A "Whole Product" Approach to Requirements Identification

Lewis Gray, PhD, PMP

Abelia Corporation
www.abelia.com

ABELIA

Your Background?

Have you heard of a “whole product” before?

Do you do requirements identification?

- **requirements elicitation (discovering)**
- **requirements analysis (clarifying, evaluating, refining)**
- **requirements specification (documenting)**
- **requirements verification (confirming).**

ABELIA

The paper handout that you have

Your handout covers similar points and lists most of the same references as my slides today. But, my thoughts about the topic today have evolved since the handout was finished, so the content of my slides has changed.

You can download the slides for today from www.abelia.com/docs/stc_03.pdf. (Or, you can get a copy from the conference).

Groundhog Day (the movie)

Bill Murray is “Phil the weatherman” at WPBH TV in Pittsburgh, PA.

Andie MacDowell is “Rita,” his Producer at WPBH.

Phil goes on assignment to Punxsutawney, PA to report on Groundhog Day.

Groundhog Day (cont'd)

In Punxsutawney, Phil is trapped in February 2. At 6:00 AM each morning, he relives the same Groundhog Day.

ABELIA

Groundhog Day (cont'd)

Phil goes crazy: he's desperate to get out of February 2. He does everything he can think of to break the endless cycle of Groundhog Days.

He smashes his radio.

He kills the groundhog.

He kills himself, several different ways.

ABELIA

Groundhog Day (cont'd)

But, nothing that Phil does makes any difference.

Until, …
Phil changes his approach to life. That is the solution. After years of repeating February 2, the cycle of Groundhog Days ends.

(And Phil and Rita get together.)

Groundhog Day: The Lesson

The old Phil approached each day in his life in the same counterproductive way, and the result was that each day turned out badly.

When Phil changed his approach to life, that solved his problem.

So, the lesson is that Phil caused his own problem (even though he blamed all his problems on the "morons" around him).

Phil took many, many years to learn this.

ABELIA

Consider Requirements Identification

Poor requirements were a major cause of software acquisition failures in the 60's,

…and in the 70's, 80's, and 90's,

…and they still are.

And, on the whole, we still approach requirements identification in the same way. So, acquisition failures happen again and again.

Déjà vu, it's Phil the weatherman all over again.

ABELIA

Requirements Identification Lesson

It’s time to change our approach.

To avoid software acquisition failures, we’re going to have to change our goals for requirements identification.

ABELIA

Who Sets the Goals?

