Avoid Creep—Discover the REAL Requirements

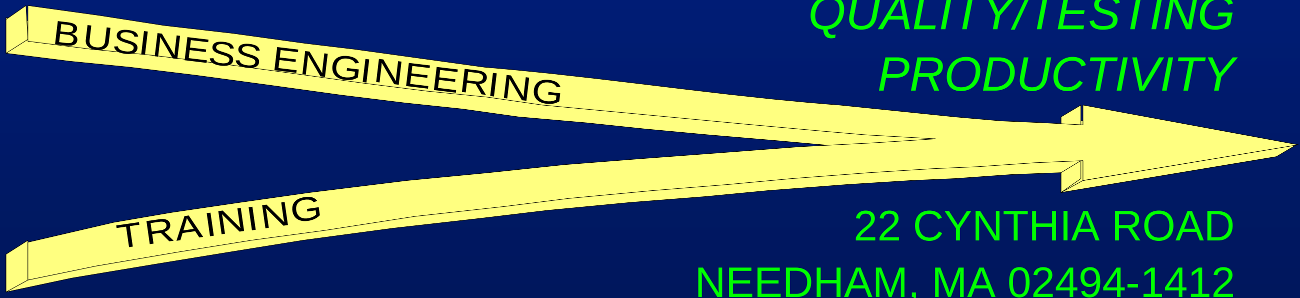
Robin F. Goldsmith, JD

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

How to Cook a Ham

- Cut off ends ***Why?***
- Put in roasting pan
- Cover with candied yams, raisins, and honey ***?***
- Cook at 375 degrees for 20 minutes plus 12 minutes per pound ***Sure?***

This is like a use case. Is it the requirements?

Is the Business Requirement “Cook a Ham”?

How about: Provide food

What if:

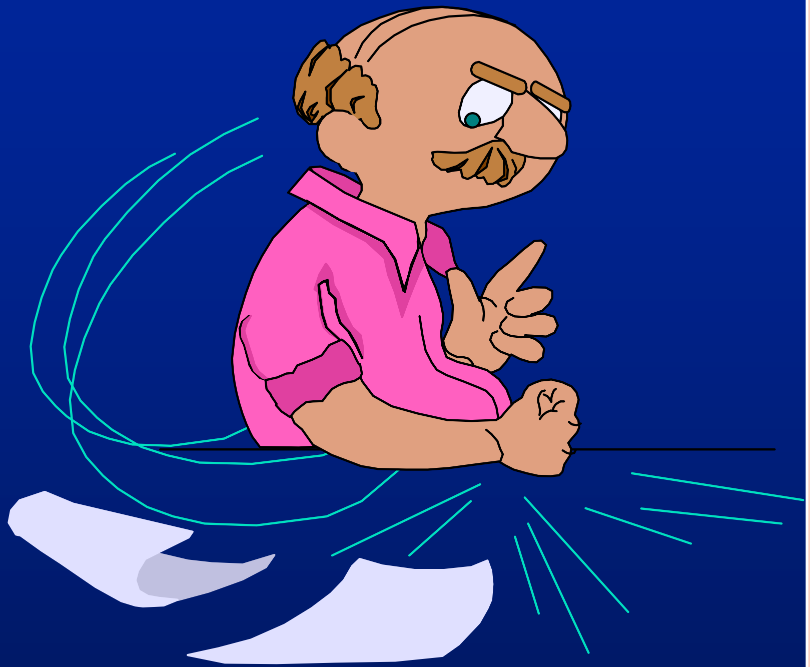
It’s Thanksgiving, breakfast
We need to feed 2 people, 200 people
They’re Orthodox Jews or Muslims
They’re diabetic
We use a microwave oven instead

Where’s a computer fit in?

Objectives

- Identify and differentiate “requirements” which are
  - Business/user/customer/stakeholder requirements
  - Product/system/software/functional (and non-functional) requirements/specifications
  - Use cases
- Introduce the Problem Pyramid™ technique for identifying the REAL business/user requirements
- Describe seven guidelines for documenting business requirements

Are You Familiar With this Scenario?



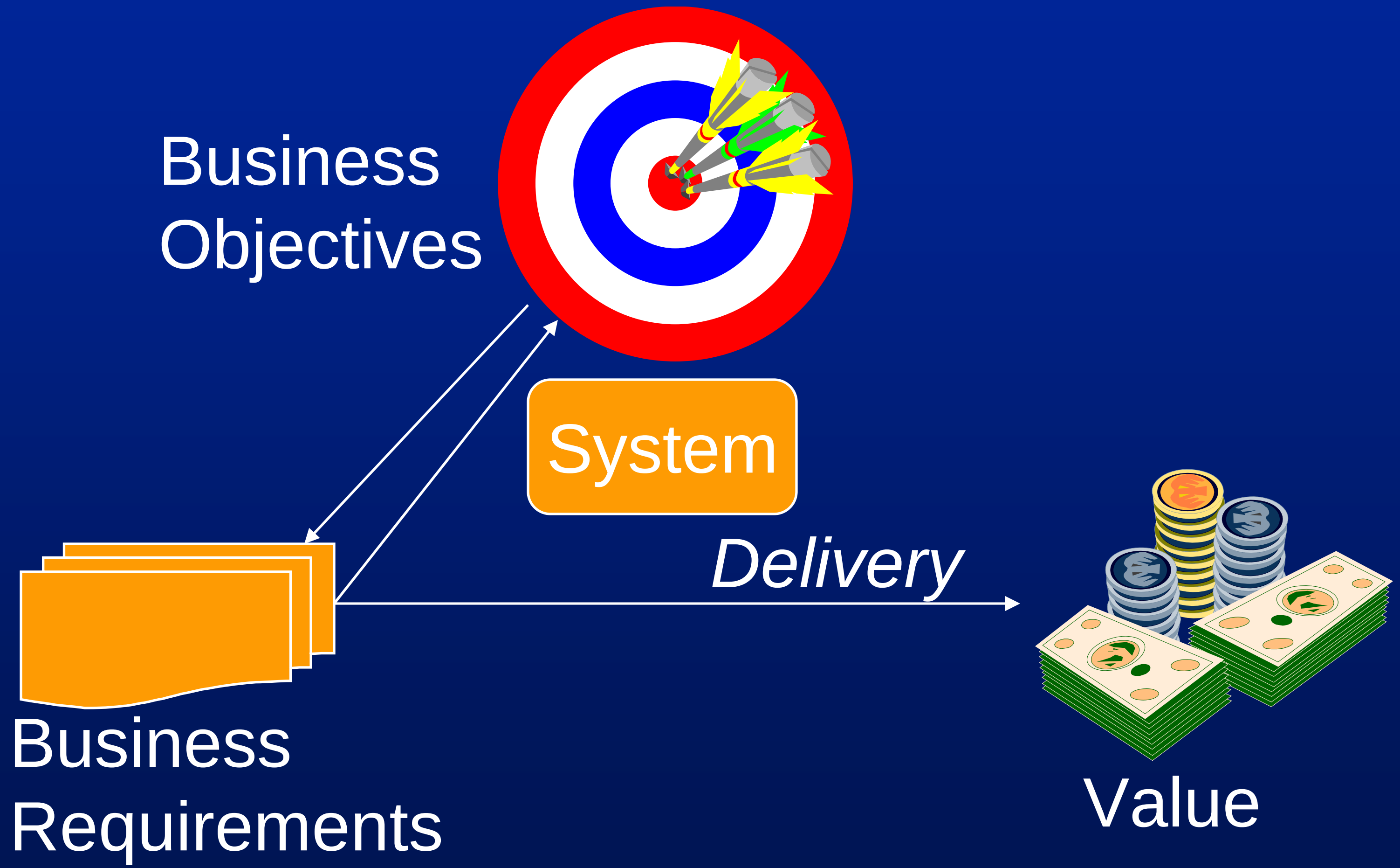
It's not right!

