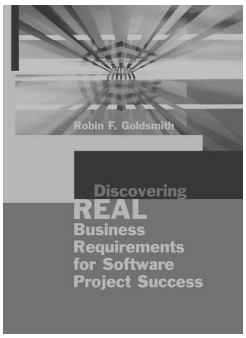
Involving Users Meaningfully in Testing

Robin F. Goldsmith, JD
GO PRO MANAGEMENT, INC.
SYSTEM ACQUISITION & DEVELOPMENT
QUALITY/TESTING
PRODUCTIVITY
BUSINESS ENGINEERING
TRAINING
22 CYNTHIA ROAD
NEEDHAM, MA 02494-1412
INFO@GOPROMANAGEMENT.COM
WWW.GOPROMANAGEMENT.COM
(781) 444-5753



Objectives

- Explain the need to keep technical testing separate from acceptance testing
- Describe dangers of over-relying on users' testing
- Identify technical testing tasks where users can participate productively, and where they shouldn't
- Show how typical user acceptance testing (UAT) impedes effective user contribution
- Present empowering methods for user-driven UAT

Have You Ever Heard: "Users have to do the testing; they're the only ones who know how the system's supposed to work"

Then what do developers and QA/Testing people know?

And how would users possibly have any way to know how a newly-developed system is supposed to work?

Amount of User Involvement Is the Most Important Project Success Factor

Closely followed by:

- Executive Management Support
- Clear Statement of Requirements
- Proper Planning
- Realistic Expectations

Reports the most widely-cited project success study

-- The CHAOS Report (1995)
The Standish Group
www.standishgroup.com

These are all inter-related, aren't they?

Main Factors Causing Projects to Be Challenged or Impaired/Canceled

Lack of User Input/Involvement
Incomplete Requirements & Specifications
Changing Requirements & Specifications
Unrealistic Expectations
Lack of Executive Support
Unclear Objectives

Ditto, they're inter-related too

Do user involvement issues differ for projects vs. testing?

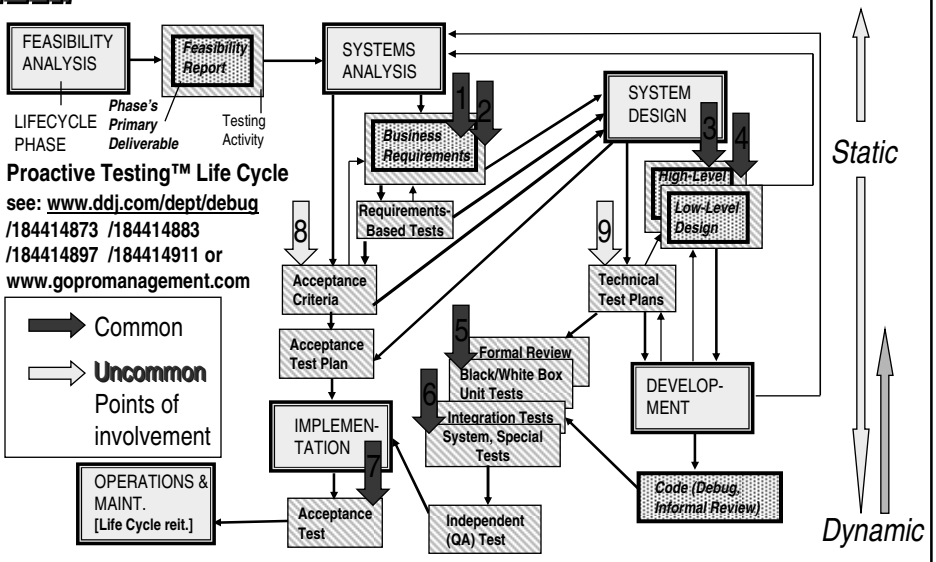
-- The CHAOS Report (1995)
The Standish Group
www.standishgroup.com

Time Is *Needed; But **Quantity** of User Involvement Alone Is Not the Full Story*