Acquirers

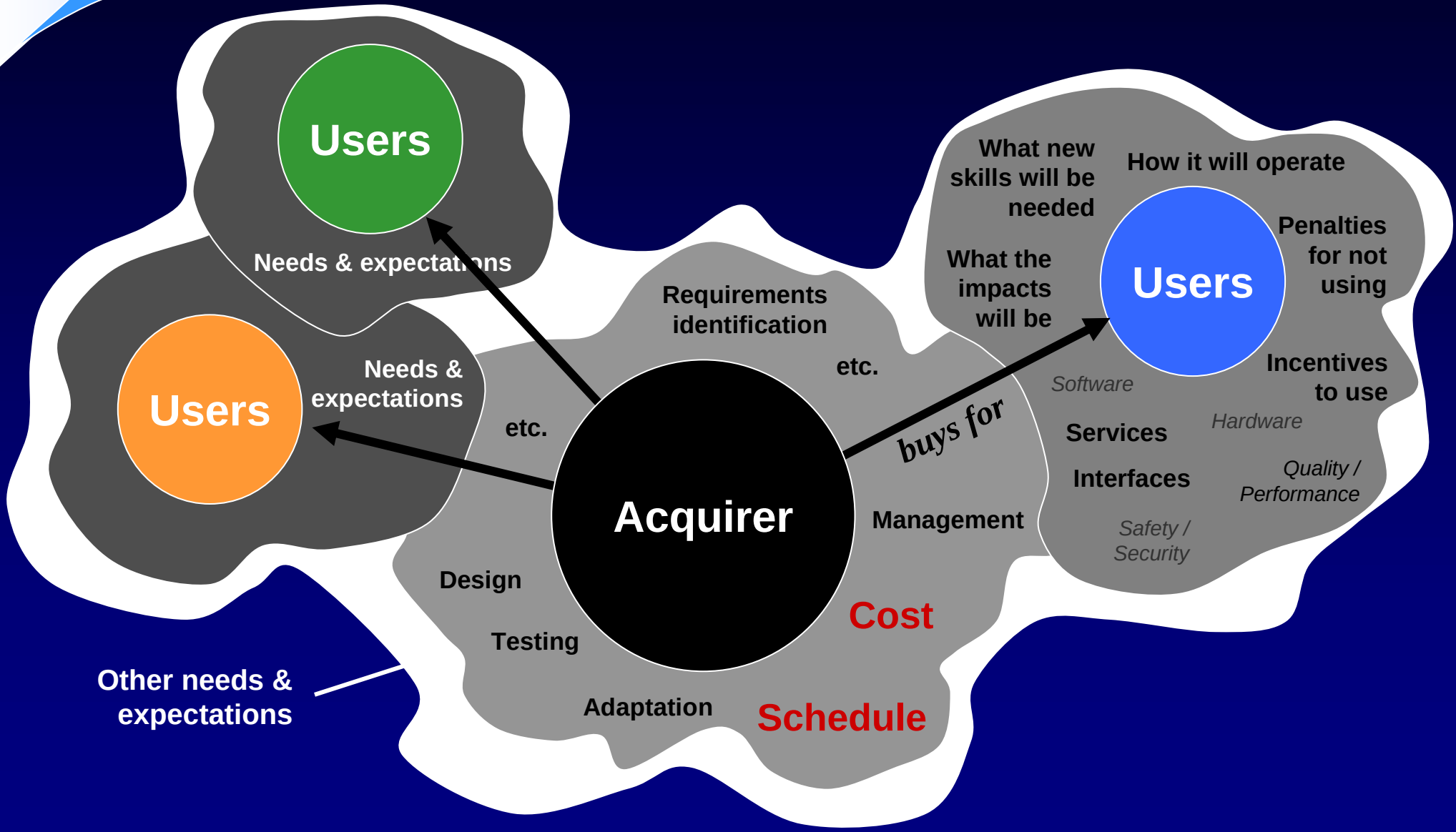
- arrange for requirements identification
- set each supplier's goals for requirements identification
- define the overall approach to requirements identification
- approve each supplier's requirements related deliverables.

ABELIA

How are Requirements Identified?

The acquirer attempts to develop (often they contract for) descriptions of the needs and the expectations of stakeholders that will be conditions for acceptance of the software product.

Many Needs and Expectations Must Be Captured



ABELIA

To Document Requirements…

Concept of operations description (COD)
(e.g., Operational Concept Description (OCD): DI-IPSC-81430A)

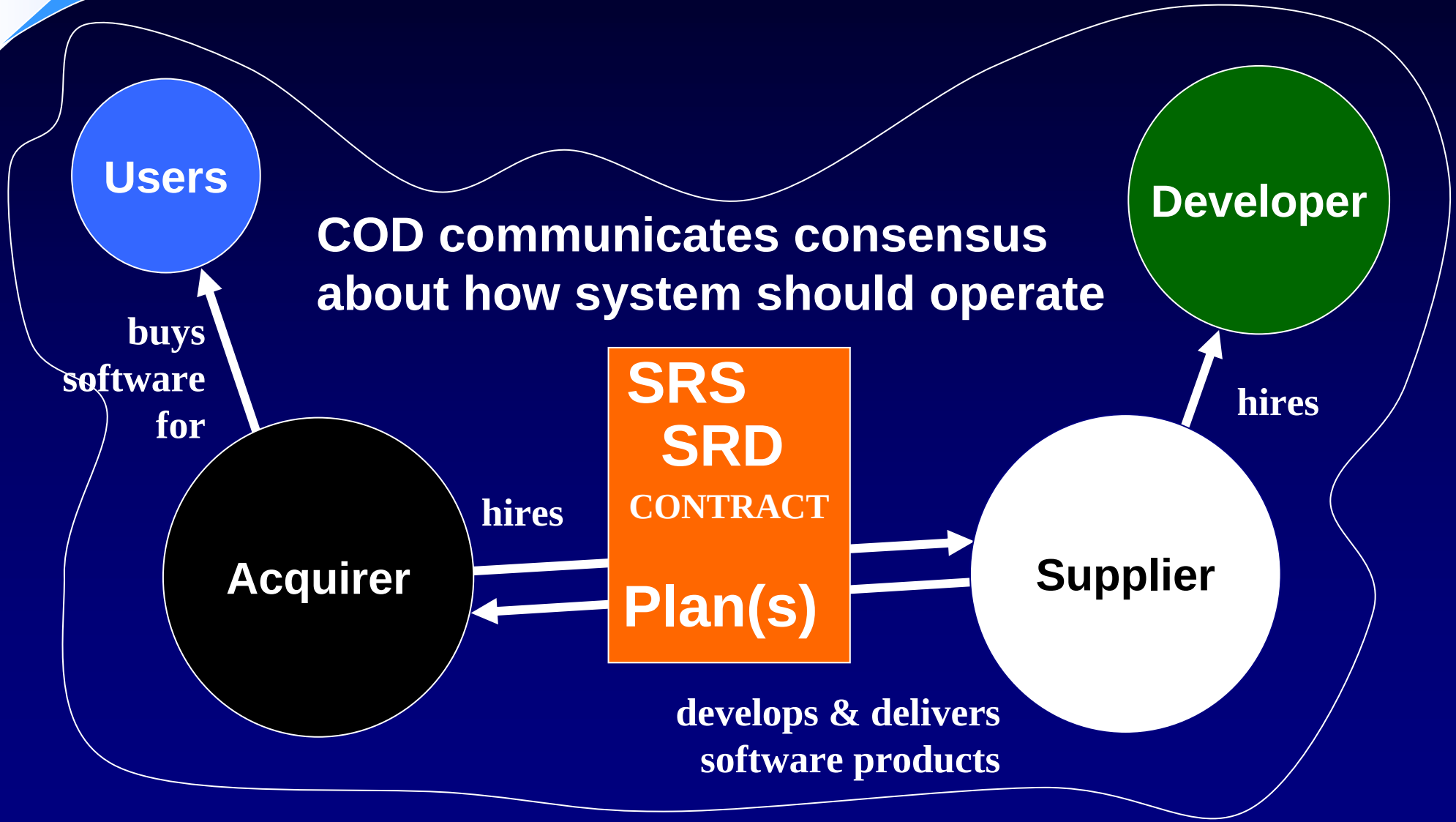
System requirements specification (SRS)
(e.g., System/Subsystem Specification (SSS): DI-IPSC-81431A)

