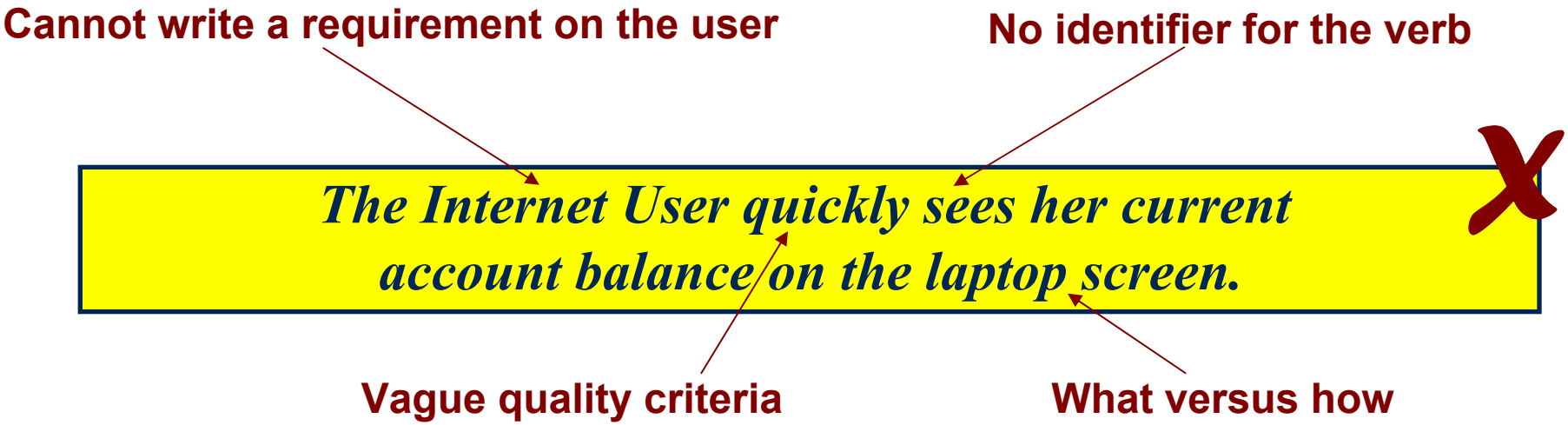
Anatomy of a Good User Requirement



- Identifies the system under discussion and a desired end result that is wanted within a specified time that is measurable

- The challenge is to seek out the system under discussion, end result, and success measure in every requirement

Example of a Bad User Requirement



Standard for Writing a Requirement

- Each requirement must first form a complete sentence
  - Not a bullet list of buzzwords, list of acronyms, or sound bites on a slide
- Each requirement contains a subject and predicate
  - Subject: a user type (watch out!) or the system under discussion
  - Predicate: a condition, action, or intended result
  - Verb in predicate: "shall" / "will" / "must" to show mandatory nature; "may" / "should" to show optionality
- The whole requirement provides the specifics of a desired end goal or result
- Contains a success criterion or other measurable indication of the quality

Standard for Writing a Requirement

- Look for the following characteristics in each requirement
  - Feasible (not wishful thinking)
  - Needed (provides the specifics of a desired end goal or result)
  - Testable (contains a success criterion/other measurable indication of quality)
  - Clear, unambiguous, precise, one thought
  - Prioritized
  - ID

- Note: several characteristics are mandatory (answer a need, verifiable, satisfiable) whereas others improve communication

Writing Pitfalls to Avoid

- Never describe how the system is going to achieve something (over-specification), always describe what the system is supposed to do
  - Refrain from designing the system
    - Danger signs: using names of components, materials, software objects, fields & records in the user or system requirements
  - Designing the system too early may possibly increase system costs
  - Do no mix different kinds of requirements (e.g., requirements for users, system, and how the system should be designed, tested, or installed)
  - Do not mix different requirements levels (e.g., the system and subsystems)
    - Danger signs: high level requirements mixed in with database design, software terms, or very technical terms

Writing Pitfalls to Avoid

- "What versus how" test

The system shall use Microsoft Outlook to send an email to the customer with the purchase confirmation. ✗

The system shall inform the customer that the purchase is confirmed.

Writing Pitfalls to Avoid

- Never build in let-out or escape clauses
  - Requirements with let-outs or escapes are dangerous because of problems that arise in testing
  - Danger signs: if, but, when, except, unless, although
    - These terms may however be useful when the description of a general case with exceptions is much clearer and complete that an enumeration of specific cases
- Avoid ambiguity
  - Write as clearly and explicitly as possible
  - Ambiguities can be caused by:
    - The word or to create a compound requirement
    - Poor definition (giving only examples or special cases)
    - The words etc, …and so on (imprecise definition)

Writing Pitfalls to Avoid

- Do not use vague indefinable terms
  - Many words used informally to indicate quality are too vague to be verified
  - Danger signs: user-friendly, highly versatile, flexible, to the maximum extent, approximately, as much as possible, minimal impact

The Easy Entry System shall be easy to use and require a minimum of training except for the professional mode. ✗

Writing Pitfalls to Avoid

- Do not make multiple requirements
  - Keep each requirement as a single sentence
  - Conjunctions (words that join sentences together) are danger signs: and, or, with, also
- Do not ramble
  - Long sentences with arcane language
  - References to unreachable documents

The Easy Entry Navigator module shall consist of order entry and communications, order processing, result processing, and reporting. The Order Entry module shall be integrated with the Organization Intranet System and results are stored in the group's electronic customer record.

