



Early Warning Indicators in the Acquisition of Software-Intensive Systems

Barry Boehm, USC
Keynote Address

Third Annual Conference on the Acquisition of
Software-Intensive Systems

January 27, 2004
(boehm@sunset.usc.edu)
(<http://sunset.usc.edu>)



Outline

- **Trends in Defense Software-Intensive Systems**
- **Need for New Acquisition Approaches**
- **Trilateral SIS Working Group Effort to Develop SIS Acquisition Early Warning Indicators**
- **Current Goal/Critical Success Factor Framework**
- **Community Opportunity to Improve Framework**



Trends in Defense Software-Intensive Systems

- **Transformational, network-centric systems**
 - These are fundamentally software-intensive
- **Emphasis on joint, interoperable, capability-based systems**
 - And increasingly, systems of systems
- **Increasing requirements emergence, COTS-dependence, environmental change**
- **Traditional sequential acquisition practices increasingly inadequate**
 - Fixed-requirements, -cost, -schedule contracting
 - Waterfall legacies: MIL-STD-1521B, parts of Software CMM



Waterfall Legacies: SW CMM v.1.1

- **Requirements Management, Ability 1:**
“Analysis and allocation of the system requirements

is not the responsibility of the software engineering group

but is a prerequisite for their work.”



Defense Management Evolving New Acquisition Approaches

- **US: 5000.1 and 5000.2, CMMI, Acquisition CMM, evolutionary acquisition**
- **UK: DPA Stocktake Initiative**
- **Australia: Software Material Reform Program, Capability Systems Life Cycle**



Trilateral Working Group Addressing Early Warning Indicators

- **US, UK, and Australian participants**
- **Mix of traditional and emerging indicators**
- **Current version a work in progress; seeking community feedback**
- **Organized around program goals and critical success factors**
- **Focused on readiness for milestone decision reviews**



Trilateral Working Group

- **Matt Ashford, AUS MOD**
- **Shonnag Allison, UK DPA**
- **Barry Boehm, USC**
- **Julien Burridge, UK DPA**
- **Kathleen Dangle, Fraunhofer Center – Maryland**
- **Brian Gallagher, SEI**
- **Jim Linnehan, USA(ALT)**
- **Peter Nolte, OUSD(AT&L)/DS**
- **Don Reifer, USC**
- **Richard Turner, GWU**



Current Goal/Critical Success Factor Framework

- **Counterexamples of what happens if factors not addressed**
- **Examples of more detailed checklists from current working group material**
- **Examples of potential early warning indicator tracking system displays.**



Software Acquisition Goals

- **Goal 1. System and software objectives and constraints have been adequately defined and validated.**
- **Goal 2. The system and software acquisition strategies are appropriate and compatible.**
- **Goal 3. The success-critical stakeholders have committed adequate software capability to perform their software-related tasks.**
- **Goal 4. The software product plans and process plans are feasible and compatible. A Feasibility Rationale provides convincing evidence that:**
- **Goal 5. Software progress with respect to plans is satisfactory.**