It's what you said you wanted.

It's still not right!

Just Because a User Says It
Doesn't Make It a REAL Business/User Requirement

Business/User Requirements: What Must Be Delivered to Provide Value



Business Objectives

System

Delivery

Business Requirements

Value

Typical Business Requirements

- Automated Teller Machine (ATM) must
  - Require customer to insert card
  - Read encrypted card number and ID data from magnetic stripe
  - Require customer to enter PIN (Personal Identification Number)
  - Match entered PIN to calculated PIN or PIN on file
  - Accept envelopes containing deposits or payments
  - Dispense cash in multiples of $10
  - Display selected account balances
  - Transfer indicated amounts between customer’s accounts
  - Issue printed receipts
  - Return customer’s card

Then What Are These?

- Provide secure, confidential access to banking services at time and location convenient to customer
- Confirm identity of customer
- Enable customer to perform ordinary bank transactions himself/herself quickly and accurately
- Provide customer documented evidence of the transactions

Note: ATM could be preferred operational style

➲ Two Types of Requirements:

Business/User/Customer

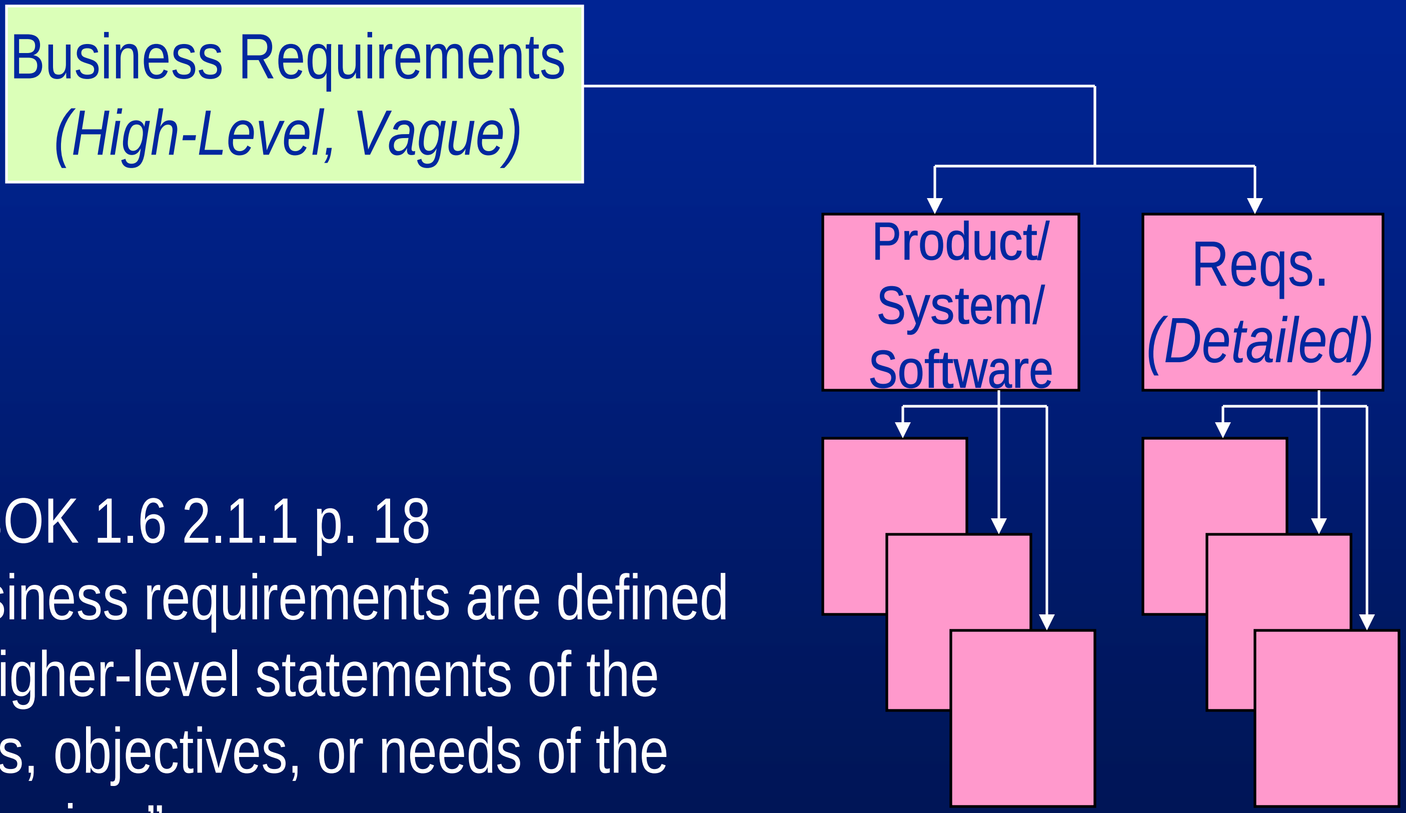
- Business/user/stakeholder/ customer language & view, conceptual; *exist* within the business environment
- Serves business objectives
- *What* business results must be delivered to solve a business need (problem, opportunity, or challenge) and provide value when delivered/satisfied/met