Software requirements description (SRD)
(e.g., Software Requirements Specification (SRS): DI-IPSC-81433A)

Project management plans (if part of contract)
(e.g., Software Project Management Plan: IEEE Std 1058
Software Development Plan (SDP): DI-IPSC-81427A)

Contract (if plans are not part of the contract).

How Needs and Expectations Influence Software Acquisitions



ABELIA

Evaluation of the Usual Practices

The usual requirements identification practices are useful for writing an enforceable contract between an acquirer and a supplier:

- **"We will pay you if you deliver what we require you to deliver. We will penalize you if you don't."**

They are effective at providing the information that they are intended to provide about software products.

ABELIA

Much Guidance is Available

High Sales Rank and high Best-seller Rank at Amazon.com (4 March 03)

- ***Software Requirements*, by Karl E. Wiegers [Wie99]**
- ***Mastering the Requirements Process*, by Suzanne Robertson, James Robertson [Rob99]**
- ***Exploring Requirements: Quality Before Design*, by Donald C. Gause, Gerald M. Weinberg [Gau89]**
- ***Requirements Analysis: From Business Views to Architecture*, by David C. Hay [Hay03]**

So, Why Do We Have Problems with Software Acquisitions?

Because we have the wrong view of requirements.

As a result, we always ignore some very important requirements.

And, worse, our usual methods and practices for identifying requirements aren't good for identifying the requirements that we ignore.

ABELIA

The Usual View of Requirements

"A requirement is something a product must do or a quality that a product must have." [Rob99]

ABELIA

A Better View of Requirements

A requirement is what a stakeholder needs or expects.

A requirement is a, “need or expectation that is stated, generally implied* or obligatory.” [ISO00]

(*Generally implied: “means that it is custom or common practice for the organization, its customers and other interested parties, that the need or expectation under consideration is implied.” [ISO00])

ABELIA

What Requirements Do We Ignore?

We ignore

- The incentives that are required for users to adopt and use the newly acquired software, and

- The penalties that we must invoke against users who attempt to avoid the new software.

Neither of these is something a product must do, or a quality that a product must have.

Neither of these is captured by our usual documents (COD, SRS, SRD, plans, contract, etc.).

ABELIA

What's Wrong with Usual Methods?

Our usual practices for eliciting requirements depend heavily on stakeholders telling you what the requirements are: e.g.,

- Identify use cases, hold JAD sessions, define quality attributes, prioritize requirements, write test cases from requirements, define acceptance criteria. [Wie99]

But, stakeholder testimony isn't a good guide to the incentives and the penalties. Usually, it isn't helpful to ask a user:

- "How could we entice you to use this new software?"
- "How could we force you to use this new software?"