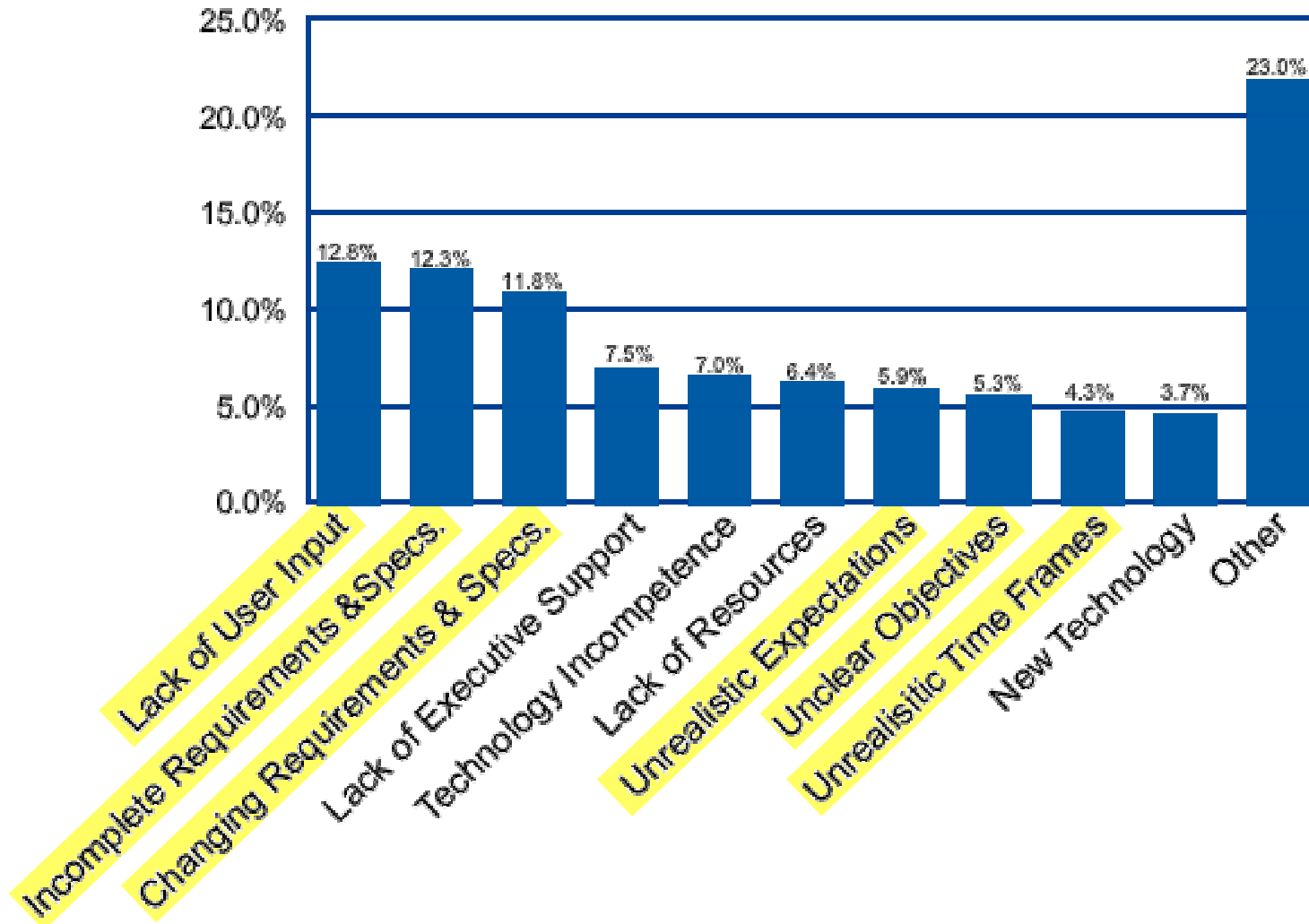
Critical Success Factors: Goal 1

Goal 1. System and software objectives and constraints have been adequately defined and validated.

- 1.1 System and software functionality and performance objectives have been defined and prioritized.**
- 1.2 The system boundary, operational environment, and system and software interface objectives have been defined.**
- 1.3 System and software flexibility and evolvability objectives have been defined and prioritized.**
- 1.4 System and software environmental, resource, infrastructure, and policy constraints have been defined.**
- 1.5 System and software objectives have been validated for overall achievability within the system and software constraints.**