- *Inappropriate* and/or *ineffective* user involvement is
  - Far more typical than realized
  - At best limited in value
  - *More* such involvement actually may be counter-productive
- User unwillingness to cooperate/participate ("we're too busy" etc.) is often cited as a *cause* of problems but really may be an *effect* of user's expectations (perhaps based on own or others' prior experience) that involvement will be ineffective and/or unpleasant

How *users are involved is most important*

Changes Needed at Each User Involvement Point



Requirements Gathering: Inappropriate Approaches Limit User's Value/Interest

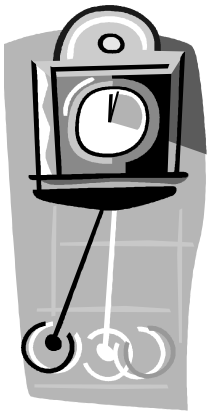
- When the Development side defines the requirements, they don't always listen well
  - Think they already know what is required
  - The "users don't know what they want" anyhow
  - Users don't talk about things that interest developers or talk in systematic terms that can be implemented in code
- Often (unconsciously) convey that users are dumb, compounded when users try to "talk techie"
- Routinely avoid taking the responsibility for their results needed to break the cycle—"enhancements"

Who Must Change? How?

Leaving Requirements Definition Solely to the Business People Is Ineffective Too

- Business domain knowledge is not the same as analytical skills
- Lack objectivity
- Hard to conceive of alternatives to the familiar

Pendulum swings to other extreme



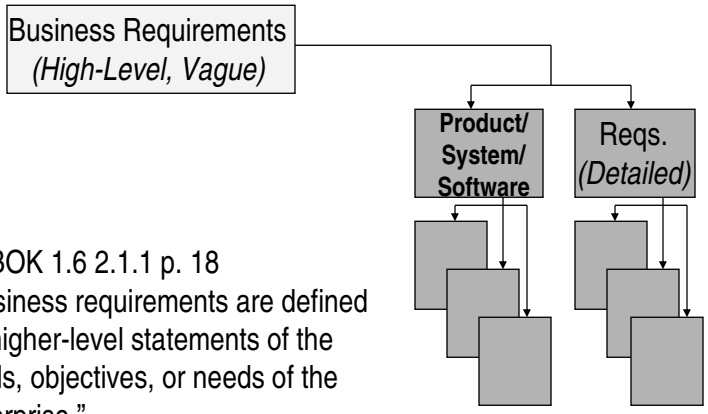
"We told them what we wanted. Surely they must understand."

"They said they wanted it. It must be the requirements."

➲Two Types of Requirements:

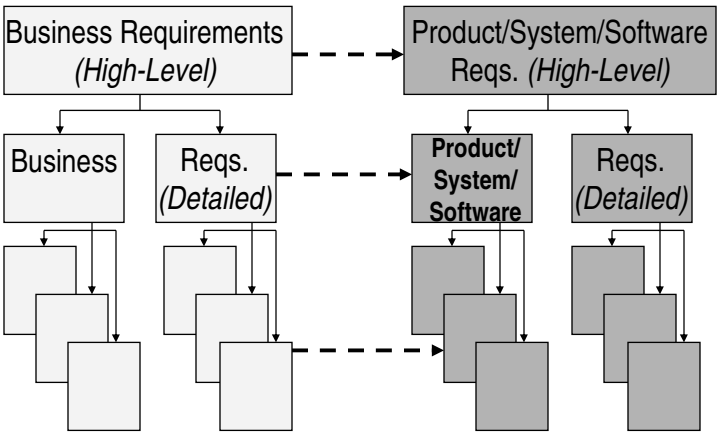
Business/User	Product/System/Software
• Business/user language & view, conceptual; *exists* within the business environment • Serves business objectives • ***What*** business results must be delivered to solve a business need (problem, opportunity, or challenge) and provide value when delivered/satisfied/met **Many possible ways to accomplish**	• Language & view of a *human-defined product/system* • **One of the possible ways *How*** (design) presumably to accomplish the presumed business requirements • Often phrased in terms of external functions each piece of the product/system must perform to work as designed (Functional Specifications)

Even Requirements "Experts" Think the Difference is Detail



BABOK 1.6 2.1.1 p. 18
"Business requirements are defined as higher-level statements of the goals, objectives, or needs of the enterprise."