ABELIA

A Different Approach

For many software acquisitions, a software product from a supplier will be only part of what the acquirer should deliver to the users.

Rather than limiting ourselves to software products, as we do now, we should identify the requirements for the whole product. This change of approach will reduce acquisition risk by focusing directly on software acquisition success.

(Process) What's a Whole Product?

In software process terms, a whole product is what it takes to create and sustain an organizational culture that institutionalizes a process (where people believe, "That's the way we do things around here.").

SEI lists:

- *External pressures* - such as customer needs, changing technology, and competition
- *Infrastructure* - such as policies, standards, procedures, training, oversight, reviews, and audits
- *Internal factors* - such as champions, sponsors, competence, skills, abilities, and tools.

ABELIA

Examples

Every organization with system/software processes at CMM® /CMMI® maturity level 3 and higher has demonstrated to a lead appraiser that it has a process whole product that includes:

- **the organization's set of standard processes (OSSP)**
- **plus everything it takes to institutionalize the OSSP: e.g.,**
  - **Policies and plans**
  - **Adequate resources**
  - **Responsibilities and sponsorship**
  - **A training capability, training resources and records**
  - **Involved stakeholders**
  - **A process database and other process assets**
  - **Internal reviews / audits**
  - **High-level management review.**

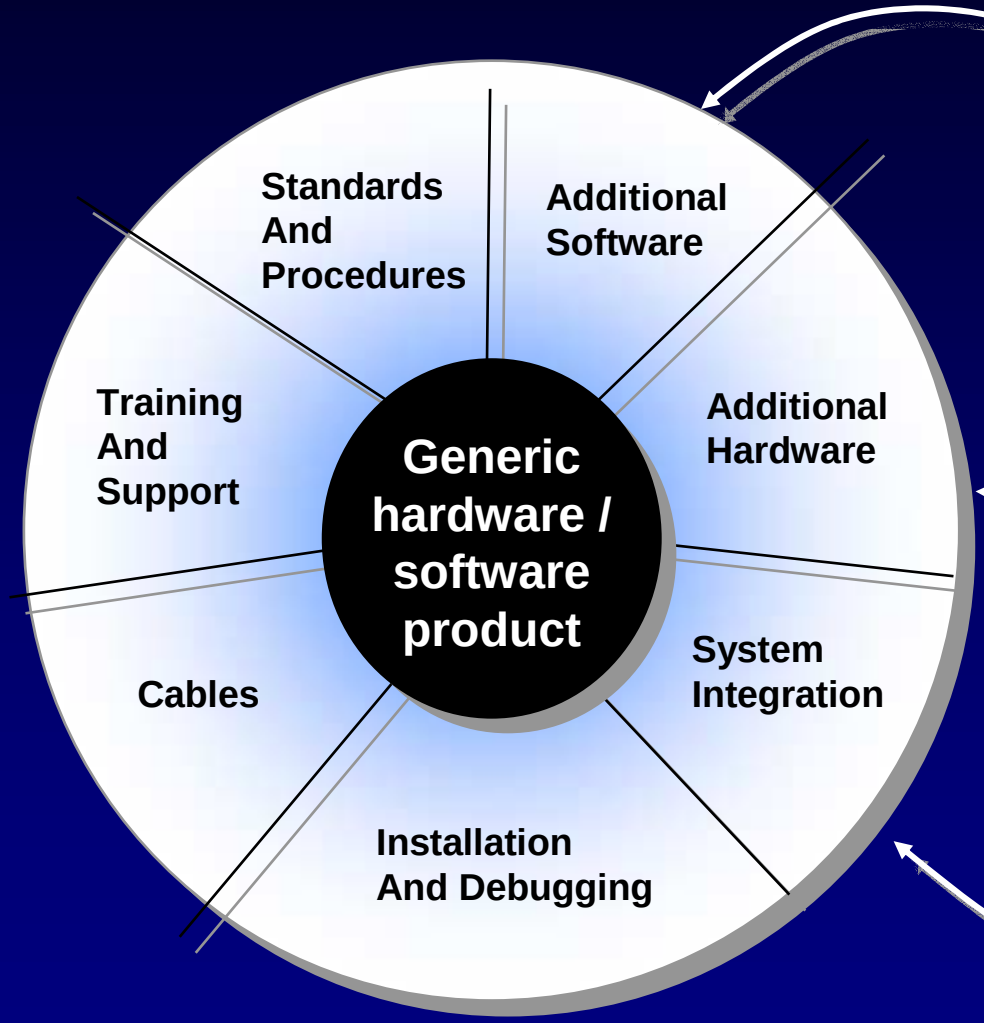
(Marketing) What's a Whole Product? [Mo99]

Background: With high-tech products, usually there is, "a gap between the marketing promise made to the customer – the compelling value proposition – and the ability of the shipped product to fulfill that promise." I.e., the shipped product isn't everything the customer expected.

