Putting the User Back in User Acceptance Testing

Robin F. Goldsmith, JD

GO PRO MANAGEMENT, INC.

SYSTEM ACQUISITION & DEVELOPMENT

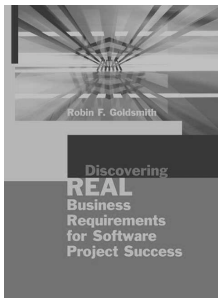
QUALITY/TESTING

PRODUCTIVITY

BUSINESS ENGINEERING

TRAINING

22 CYNTHIA ROAD
NEEDHAM, MA 02494-1412
INFO@GOPROMANAGEMENT.COM
WWW.GOPROMANAGEMENT.COM
(781) 444-5753 *VOICE/FAX*



Most Acceptance Testing Is Reactive

- At the end, based on system as written
- Technical view dominates with little user role/awareness of how and what to test
- A lot of work, often with little payback
  - Doesn't find many of the errors
  - Adversarial, arguments, blame
  - Too late to fix errors anyhow

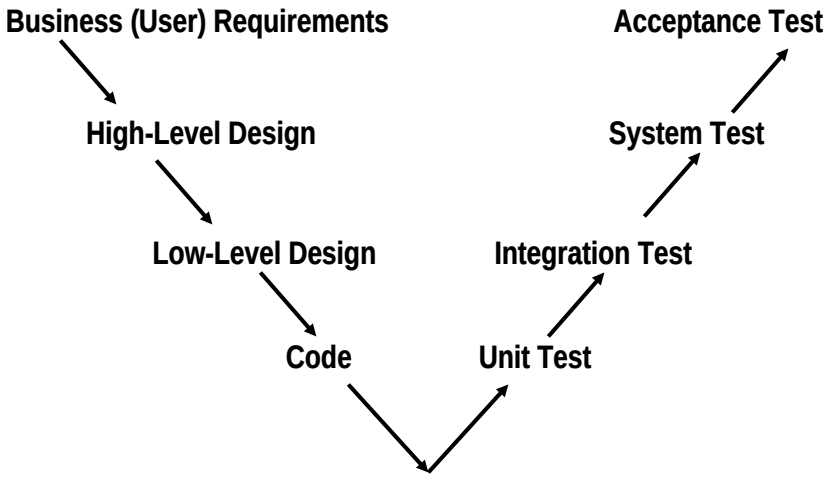
Lacking competence leads to lacking confidence leads to lacking commitment

Proactive Testing™ provides the basis for more confident and competent user commitment to UAT

Objectives

- Identify reasons users often are reluctant to participate in User Acceptance Testing (UAT)
- Distinguish conventional testing's often ineffective approaches to UAT from more successful Proactive User Acceptance Testing™
- Describe professional testers' role in helping users define Acceptance Criteria that create competence, confidence, and commitment

"V" Model Shows Levels of Testing



Reasons for User Resistance

- Lack of confidence in what and how to test
- Testing against system, not business requirements
- Insufficient time or resources (may signal other issues)
- "Not my job" syndrome [may be correct]
- Don't find much anyhow
- Fear of blame for found or unfound defects
- Belief that feedback will be ignored
- Intimidation and misguidance from developer- and/or tester-centric dominance

Traditional Development and Testing Gurus Tend to Miss the Boat on UAT

- ⦸ Consider UAT a test to confirm product's *system* requirements--often expect a rubber stamp
- ⦸ Some testing books/courses say
  - ⦸ UAT should be only positive/valid proof-of-concept tests to demonstrate normal functionality
  - ⦸ UAT should be a repeated subset of the System Test, run by users
  - ⦸ UAT simply needs one test for each functional requirement or use case scenario

Yet, organizations continually rely on UAT to catch many missed problems

Why Do Users Need to Do Thorough User Acceptance Testing (UAT)?

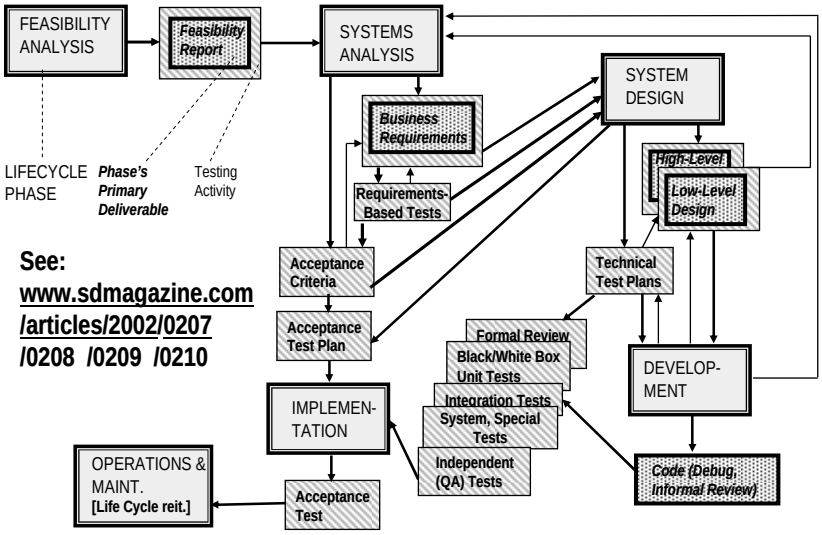
Self-defense:

Acceptance Testing is the user's chance and duty to confirm the system works properly, from the user's standpoint, before users and the organization rely upon it.