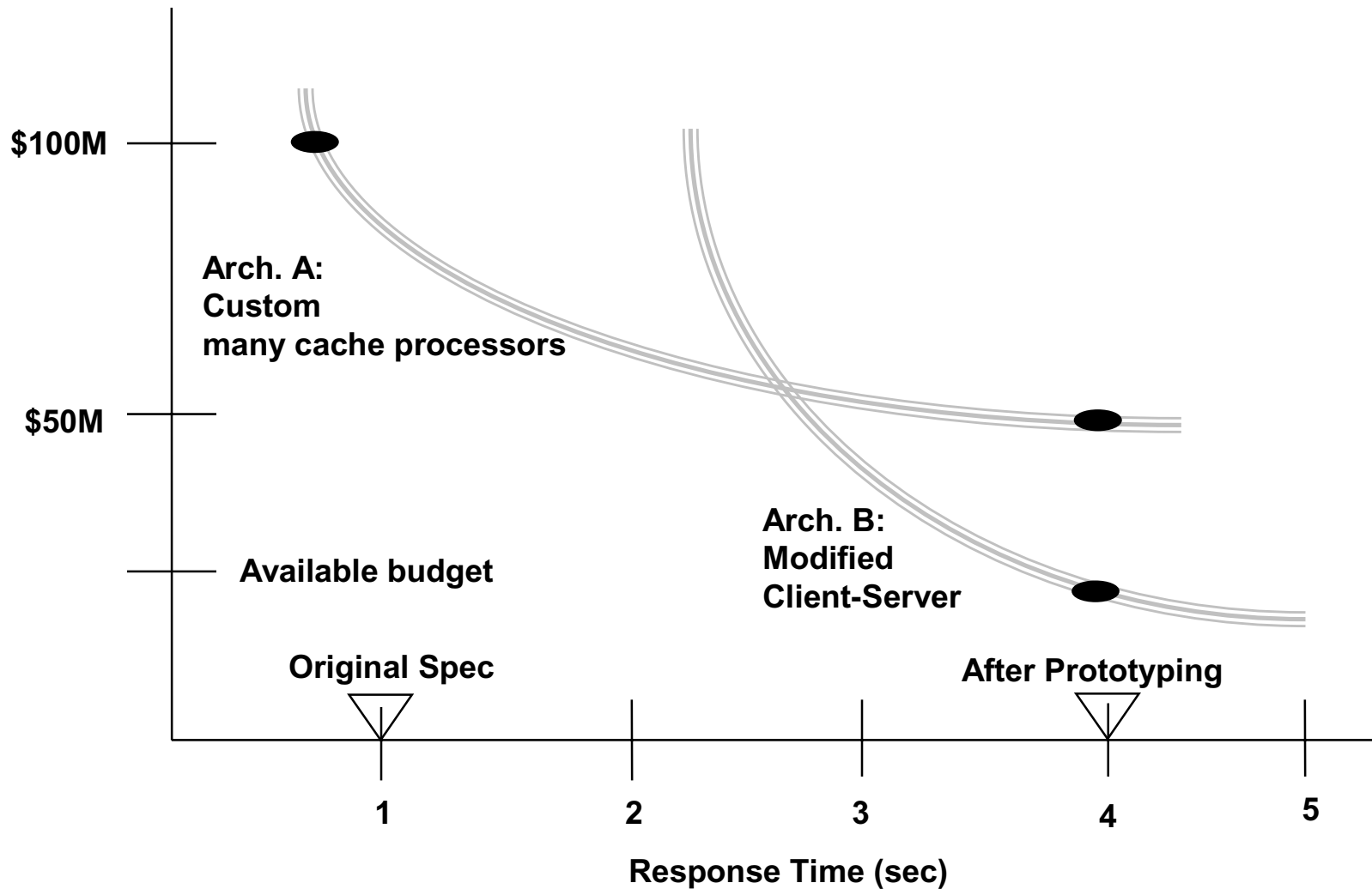
Why Software Projects Fail



352 companies - 8,000 software projects. Source: *The Standish Group, 1995*



Effect of Unvalidated Requirements





Effect of Unvalidated Software Schedules

- **Original goal: 18,000 KSLOC in 7 years**
 - Initial COCOMO II, SEER runs showed infeasibility
 - Estimated development schedule in months for closely coupled SW with size measured in equivalent KSLOC (thousands of source lines of code):

$$\text{Months} \approx 5 * 3 \text{ KSLOC}$$

- KSLOC	300	1000	3000	10,000
- Months	33	50	72	108

- **Solution approach; descope features; architect for decoupled parallel development**



Detailed Software Cost and Schedule Checklist

- **Were multiple independent estimation methods used?**
 - i. **Individual expert judgment**
 - ii. **Group consensus (Delphi, group meeting, etc.)**
 - iii. **Analogy to previous experience**
 - iv. **Single parametric model**
 - v. **Multiple parametric models**
- **Were the methods relevant to the project?**
- **For each method, were the inputs sufficient?**
- **For each input, is there a credible basis of estimate?**
- **Are the resulting estimates realistic?**
- **Were the inputs and estimates reviewed by experts?**
- **Are the estimates and the reviews current?**
- **Are there mechanisms to monitor and adjust assumptions and estimates?**