Many possible ways to accomplish

Product/System/Software

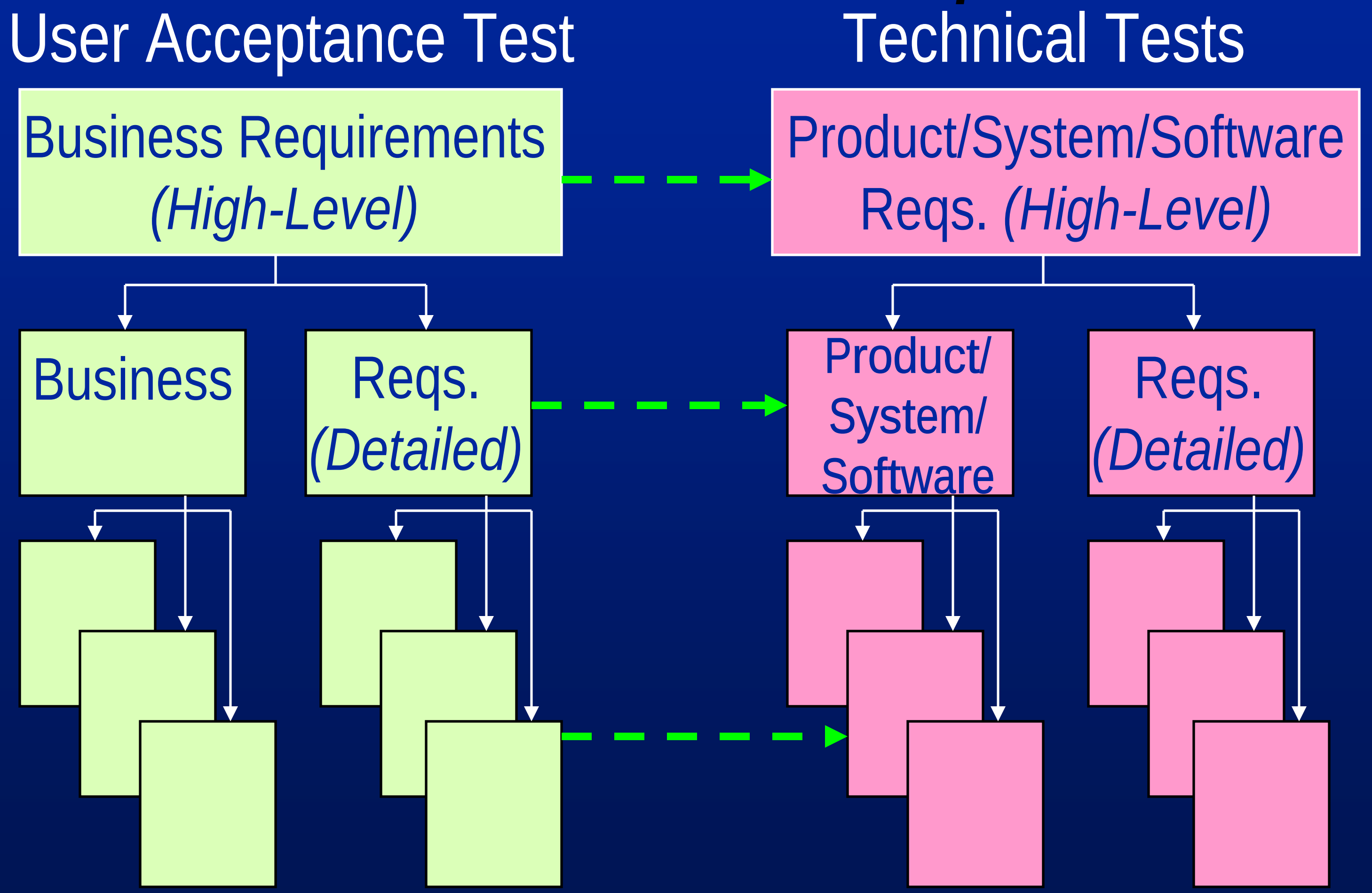
- Language & view of a *human-defined product/system*
- **One of the possible ways** *How* (design) presumably to accomplish the presumed business requirements
- Often phrased in terms of external functions each piece of the product/system must perform to work as designed (Functional Specifications)