This gap means that customers either acquire whatever additional products and services they need to use the product as they expected, or the customers feel cheated.

Definition: In marketing terms, a whole product, "is the minimum set of products and services needed to fulfill the compelling reason to buy for the target customer." So, it is everything the customer expected – there's no gap.

(Marketing) Simplified Whole Product Model*



Anything else the customer would need to achieve their compelling reason to buy

*[Mo99]

Savi Technology

www.savi.com

ABELIA

Savi's Whole Product*

Need: Real-time inventory tracking. Find a specific inventory item right now in a shipping yard full of closed containers and closed trailers with mixed contents.

Savi Technology's solution is radio frequency tags and interrogation devices: it was used successfully by the DOD in Bosnia.

Savi's whole product for developing a commercial market:

- Savi products
  - Gatemaster, Yardmaster, Dockmaster, passive RFID tags, handheld terminals, Asset Manager.
- Savi services
  - Site surveys and system integration, alarms and custom user interface software
- Non-Savi products and services (supplied by Savi partners)
  - Wintel PC-based server, inventory management systems, business process reengineering consulting, training, sales and service.

*[Mo99]

ABELIA

(Software Acquisition) What's a Whole Product?

In software acquisition, a whole product is what it takes to persuade a user

- **to accept the software product that has been acquired**
- **to institutionalize the software product as part of, "the way we do things around here".**

A whole product for software acquisition combines a whole product for marketing and a process whole product.

ABELIA

Getting the Missing Requirements

The usual requirements documents (COD, SRS, SRD, plans, contract, etc.) tell us only part of what we need to know about a whole product for software acquisition.

In software engineering, we don't have techniques for identifying the missing requirements (which are related to incentives and penalties).

Here are three sources of techniques:

- Technology diffusion research
- Marketing research
- Social epidemics (trends) studies.

ABELIA

From Technology Diffusion Research

[Rog95] No matter how willing or unwilling different individuals within a community of users may be to adopt newly acquired software, they fall into only five ideal adopter categories:

- **Innovator – truly different software is its own incentive.**
- **Early adopter – a vision of how the software can achieve big rewards is their incentive**
- **Early majority – they take deliberate initiative, seeking incremental improvement**
- **Late majority – their incentive is pressure from norms and peers**
- **Laggard – once change is inevitable, their incentive is to change so as to deviate as little as possible from tradition.**

No more than these five kinds or incentives should be considered.

ABELIA

From Marketing Research [Mo99]

Find pragmatic incentives and penalties for pragmatic users (early majority). Do not expect pragmatic users to take risks. Do not use visionaries (early adopters) as examples:

- **Visionaries want to be the first to implement a change, and they'll bear the costs of bugs and other difficulties along the way, and they'll bear the risk of failure**
- **Pragmatists want to improve performance of existing operations while minimizing change.**

When identifying incentives for pragmatists, look for symbolic, small, niche tasks where performance can be improved with new software with little risk.

ABELIA

Marketing Research [Goe02]

Ensure that users understand the emotional benefits of the functional benefits that the new software will provide: e.g.,

- **Functional benefit – "I ache…I take Bayer…I don't ache."**
- **Emotional benefit – Once the pain is gone, you'll be calmer, better with your colleagues and your kids, you'll have more confidence.**
- **Universal emotional benefits – confidence, comfort.**

ABELIA

From Social Epidemics Study [Gla02]

Find symbolic “broken windows” and build the new software to fix them.

Broken Window Theory: “If a window is broken and left unrepaired, people walking by will conclude that no one cares and no one is in charge. Soon, more windows will be broken, and the sense of anarchy will spread from the building to the street it faces, sending a signal that anything goes.”

Example: 1984 – 1996 in New York City, Bernhard Goetz and symbolic “broken windows”:

- graffiti on New York subway cars
- fare-beating at New York subway stations.