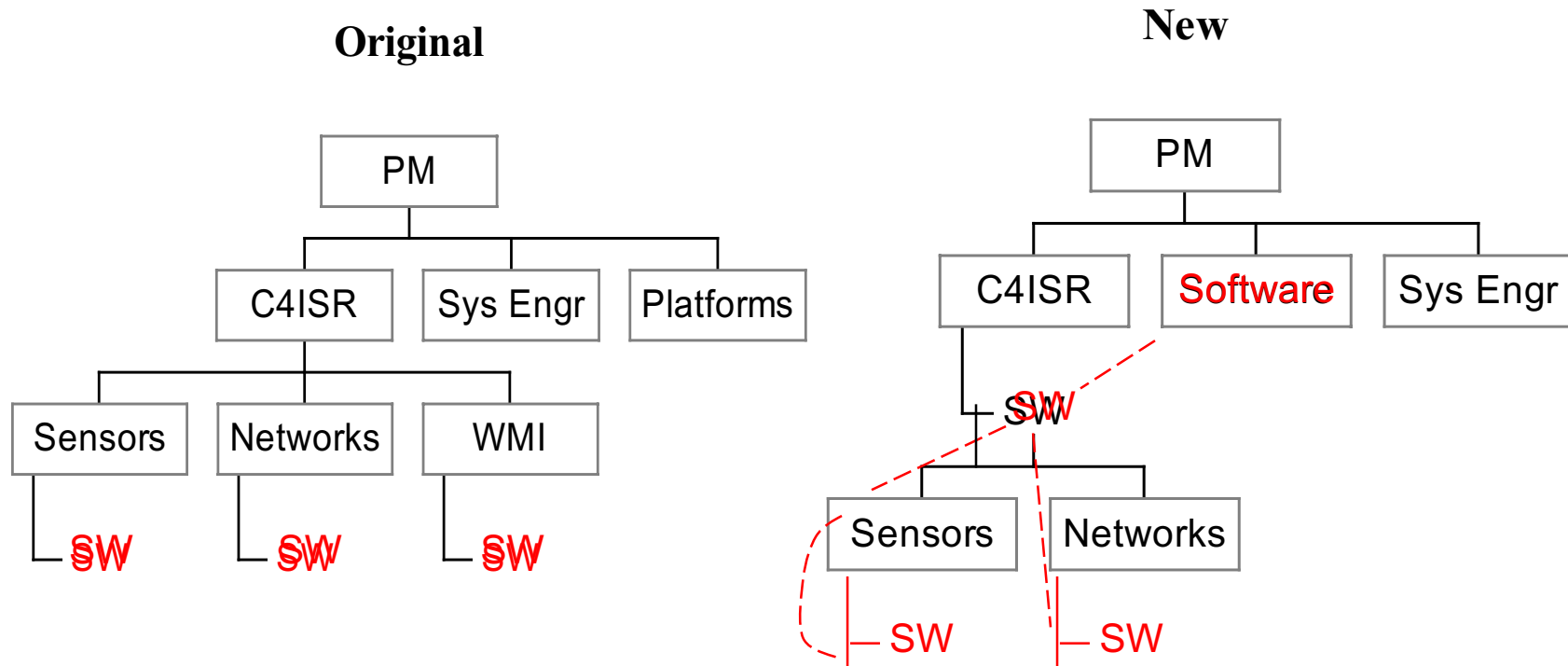


Critical Success Factors: Goal 2

- **Goal 2. The system and software acquisition strategies are appropriate and compatible.**
- **2.1 They define the acquisition life cycle, success-critical stakeholder roles and responsibilities, contracting mechanisms, and progress milestones and success criteria.**
- **2.2 They assign adequate levels of authority and responsibility for software integration and change management across the program elements, supplier chains, and external interfaces.**
- **2.3 They identify the most critical system and software risks and an effective risk management process.**
- **2.4 They are compliant with legal, policy, regulatory, standards, and security requirements.**
- **2.5 They are compatible with each other and facilitate system and software engineering concurrency, synchronization, flexibility, and stability.**

Effect of Software Underrepresentation

- Software risks discovered too late
- Slow, buggy change management
- Recent large project reorganization





Effect of Waterfall SEMP and Spiral SDP

- **Delays in starting critical software infrastructure**
 - OS, networking, DBMS, transaction processing, ...
- **Infeasible infrastructure**
 - Premature performance requirements (e.g., 1 second)
- **Premature hardware selection overconstrains software**
 - Can also induce premature COTS commitments
- **Waterfall-based progress payments undermine-spiral tasks**
 - Develop prototypes or get paid for specifications