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



User Acceptance Test

Technical Tests

Business Requirements *(High-Level)*

Product/System/Software Reqs. *(High-Level)*

Business

Reqs. *(Detailed)*

Product/ System/ Software

Reqs. *(Detailed)*

Common Erroneous Perceptions About Business/User Requirements

Whatever is requested/declared/dictated by business users or managers

Not an issue for small changes

Desired benefits

A product of requirements engineering or requirements management

What users should provide for developers to code from

Use Cases for Global Web-Based Reuse Repository

Add Reusable Item

1 Enter item's title and description in database

2 Upload item to server

Access Reusable Item

1 Enter keyword to search for in item title

2 Scroll list of titles

3 Display item's description

4 Download item from server

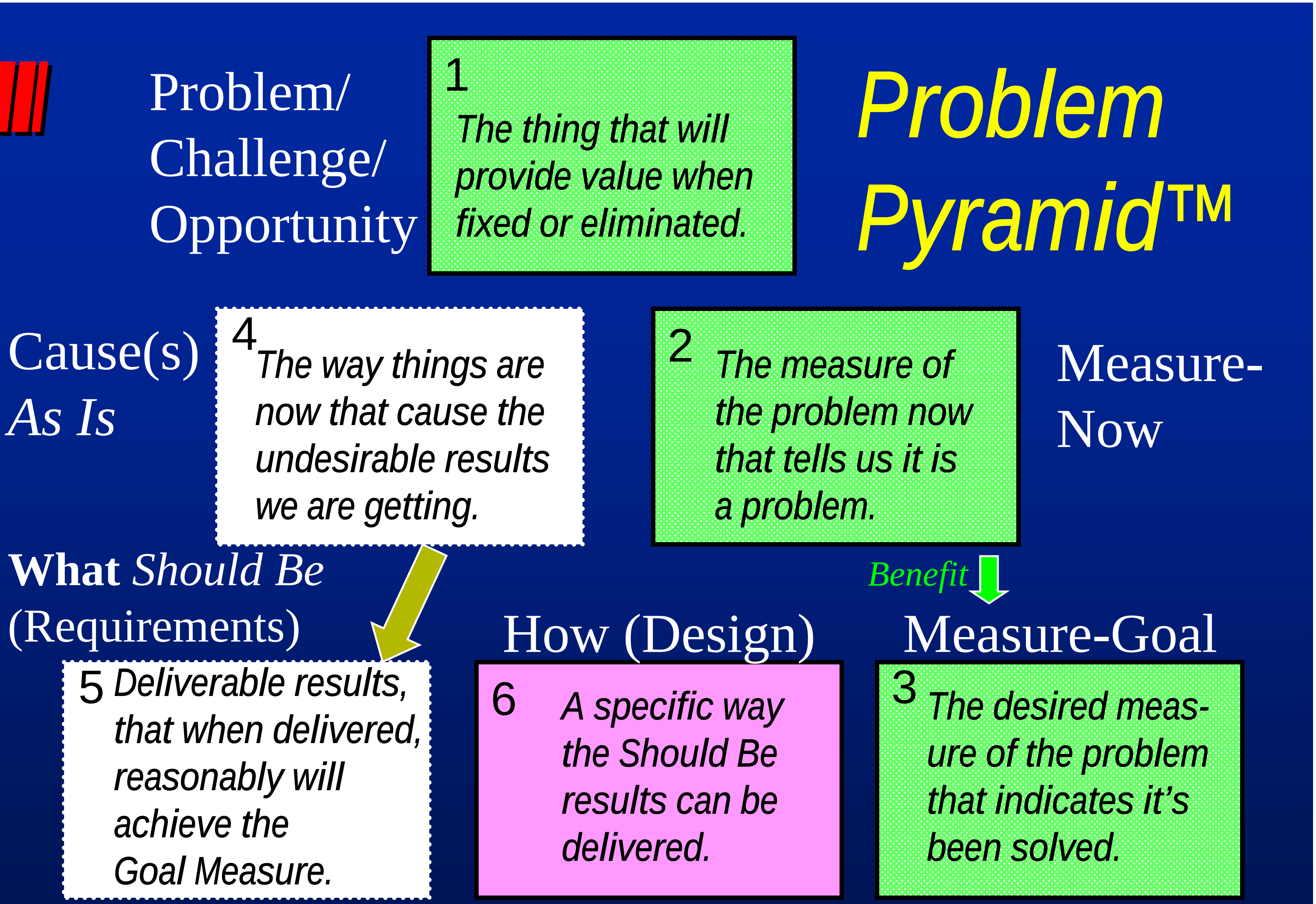
Is this the user's requirements? Some gurus say it is. Defining intended* usage *surely facilitates programming.

Functional Specifications Are These the User's Requirements?

- With any supported browser (see list) over the Web or Intranet from anywhere in the world
  - User can add a reusable item to the repository
  - User can retrieve a reusable item from the repository
- User selects type of reusable artifact: requirements, design, source code, executables
- User displays, selects , and downloads from list of specific reusable artifacts of the indicated type

How do these contrast with the use case? Ready to code?

Problem Pyramid™



Problem/ Challenge/ Opportunity

1 *The thing that will provide value when fixed or eliminated.*

Cause(s) *As Is*

4 *The way things are now that cause the undesirable results we are getting.*

2 *The measure of the problem now that tells us it is a problem.*

Measure-Now

What *Should Be* (Requirements)

Benefit

How (Design)