When Business/User Requirements Are Detailed First, Creep Is Reduced; and Users Are Far More Productive and Interested



User Review of Requirements : Often Less Effective than Usually Presumed

2

Testing Activity

- Really more of a weak passive review of the analyst's dictation than an active challenging of the requirements--it's hard for anyone to recognize his/her own mistakes
- Users often are thrown into the review with little idea what to do, how to do it, or how to tell whether it was done well
- Well-meaning but uninformed involvement of others (e.g., QA's over-emphasizing testability) actually can further distract from awareness of review's weakness

We have 21+ ways to test that requirements are right

Ineffective High Level Design Role Can Put Users on Egg Shells

3

- Often inadvertent involvement--what people call requirements definition is really design
- Business knowledge does not equal design skill
- Users are asked what they want, then zapped
  - Criticized for not thinking of everything, especially more technical and architectural issues
  - Later told, "But you signed off on it," when complains that it's not right

Design product after discovering REAL business requirements

User Review of Designs Can Be Illusory and Ineffective

4

Testing Activity

- Often (like others, unfortunately, including management, developers/analysts, and QA/Testing too) has little real idea what to look for or how to participate effectively
- Can be lost when reviewing technical aspects
  - "Trust us, we're professionals" is common, means the user's time really isn't needed
  - Technical folks have been known to be patronizing and demeaning of those without technical knowledge
- Again zapped with "You signed off on it" when later complaining about design adequacy

We have 15+ ways to test that designs are right

User Involvement in Unit Testing Is Generally Inappropriate and Ineffective

5

Testing Activity

Developers must know how to tell whether system is right is in order to do competent development.

When pushed to do technical testing, users often have an underlying sense of, "Isn't this the developer's job?"

- Unit tests focus on system structure, which is not the same as what the business needs, and is not the user's strength or way of thinking
- Developers think errors are a normal part of development; users think they're the way the system works
- The more a user attends to how the system is designed to work, the less s/he tends to stay aware of what it should accomplish (co-optation)

Testing usability of working system is exception

System Testing that Over-Relies on Users Is Ineffective and Inappropriate

6

Testing Activity

- System testing typically should include exercising technical aspects that users would not know about or have tools/skills needed
  - Internal structures, databases, environment
  - Security, performance levels, degradation, recovery
- Even though user is involved, system test's production operations orientation tends to distract from and/or co-opts the different perspective of user acceptance testing
- "You tested it" greets later problem reports

System tests should be driven from technical side

Typical Inappropriate User Acceptance Testing Structure Makes It Ineffective

7

Testing Activity

- ⦸ Consider UAT a test to confirm product's *system* requirements
- ⦸ Often expect a rubber stamp—users only need bother with a single positive/valid test per functional requirement
- ⦸ Many supposed testing experts say UAT should simply be a subset of the system tests rerun by users

- ➢ ***Users should be confirming system meets the* business requirements**
- ➢ ***Purpose of any test is to demonstrate and confirm, not assume, that system works***
- ➢ ***Independent user view finds errors they miss***

Typical Way UAT Is Presented to Users Cuts Competence, Confidence, Cooperation

"Here's the new system. Try it out. Play with it."

How? Play?

"Push this button. Watch me do it. See, it works."

Huh?

"Don't do anything stupid."

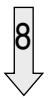
Yikes! I broke it!

"User error!"

Really a design error, isn't it?