Early Risk Resolution Quotes

**“In architecting a new software program,
all the serious mistakes are made on
the first day.”**

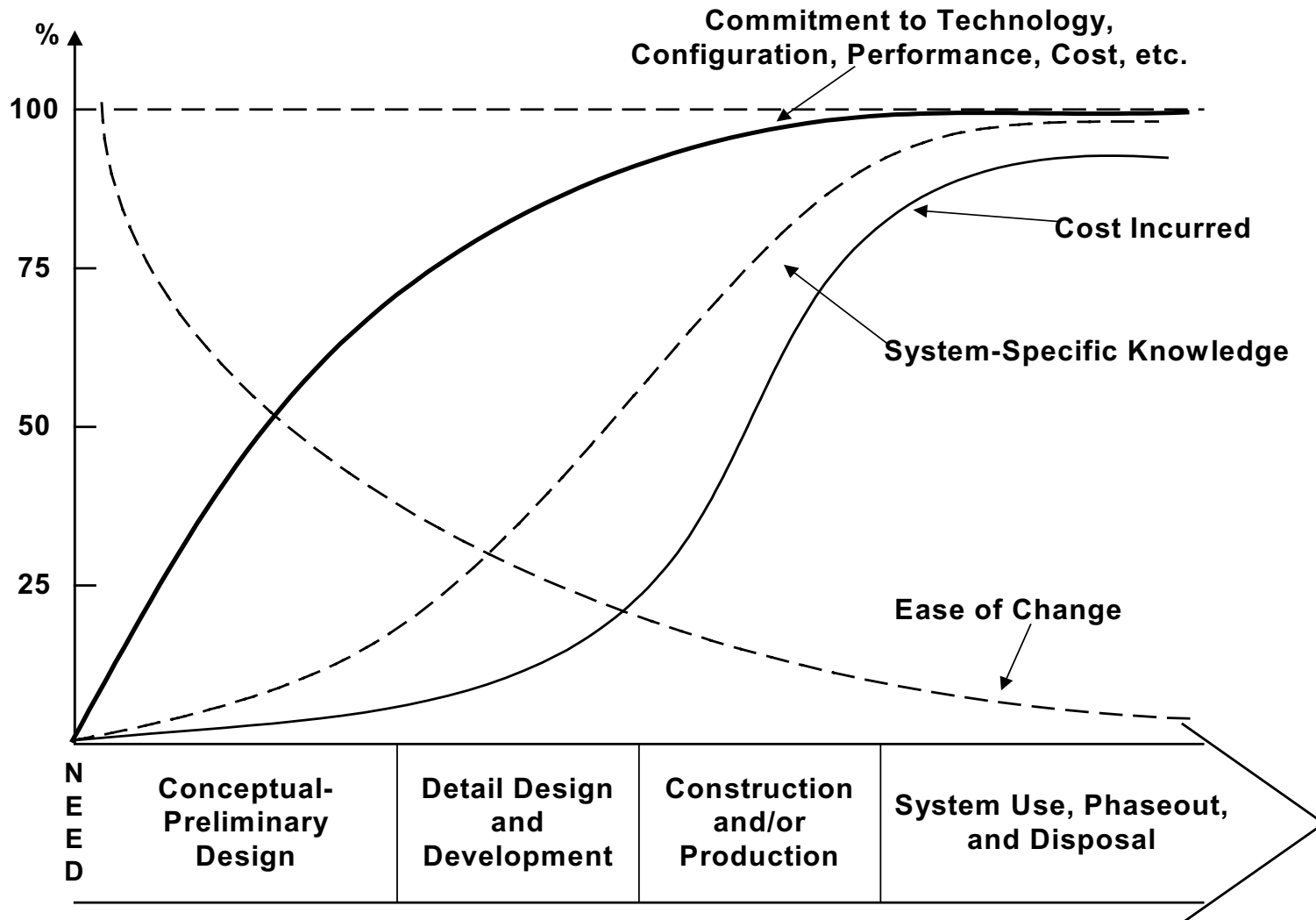
Robert Spinrad, VP-Xerox, 1988