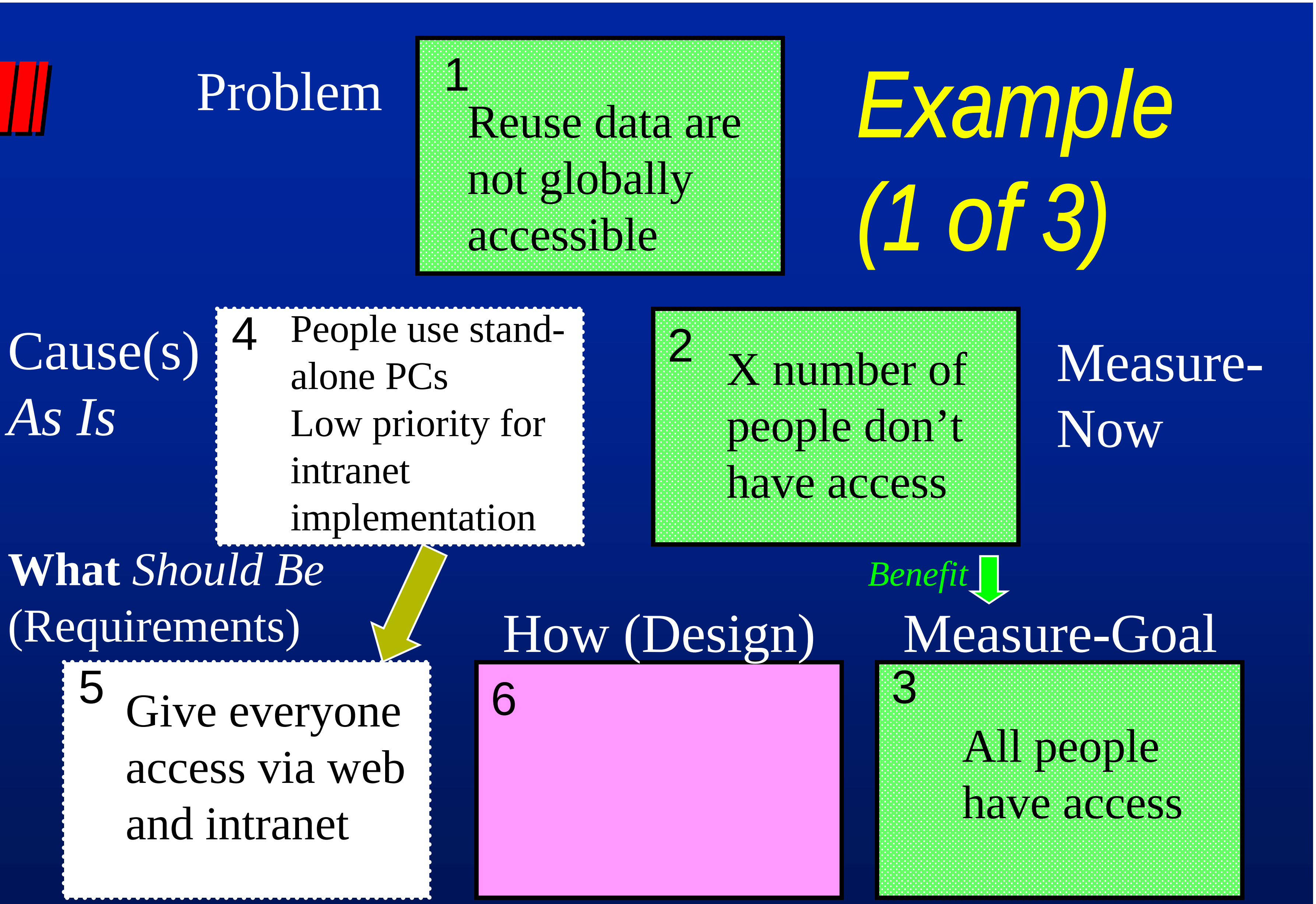
Measure-Goal

5 *Deliverable results, that when delivered, reasonably will achieve the Goal Measure.*

6 *A specific way the Should Be results can be delivered.*

3 *The desired measure of the problem that indicates it's been solved.*

Example (1 of 3)



Problem

1 Reuse data are not globally accessible

Cause(s) *As Is*

4 People use stand-alone PCs
Low priority for intranet implementation

2 X number of people don't have access

Measure-Now

What *Should Be*
(Requirements)

Benefit

How (Design)

Measure-Goal

5 Give everyone access via web and intranet

6

3 All people have access

Guidelines for Getting the Problem Pyramid™ Right (1 of 2)

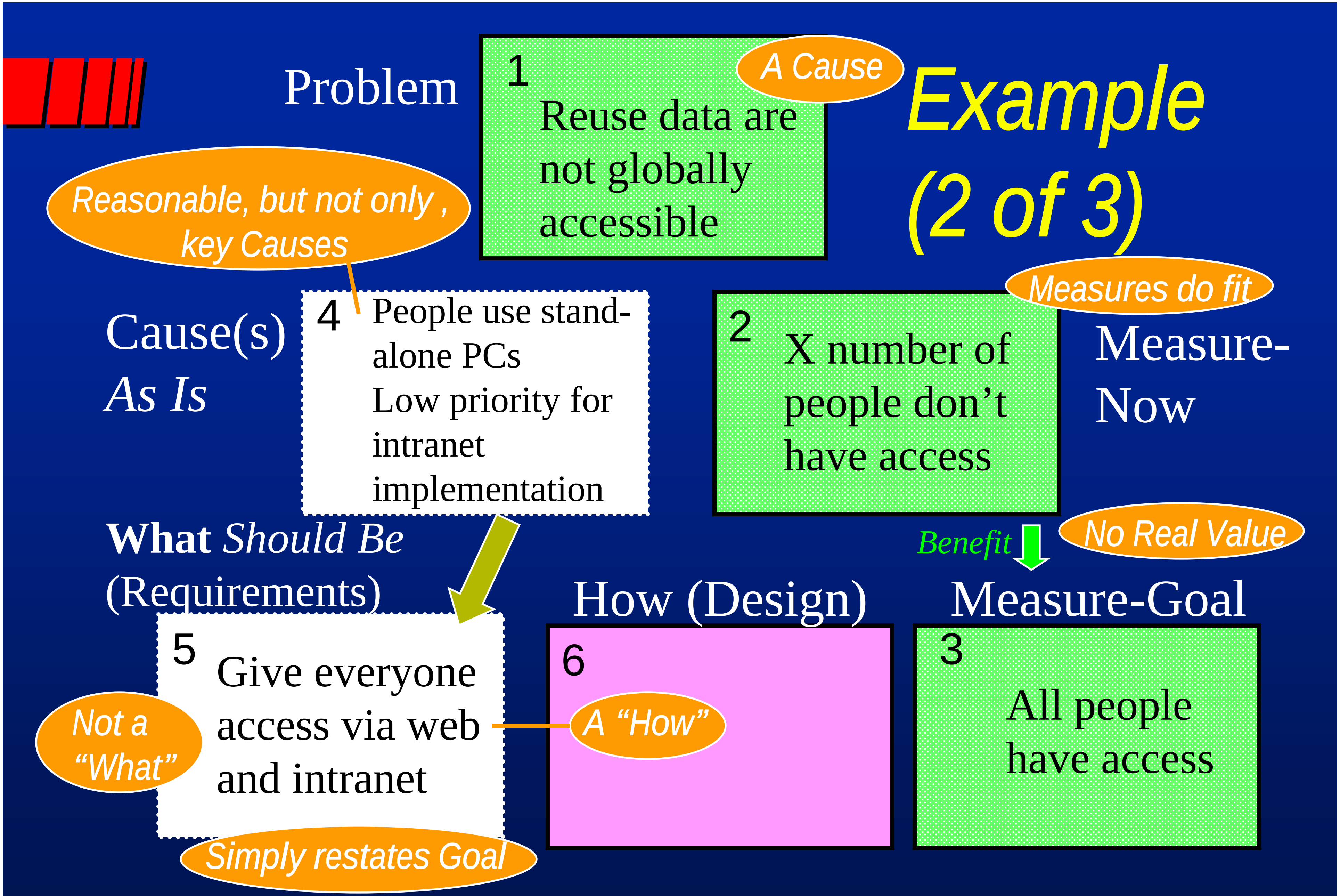
- Is the **Problem** really the problem?
  - Do the measures fit it?
  - Does it provide real value when goal achieved?
- Are the **Causes** in fact causes?
  - Do they reasonably cause the Problem?
  - Have we identified all the likely key causes?
- Does the **Should Be** solve the Problem?
  - Is it "Whats," results likely to meet goal?
  - Does it address (and reduce/eliminate) each key Cause?
  - What else to address that this affects or is affected by?