Writing Pitfalls to Avoid

- Do not speculate
  - There is no room for "wish lists" – general terms about things that somebody probably wants
  - Danger signs: vague subject type and generalization words such as usually, generally, often, normally, typically
- Do not express suggestions or possibilities
  - Suggestions that are not explicitly stated as requirements are invariably ignored by developers
  - Danger signs: may, might, should, ought, could, perhaps, probably
- Avoid wishful thinking
  - Wishful thinking means asking for the impossible (e.g., 100% reliable, safe, handle all failures, fully upgradeable, run on all platforms)

The Easy Entry System may be fully adaptable to all situations and often require no reconfiguration by the user.

A Few Simple Tests…(1)

- "What versus how" test discussed earlier
  - Example: a requirement may specify an ordinary differential equation that must be solved, but it should not mention that a fourth order Runge-Kutta method should be employed
- "What is ruled out" test
  - Does the requirement actually make a decision (if no alternatives are ruled out, then no decision has really been made)
  - Example: a requirement may be already covered by a more general one

Source: Spencer Smith, McMaster U.

A Few Simple Tests…(2)

- "Negation" test
  - If the negation of a requirement represents a position that someone might argue for, then the original decision is likely to be meaningful

The software shall be reliable. ✗

- "Find the test" test

The car shall have an engine. ✗

  - The requirement is problematic if no test can be found or the requirement can be tested with a test that does not make sense
  - Test: look, here it is!

Source: Spencer Smith, McMaster U.

Rate these Requirements

The Order Entry system provides for quick, user-friendly and efficient entry and processing of all orders.

Invoices, acknowledgments, and shipping notices shall be automatically faxed during the night, so customers can get them first thing in the morning.

Changing report layouts, invoices, labels, and form letters shall be accomplished.

The system shall be upgraded in one whack.

The system has a goal that as much of the IS data as possible be pulled directly from the T&M estimate.

Towards Good Requirements Specifications (1)

- Valid (or "correct")
  - Expresses actual requirements
- Complete
  - Specifies all the things the system must do (including contingencies)
  - ...and all the things it must not do!
  - Conceptual Completeness (e.g., responses to all classes of input)
  - Structural Completeness (e.g., no TBDs!!!)
- Consistent
  - Doesn't contradict itself (satisfiable)
  - Uses all terms consistently
  - Note: inconsistency can be hard to detect, especially in concurrency/timing aspects and condition logic
  - Formal modeling can help
- Beneficial
  - Has benefits that outweigh the costs of development

Source: Adapted from Blum 1992, pp164-5 and the IEEE-STD-830-1993

Towards Good Requirements Specifications (2)

- Necessary
  - Doesn't contain anything that isn't "required"
- Unambiguous
  - Every statement can be read in exactly one way
  - Clearly defines confusing terms (e.g., in a glossary)
- Uniquely identifiable
  - For traceability and version control
- Verifiable
  - A process exists to test satisfaction of each requirement
  - "every requirement is specified behaviorally"
- Understandable (clear)
  - E.g., by non-computer specialists
- Modifiable
  - Must be kept up to date!

Source: Adapted from Blum 1992, pp164-5 and the IEEE-STD-830-1993

Typical Mistakes

- Noise = the presence of text that carries no relevant information to any feature of the problem
- Silence = a feature that is not covered by any text
- Over-specification = text that describes a feature of the solution, rather than the problem
- Contradiction = text that defines a single feature in a number of incompatible ways
- Ambiguity = text that can be interpreted in >=2 different ways
- Forward reference = text that refers to a feature yet to be defined
- Wishful thinking = text that defines a feature that cannot possibly be validated
- Jigsaw puzzles = e.g., distributing requirements across a document and then cross-referencing
- Inconsistent terminology = inventing and then changing terminology
- Putting the onus on the development staff = i.e. making the reader work hard to decipher the intent
- Writing for the hostile reader (fewer of these exist than friendly ones)

Source: Steve Easterbrook, U. of Toronto

Key Questions and Characteristics

- Remember the key questions "Why?" or "What is the purpose of this?"
- Feasible
- Needed
- Testable

A Few Syntactic Analysis Tools

- QuARS
  - Quality Analyzer of Requirements Specification
    http://www.sei.cmu.edu/publications/documents/05.reports/05tr014.html
- ARM
  - Automated Requirement Measurement Tool
    http://satc.gsfc.nasa.gov/tools/arm/