**“If you don’t actively attack the risks,
the risks will actively attack you.”**

Tom Gilb, 1988

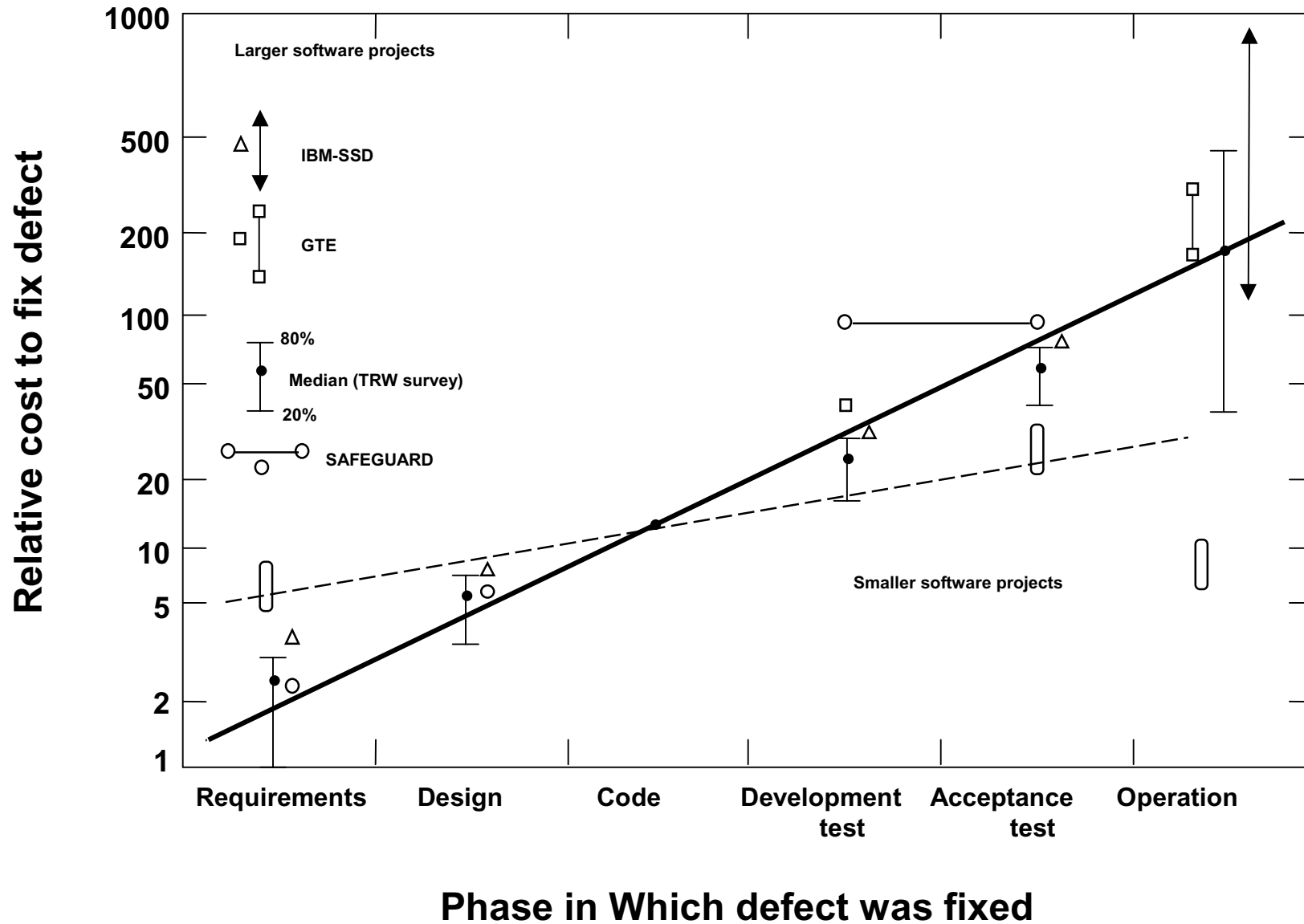
Risk of Delaying Risk Management: Systems