Guidelines for Getting the Problem Pyramid™ Right (2 of 2)

- Problems can be hierarchical, in which case Problems at one level are Causes of a Problem at the next higher level
- Causes can seem like Problems
  - A Cause can be hierarchical too, with contributing sub-causes
  - A Cause can have Current and Goal Measures
  - But, achieving a Cause's Goal Measure does not produce Real Value
- The appropriate level of Problem to use for defining Requirements is
  - The lowest level Problem, which
  - Produces Real Value when the Goal Measure is achieved

Why are we doing it at all? What if we just didn't do it?
Take measures to extremes

Example (2 of 3)



Problem

1 Reuse data are not globally accessible

A Cause

Reasonable, but not only , key Causes

Cause(s) *As Is*

4 People use stand-alone PCs
Low priority for intranet implementation

2 X number of people don't have access

Measures do fit

Measure-Now

Benefit

No Real Value

What *Should Be* (Requirements)

How (Design)

Measure-Goal

5 Give everyone access via web and intranet

Not a "What"

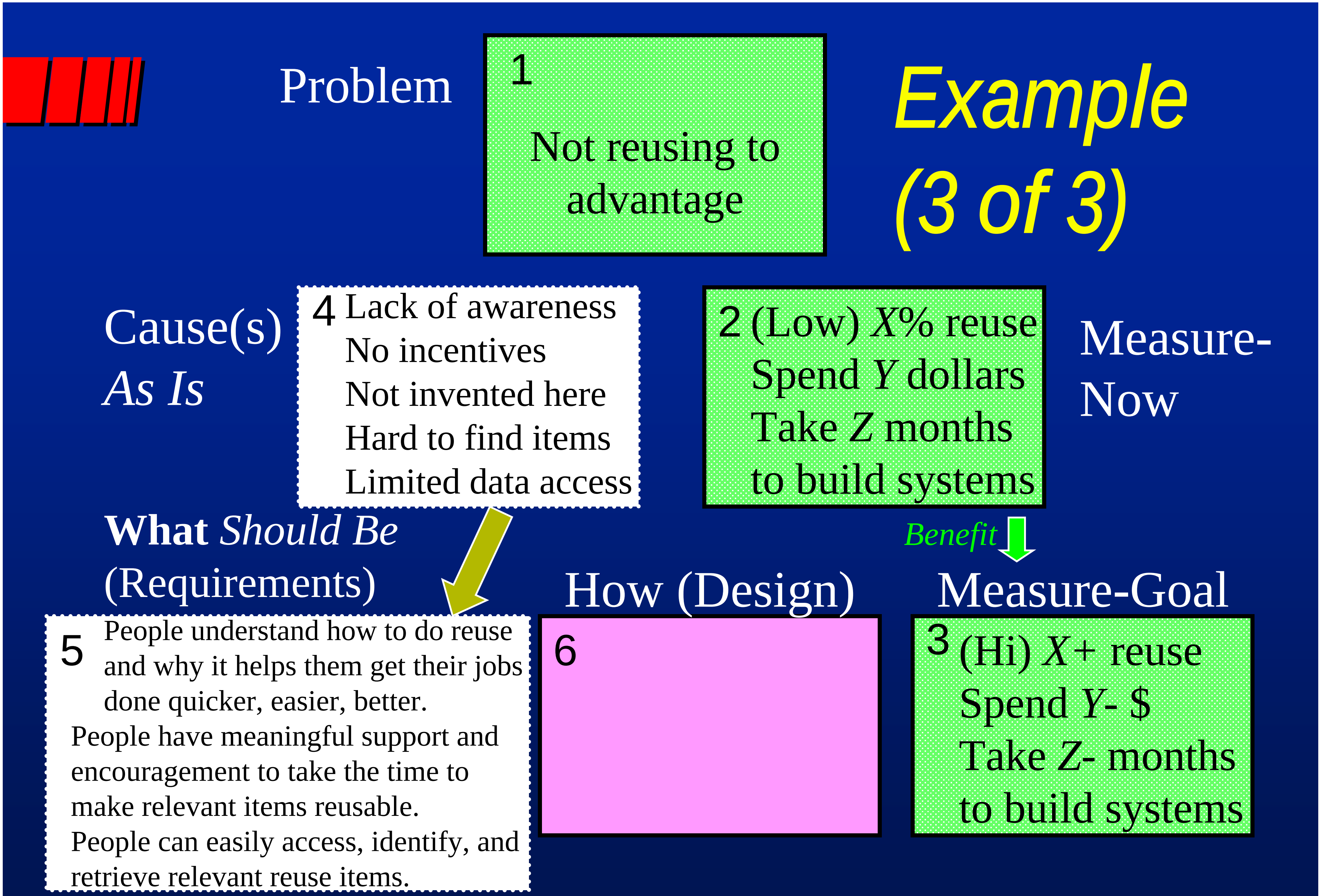
Simply restates Goal

6

A "How"

3 All people have access

Example (3 of 3)



Problem

1 Not reusing to advantage

Cause(s) *As Is*

4 Lack of awareness
No incentives
Not invented here
Hard to find items
Limited data access

2 (Low) *X*% reuse
Spend *Y* dollars
Take *Z* months
to build systems

Measure-Now

Benefit

What *Should Be* (Requirements)

5 People understand how to do reuse and why it helps them get their jobs done quicker, easier, better.
People have meaningful support and encouragement to take the time to make relevant items reusable.
People can easily access, identify, and retrieve relevant reuse items.

How (Design)

6

Measure-Goal

3 (Hi) *X*+ reuse
Spend *Y*- $
Take *Z*- months
to build systems

7 Guidelines for Documenting Requirements

❶ Go top-down in the user's/customer's language, focusing on end results/outputs of the "should be" business model; address

- Strategy, mission, competitiveness
- Business functionality, business rules
- Management information, controls
- Operational effectiveness, workflow
- Performance and quality ("ility") levels
- Interfaces, environment, project compatibility/constraints

7 Guidelines for Documenting Requirements (cont.)

❷ Identify everything the system should do

- ➲ Routine functions
- ➲ Exceptional, unusual, and ideal functions

❸ Anticipate change and support

❹ Break down each item to a level of detail

- ➲ Clear, complete, testable, and stands on own
- ➲ Meaningful to people in the user/customer industry

7 Guidelines for Documenting Requirements (cont.)

❺ Supplement as appropriate with narratives, diagrams, and examples

❻ Prioritize and weight meaningful groupings (being attentive to interdependencies)

- ➲ Mandatory, desirable, ideal
- ➲ Weighting (1=low to 10=high)
- ➲ Rank (must be limited number of groupings)
- ➲ 100-point “must system” (limited number)

❼ Every item is an observable deliverable

REAL Business/User Requirements

Compare to Use Cases/Func Specs 1 of 5

Objective: Reduce development time, cost, and defects by increasing effective reuse of development artifacts.