Testing Activity

User Defines Acceptance Criteria Without Referring to Requirements



- What the users/customers/stakeholders (*may be multiple perspectives*) must have demonstrated to be confident the delivered system works before they're willing to stake their job on system
- Determination will be made whether or not it is conscious, planned, or explicit
- True empowerment builds cooperation
  - Ability to accept or reject delivered system
  - Thinking through how decision will be made

User Participation in Proactive Master Test Planning Prevents Major Cause of Overruns

9

Testing Activity

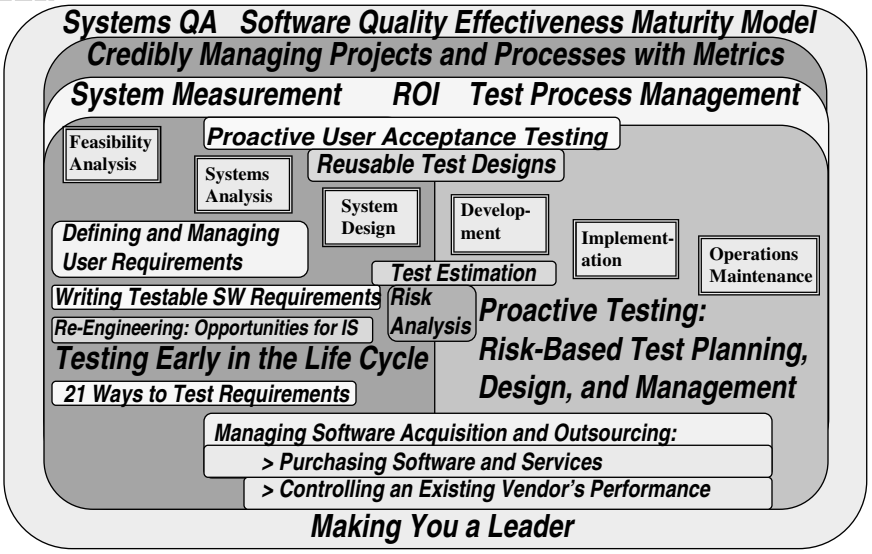
- From showstoppers that traditional reactive testing does not catch, at the worst possible time
- Late unplanned redesign and rework is often up to 80 percent of total project effort--all unanticipated
- Proactive Testing™ lets testing drive development to avoid throwing out and replacing significant amounts of completed work

The Stomach-Ache Metric

Summary

- The amount of time a user spends is only one of the factors affecting system development. How the time is spent is also important; and just spending more time inappropriately could be counter-productive.
- Though seldom recognized, many of the common forms of user involvement actually may be inappropriate and/or ineffective, typically emphasizing product/system design rather than REAL business requirements, thereby limiting the value of the user's time and possibly also reducing user cooperation.
- There are at least nine points in the development process where more appropriate and effective approaches to user involvement can increase value to the project as well as to the user.

Go Pro Management, Inc. Seminars--Relation to Life Cycle



Robin F. Goldsmith, JD

robin@gopromanagement.com (781) 444-5753
www.gopromanagement.com

- President of Go Pro Management, Inc. consultancy since 1982, working directly with and training professionals in business engineering, requirements analysis, software acquisition, project management, quality and testing.
- Previously a developer, systems programmer/DBA/QA, and project leader with the City of Cleveland, leading financial institutions, and a "Big 4" consulting firm.
- Degrees: Kenyon College, A.B.; Pennsylvania State University, M.S. in Psychology; Suffolk University, J.D.; Boston University, LL.M. in Tax Law.
- Published author and frequent speaker at leading professional conferences.
- Formerly International Vice President of the Association for Systems Management and Executive Editor of the *Journal of Systems Management*.
- Founding Chairman of the New England Center for Organizational Effectiveness.
- Member of the Boston SPIN and SEPG'95 Planning and Program Committees.
- Chair of BOSCON 2000 and 2001, ASQ Boston Section's Annual Quality Conferences.
- Member ASQ Software Division Methods Committee.
- Member IEEE Std. 829 for Software Test Documentation Standard Revision Committee
- Admitted to the Massachusetts Bar and licensed to practice law in Massachusetts.
- Author of book: ***Discovering REAL Business Requirements for Software Project Success***