ABELIA

Social Epidemics (cont'd) [Gla02]

Seek out and use 'connectors' (Paul Revere) and 'mavens' (Paul Revere) to spread the message, and 'salesmen' to deliver the message persuasively.

Make the message memorable by

- **making it practical and personal**
- **making the user a participant (Columbia Record Club "Gold Box")**
- **testing and refining it (*Sesame Street*, *Blue's Clues*).**

ABELIA

Are These Techniques Scientific?

They’re at least as “scientific” as the requirements identification techniques that we use typically.

ABELIA

Breaking Out of Failure

Phil (Conners) the weatherman seemed trapped in endless February 2's in *Groundhog Day*.

Software acquirers seem to be caught in their own endless cycle of acquisition failures where software is developed and thrown away.

Poor software requirements are said to be a major cause of the problem.

ABELIA

Breaking Out of Failure (cont'd)

Our usual approach to requirements identification focuses on software product requirements. What we do here we do well. But it isn't enough. Users reject good software too.

To break the cycle of software acquisition failure, we must extend our requirements identification approach beyond the software product. We must identify incentives and penalties too.

ABELIA

Breaking Out of Failure (cont'd)

We must identify and deliver a whole product to users that provides a compelling reason for them to adopt and use the good software that we develop.

Questions

Contact Information

Lewis Gray
Abelia Corporation
Fairfax, VA 22033
www.abelia.com

lewis@abelia.com
703-591-5247

ABELIA

Sources

[DOR] Dorfman, Merlin, "Requirements Engineering," at SEI web site (http://interactive.sei.cmu.edu/Features/1999/March/Background/Background.mar99.htm), excerpt from *Software Requirements Engineering,* 2nd Ed., Thayer, Richard H. & Merlin Dorfman Eds. (IEEE Computer Society: Los Alamitos, CA 1997)

[Gau89] Gause, Donald C., and Gerald M. Weinberg, *Exploring Requirements: Quality Before Design*, (Dorset House: NY 1989)

[Gla02] Gladwell, Malcolm, *The Tipping Point*, (Back Bay/Little, Brown: Boston 2002)

[Goe02] Goebert, Bonnie with Herma M. Rosenthal, *Beyond Listening*, (John Wiley: New York 2002)

[Hay03] Hay, David C., *Requirements Analysis: From Business Views to Architecture*, (Prentice Hall, PTR: Upper Saddle, NJ 2003)

ABELIA

Sources (cont'd)

[IEEE] IEEE/EIA 12207 Industry Implementation of International Standard ISO/IEC 12207: 1995, Standard for Information Technology, Software Life Cycle Processes, (IEEE: NY 1998)

[ISO00] ISO 9000 Quality Management Systems – Fundamentals and Vocabulary, (ISO: Geneva 2000)

[LEI02] Leishman, Theron R., & Dr. David A. Cook, "Requirements Risks Can Drown Software Projects," in *CrossTALK*, (Software Technology Suport Center: Hill AFB, UT) April 2002

[Mo99] Moore, Geoffrey A., *Crossing the Chasm*, Revised Ed. (HarperBusiness: New York 1999)

[Rob99] Robertson, Suzanne & James, *Mastering the Requirements Process*, (ACM Press/Addison-Wesley: Harlow, England 1999)

[Rog95] Rogers, Everett M., *Diffusion of Innovations*, 4th Ed. (Free Press: New York 1995)

[Wie99] Wiegers, Karl E., *Software Requirements,* (Microsoft: Redmond, WA 1999)

ABELIA

Acronyms

CMM®	-	Capability Maturity Model (SEI)
CMMI®	-	Capability Maturity Model Integration (SEI)
COD	-	Concept of Operations Description (IEEE/EIA 12207)
DOD	-	Department of Defense
EIA	-	Electronic Industries Alliance
IEC	-	International Electrotechnical Commission
IEEE	-	Institute of Electrical and Electronics Engineers
ISO	-	International Organization for Standardization
JAD	-	Joint Application Development
PC	-	Personal Computer
PMI	-	Project Management Institute
PMP	-	Project Management Professional (PMI)
OCD	-	Operational Concept Description
OSSP	-	Organization's Standard Software Process (CMM)
	-	Organization's Set of Standard Processes (CMMI)
RFID	-	Radio Frequency Interrogation Device
SDP	-	Software Development Plan
SEI	-	Software Engineering Institute
SRD	-	Software Requirements Description (IEEE/EIA 12207)
SRS	-	Software Requirements Specification
	-	System Requirements Specification (IEEE/EIA 12207)
SSS	-	System/Subsystem Specification