Users often are like goalkeepers defending alone without protection from fullbacks and halfbacks

Can't we just trust the developers to do their jobs right?

Proactive Testing™ Life Cycle



See:
www.sdmagazine.com/articles/2002/0207/0208 /0209 /0210

Key Proactive Testing™ Concepts
Find WIIFMs

- Intertwine testing with each development deliverable
- Plan before acting at any point/level, independent paths
  - Acceptance testing first; users shouldn't do technical tests
  - Prepare test plans/designs during Design, promote reuse
  - Prioritize by level, from full choices, to avoid wasted effort
- Let testing drive development
  - Feedback test plans/designs to guide correct coding
  - Selectively structure for early tests that prevent rework
  - Measure objectively to guide decisions, more CAT-Scans

Proactive Testing™ Strategy/Roles: More Thorough Tests of Cleaner Code

- Enable <u>developers</u> to code more correctly by using testing to improve requirements and designs
- Enable <u>developers</u> to catch more of their own (fewer) remaining errors by more explicit systematic testing from business and technical perspectives
- <u>Professional testers</u>
  - Specialize in test methods
  - Tend to like testing, detail
  - Double check, any/all test levels, technical view, but with greater user sensitivity
- <u>Users</u> double check cleaner software from business perspective as they actually will use system

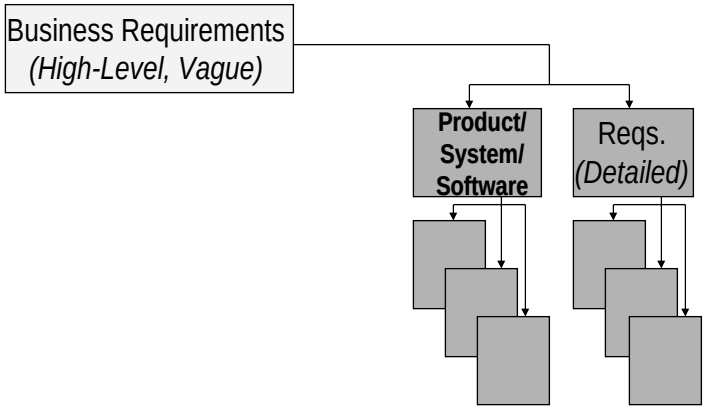
Approaches to Acceptance Testing

Reactive--Run what's given to you	Proactive--Plan what to run
● Here it is, try it out – Assumes understanding of how system works and is to be used – Reasonable for changes to known system ● (Variant) Test that it works the way it's written – Check choices, links, fields – Match to user instructions	✓ Test each requirement ✓ Minimum for acceptability ✓ Assures requirements are testable (sufficient?) ✓ (Variant) Run Use Cases ✓ Define acceptance criteria ✓ Independently of requirements, sets priorities ✓ A test of the requirements ➢ Execute the plan

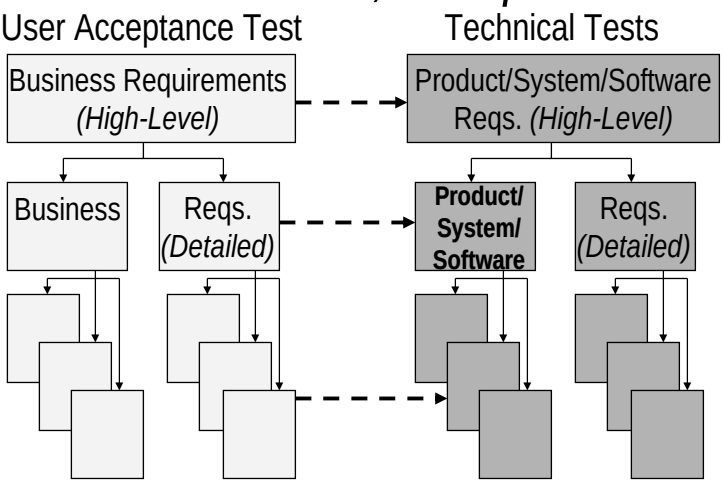
➲ Two Types of Requirements:

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



When Business/User Requirements Are Detailed First, Creep Is Reduced



What Tests Are Needed to Be Confident This Requirement Is Met?