1. People understand how to do reuse and why it helps them get their jobs done quicker, easier, better.
   - A. Types of reusable artifacts can be defined.
     - 1) Minimum standard set for everyone’s use.
       - [a. Requirements….]
     - 2) Custom/unique artifacts for selected use.
     - 3) New artifacts and groupings can be added.
   - B. Rules for making artifacts reusable are defined.
   - C. Guidelines for reusing artifacts are defined.

Scope?

REAL Business/User Requirements

Compare to Use Cases/Func Specs 2 of 5

2. People have meaningful support and encouragement to take the time to make relevant items reusable.
   - A. Tasks to achieve each reusability characteristic are defined.
   - B. Resource, effort, and duration estimates for these tasks are included in initial project plans.
   - C. Approved project plans include necessary tasks with their associated added time and cost to make items reusable where payback is likely within one year.

Hierarchical itemized business deliverable whats that lead to value

REAL Business/User Requirements

Compare to Use Cases/Func Specs 3 of 5

3. People can easily provide relevant reuse items.
   A. Where time and budget has been approved.
   B. To add item to repository, must also provide:
      1) Type of artifact (from approved list).
      2) Title briefly describing the item’s use.
      3) Longer, complete description of the item.
      4) Instructions for reusing the item.
      5) Identification of related items.
      6) Source of further information and assistance.
   C. Item’s creator can modify items in the repository.

REAL Business/User Requirements

Compare to Use Cases/Func Specs 4 of 5

4. People can easily identify relevant reuse items.
   A. Items can be located/accessed within type by:
      1) Title.
      2) Keywords (in title and additional).
      3) Related uses of same or other type.
         a. Via reuse history for the item.
         b. Where item is derived from another.
   B. The item and related descriptive information can be accessed within five minutes of identifying it.
   C. Degree of access is limited by authorization level.

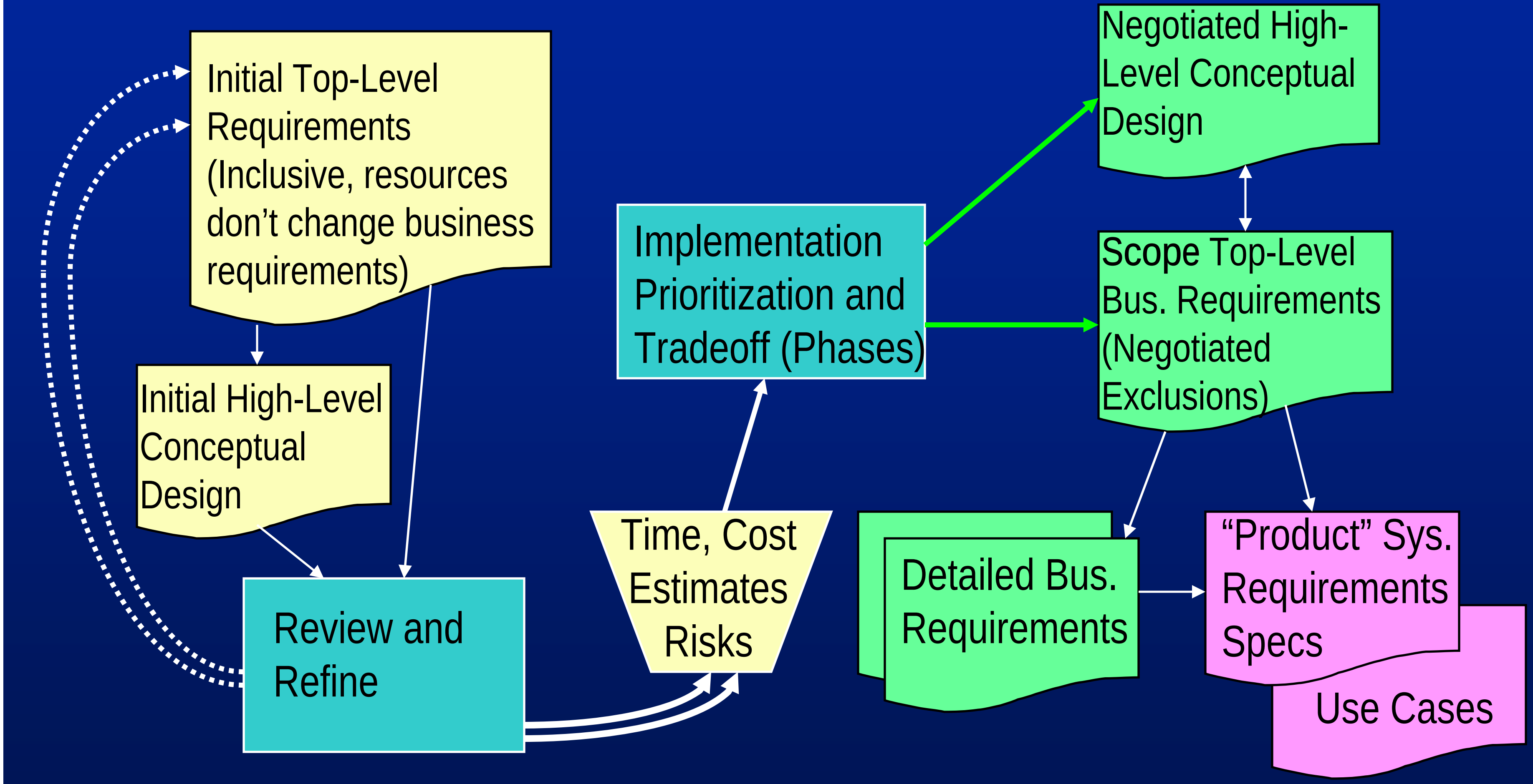
REAL Business/User Requirements

Compare to Use Cases/Func Specs 5 of 5

5. People can easily retrieve relevant reuse items.
   - A. Individuals can retrieve items themselves.
     - 1) Using their existing readily available means.
     - 2) With additionally-provided assistance.
       - a. Hardware and/or software.
       - b. HELP and documentation information.
       - c. Knowledgeable humans.
   - B. Alternate retrieval means are provided as needed.
   - C. Notification of modifications to repository items is provided to those who have reused the items.