—Blanchard- Fabrycky, 1998



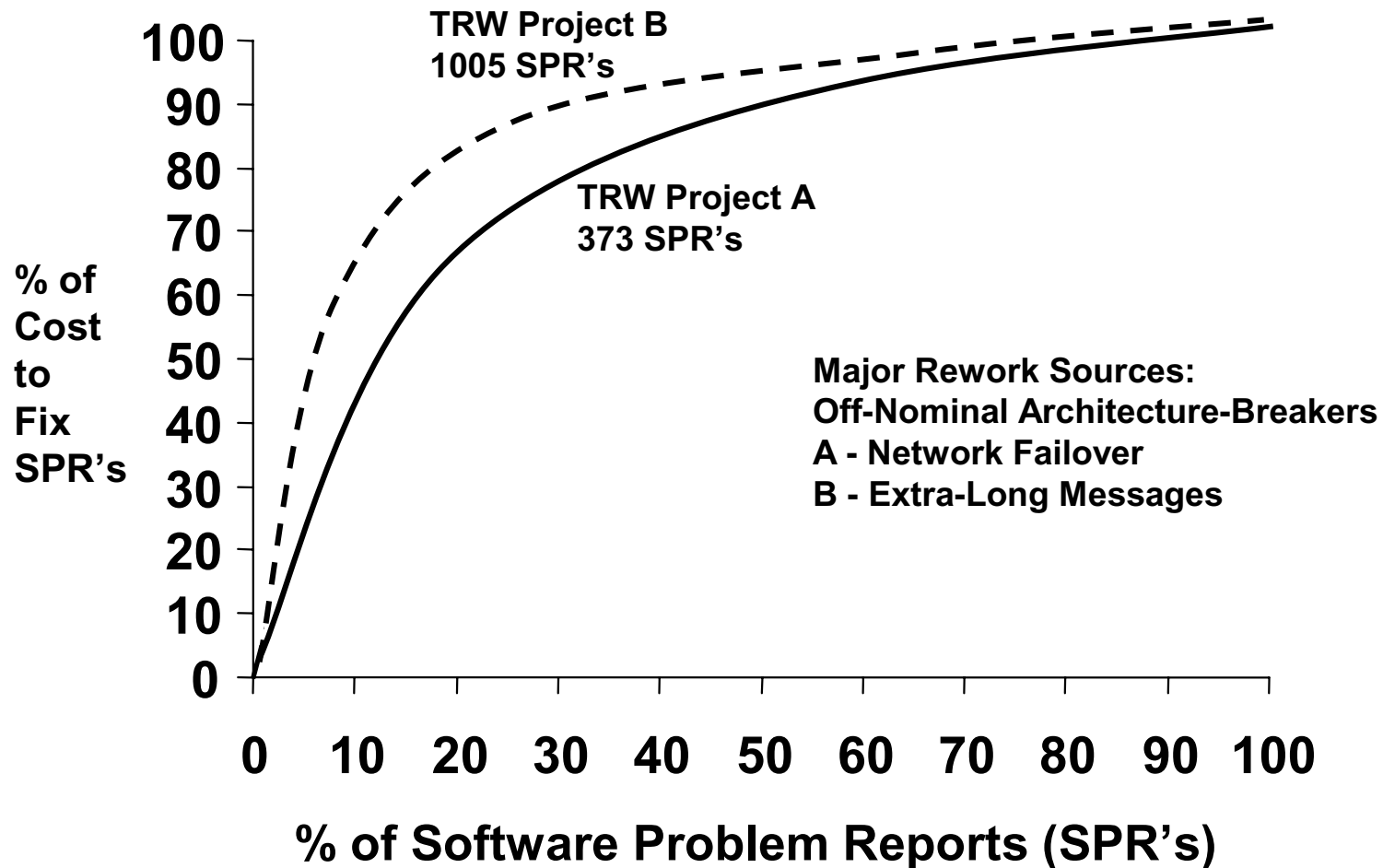


Risk of Delaying Risk Management: Software





Steeper Cost-to-fix for High-Risk Elements





Acquisition Strategy Checklist

- **Is the acquisition strategy consistent with the level of software risk on the program?**
 - i. **List of software risks (how many, how identified, what are they, breadth and depth, criticality, risk exposure)**
 - ii. **List of mitigation strategies that address software risks (number of risks addressed by strategy, available resources to perform mitigation)**
 - iii. **Map of software risks and mitigation strategies to reveal coverage**
 - iv. **Milestone decision criteria that address software risks**
 - v. **Activities performed to assure acquisition strategy is appropriate**
 - vi. **Positions of software experts within the acquisition program's organizational structure or hierarchy**
 - vii. **System engineering trade off analyses to be performed and assessment of their impact on software; software trade off analyses and when they will be performed**
- **Are qualified stakeholders involved in the preparation and execution of the acquisition strategy?**
- **Do the acquisition strategy and pre-award processes (solicitation) address software-related qualifications in selecting contractor(s)?**
- **Do the post-award processes (contract monitoring) facilitate the acquisition strategy that addresses software issues?**



Critical Success Factors: Goal 3

Goal 3. The success-critical stakeholders have committed adequate software capability to perform their software-related tasks.

- **3.1 The success-critical stakeholders have been identified and their roles and responsibilities negotiated.**
- **3.2 The stakeholders involved in software-related negotiations are represented by collaborative, representative, authorized, committed, and knowledgeable personnel.**
- **3.3 Acquisition executives and contracting personnel have adequate software acquisition experience or training.**
- **3.4 Suppliers, integrators, and acquirers have adequate software skills and process maturity for their software-related tasks.**
- **3.5 Software staffing estimates have been validated for feasibility and achievability.**