Separately identify sales tax calculated as 5% of purchase price and include it in the total amount due

Define the Use Case Scenario(s) that Test This Requirement?

Requirements-Based Test Issues: Must Be **Business** *Requirements*

- Tests are only as good as the requirements definition-irrelevant for missing/wrong
- It can be hard to tell what the requirements are
  - Narratives are confusing
  - Itemized format helps
  - Needs detail, much more
- Need at least 2 tests per requirement, usually more
- Use Cases define *usage, not user, requirements* of a product
  - Users are unlikely to write Use Cases; they'll use existing ones, but then not user's view
  - Don't address business rules or quality factors
- Seldom written from a thorough testing perspective, happy path but not the multiple (esp. negative) test conditions

Acceptance Criteria--Define Without Referring to the Requirements

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

Identifies overlooked and incorrect requirements

1. What Functions the System Must Perform

- From user's perspective
  - Real world business processes, not programs
  - Original source through end-use
  - Common and likely to be problematical
- May include structure, e.g., checking every menu choice, every field, all outputs
- Technical role is facilitator, gatherer, organizer (normal, not just new/problems)

2. How Much Must Be Shown to Give Confidence It Works

- How many (and which)
  - Transactions, accounts
  - Users, locations
  - Days, business cycles
- How frequently, how extensively should they be demonstrated
- What environments should they be in

3. How Well, How System Quality Will Be Assessed

- Performance measures
  - Response times, turnaround, throughput
  - Staffing requirements and productivity
  - Accuracy, errors and responses to them
- Subjective judgments
  - Usability
  - Suitability
  - Whose opinion counts, guidelines

4. By Whom, Who Should Perform Tests for Confidence We Can Use It

- Actual end-users themselves
  - Typical, randomly selected
  - Mix of locations, skills, roles
- "Super," "power," or "good" users
- User stand-ins, e.g., business analysts, liaisons, temporary helpers, full-time acceptance testers
- Non-users, e.g., QA or testing group, documentation writer, developers

5. *In What Manner, Test Format for Confidence the System Works*

- Stand-alone--test inputs and outputs
- Parallel--production inputs, test outputs
  - Most complete definition of "right" results
  - Old, new, reconcile is three times the work
- Pilot--subset, production inputs and outputs
  - No definition of expected results
  - Bridging adds complexity and risk to test
- Big Bang--production inputs and outputs

What Acceptance Criteria Does This Requirement Suggest?

Separately identify sales tax calculated as 5% of purchase price and include it in the total amount due

Functions

Acceptance Criteria (cont.)

How Much

How Well

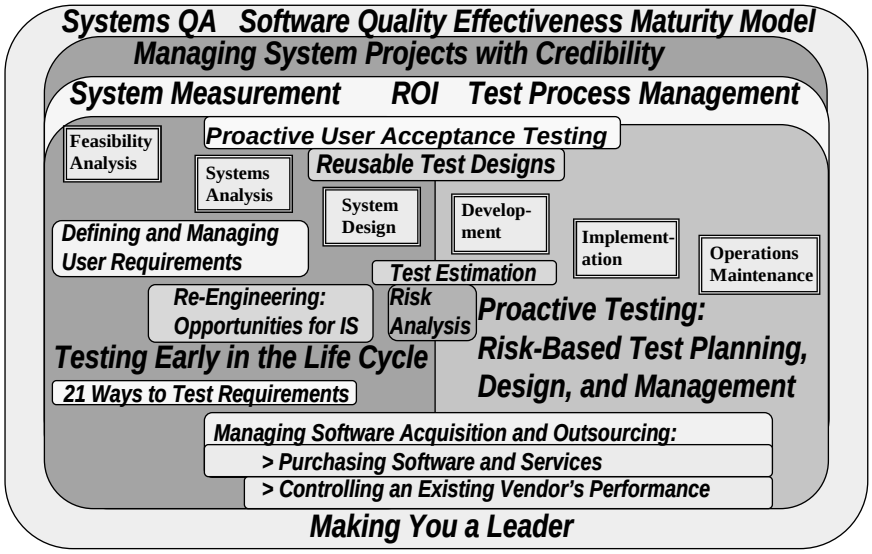
By Whom

In What Manner

Summary

- Perceived lack of competence and confidence often makes users reluctant to participate in User Acceptance Testing (UAT)
- Conventional testing's common testing-centric views of UAT render UAT an ineffective disempowering rubber-stamp time-waster
- Proactive User Acceptance Testing™ empowers users by defining Acceptance Criteria that are important to the users and create competence, confidence, and commitment

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. ProveIT.net ROI Value Modeling™ partner.
- 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 Software Test Documentation 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***