Requirements Negotiation

Implementation Project Scope



Initial Top-Level Requirements (Inclusive, resources don't change business requirements)

Initial High-Level Conceptual Design

Review and Refine

Time, Cost Estimates Risks

Implementation Prioritization and Tradeoff (Phases)

Negotiated High-Level Conceptual Design

Scope Top-Level Bus. Requirements (Negotiated Exclusions)

Detailed Bus. Requirements

"Product" Sys. Requirements Specs

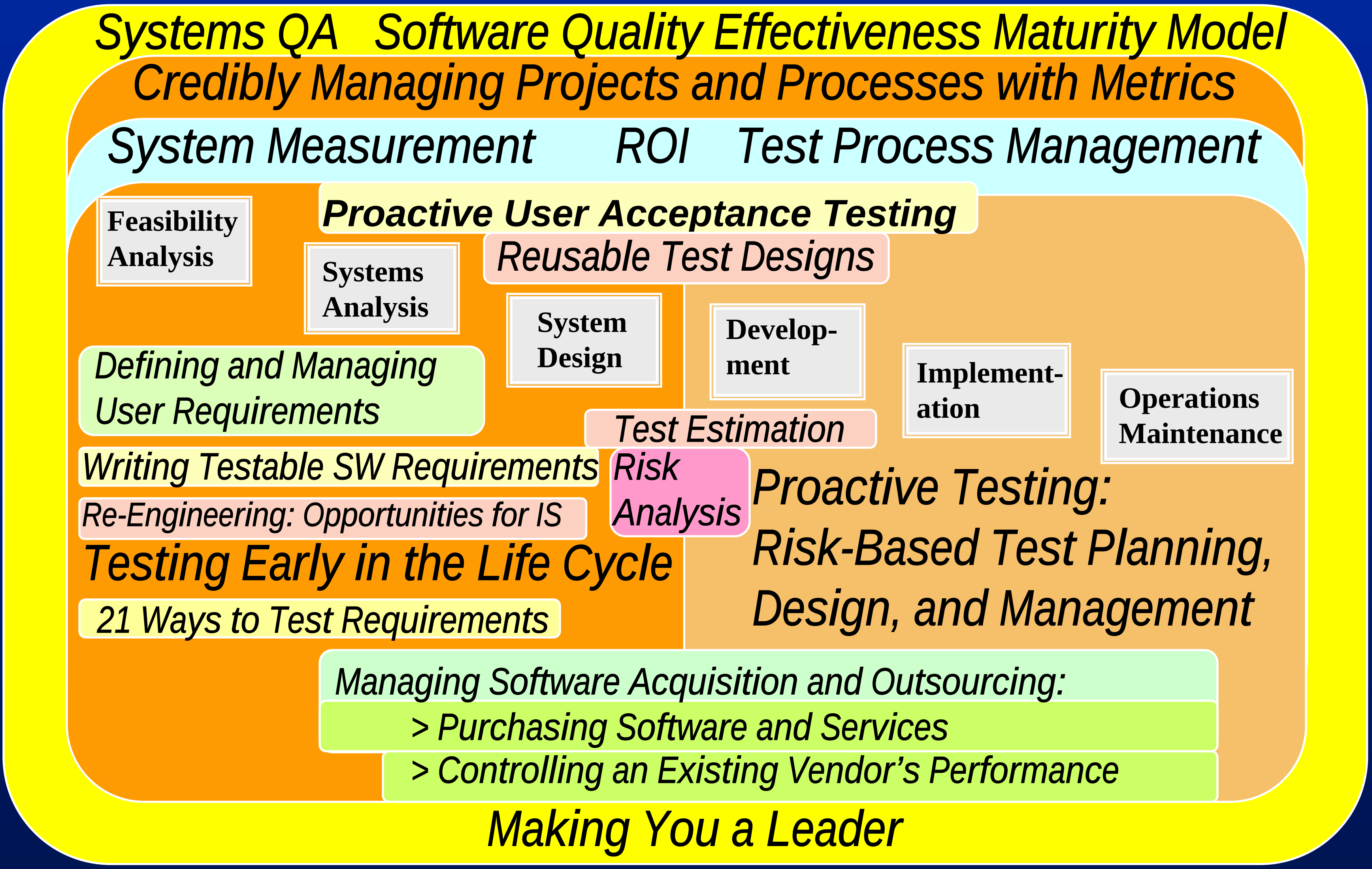
Use Cases

Requirements Scope: Problem Pyramid™

Summary

- Requirements creep mainly because system/software requirements (functional specification) and use cases for the expected *product* don’t meet the REAL, business/user requirements--*whats* that provide value when delivered.
- The Problem Pyramid™ reliably guides identifying the REAL problem and business/user requirements—**scope of the requirements**, is not a function of resources available.
- Follow 7 guidelines for documenting inclusive business/ user requirements; supplement with high-level conceptual design, then negotiate exclusions--**scope to implement.**

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



Systems QA   Software Quality Effectiveness Maturity Model

Credibly Managing Projects and Processes with Metrics

System Measurement   ROI   Test Process Management

Feasibility Analysis

Proactive User Acceptance Testing

Systems Analysis

Reusable Test Designs

System Design

Develop-ment

Defining and Managing User Requirements

Implement-ation

Operations Maintenance

Test Estimation

Writing Testable SW Requirements

Risk Analysis

Re-Engineering: Opportunities for IS

Proactive Testing: Risk-Based Test Planning, Design, and Management

Testing Early in the Life Cycle

21 Ways to Test Requirements

Managing Software Acquisition and Outsourcing:

> Purchasing Software and Services

> Controlling an Existing Vendor's Performance

Making You a Leader

Robin F. Goldsmith, JD

robin@gopromanagement.com (781) 444-5753

- 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.
- Partner with ProveIT.net on ROI Value Modeling™ and REAL ROI™.
- 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 International Institute of Software Testing (IIST) and International Institute for Software Process (IISP) Body of Knowledge Advisory Boards..
- International Institute for Business Analysis (IIBA) Business Analysis Body of Knowledge (BABOK) subject expert.
- Member IEEE Std. 829-2008 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*