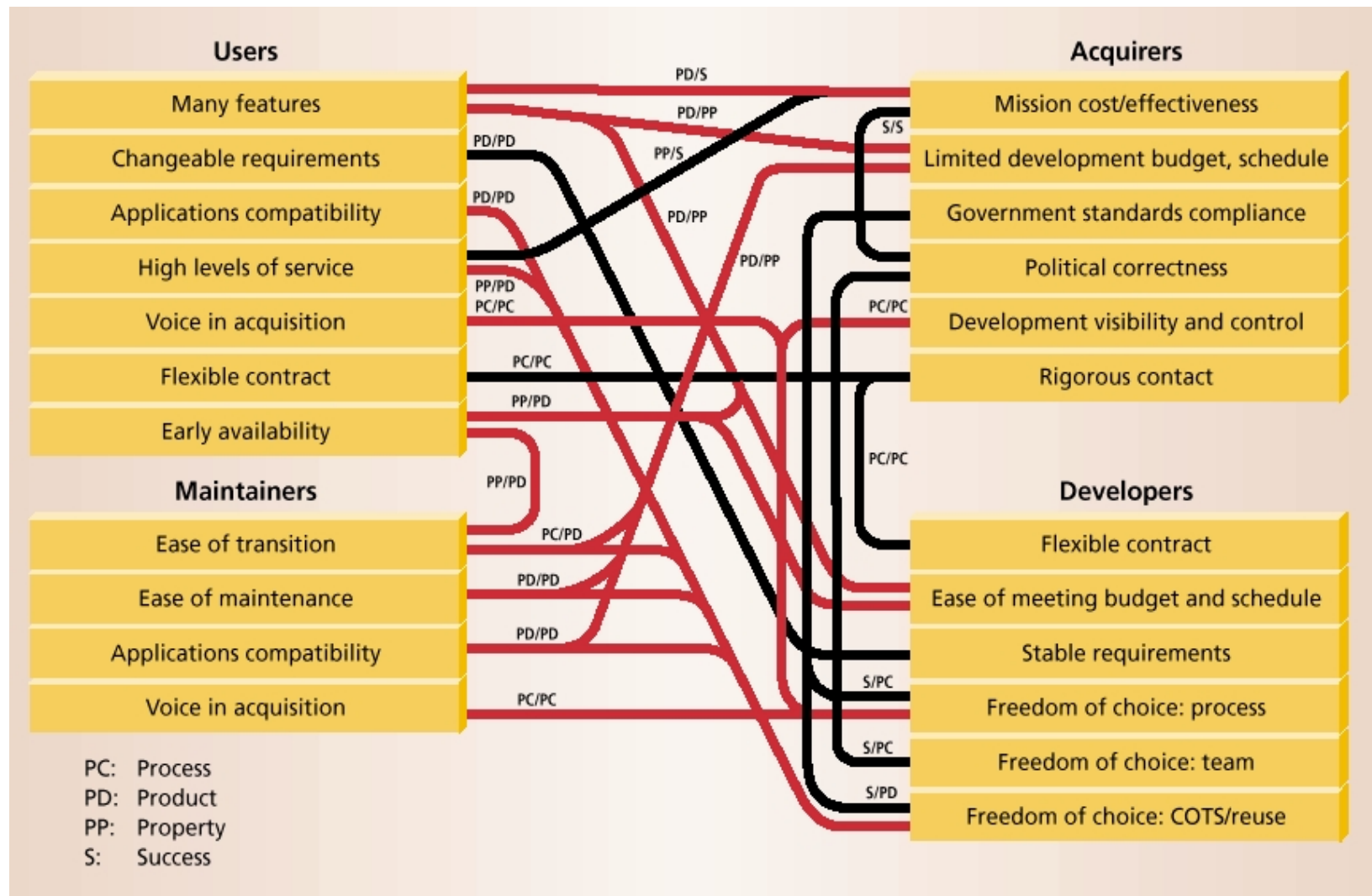


Effect of Missing Operational Stakeholders

- **Interoperators: Problems detected during critical missions**
- **Maintainers: Slow, expensive response to change**
- **Users: Information overload, wrong mission decisions**
- **Administrators: Workflow slowdowns, unused flexibility**

The Model-Clash Spider Web: **Master Net**

- Stakeholder value propositions (win conditions)





Critical Success Factors for Integrated Team Members

- CrossTalk, December 2003

- **Not collaborative: Discord, frustration, loss of morale**
- **Not representative: Delivery of unacceptable systems, late rework**
- **Not authorized: Authorization delays, unsupported systems**
- **Not committed: Missing homework, discontinuities, delays**
- **Not knowledgeable: Unacceptable systems, delays, late rework**



Critical Success Factors: Goal 4

Goal 4. The software product plans and process plans are feasible and compatible. A Feasibility Rationale provides convincing evidence that:

- **4.1 The lifecycle benefits determined by the system and software requirements are worth the lifecycle investments determined by the system and software architecture and choice of components.**
- **4.2 A system built to the system and software architecture will support the operational concept, satisfy the requirements and success-critical stakeholders, and be buildable within the budgets and schedules in the process plans.**
- **4.3 All major software risks have been either resolved or covered by risk management plans.**
- **4.4 Plans for evolutionary/incremental software development and integration are validated for achievability and kept stable during each increment.**
- **4.5 Software technologies, COTS, and NDI components have been validated for maturity and compatibility.**



Life Cycle Anchor Points

- **Common System/Software stakeholder commitment points**
 - Defined in concert with 30 Government, industry organizations
 - Coordinated with Rational's Unified Software Development Process
- **Life Cycle Objectives (LCO; DoD Milestone A)**
 - Stakeholders' commitment to support system architecting
 - Like getting engaged
- **Life Cycle Architecture (LCA; DoD Milestone B)**
 - Stakeholders' commitment to support full life cycle
 - Like getting married
- **Initial Operational Capability (IOC)**
 - Stakeholders' commitment to support operations
 - Like having your first child



LCO (MS A) and LCA (MS B) Pass/Fail Criteria

– Cross Talk, December 2001

A system built to the given architecture will

- **Support the operational concept**
- **Satisfy the requirements**
- **Be faithful to the prototype(s)**
- **Be buildable within the budgets and schedules in the plan**
- **Show a viable business case**
- **Establish key stakeholders' commitment to proceed**

LCO: True for at least one architecture

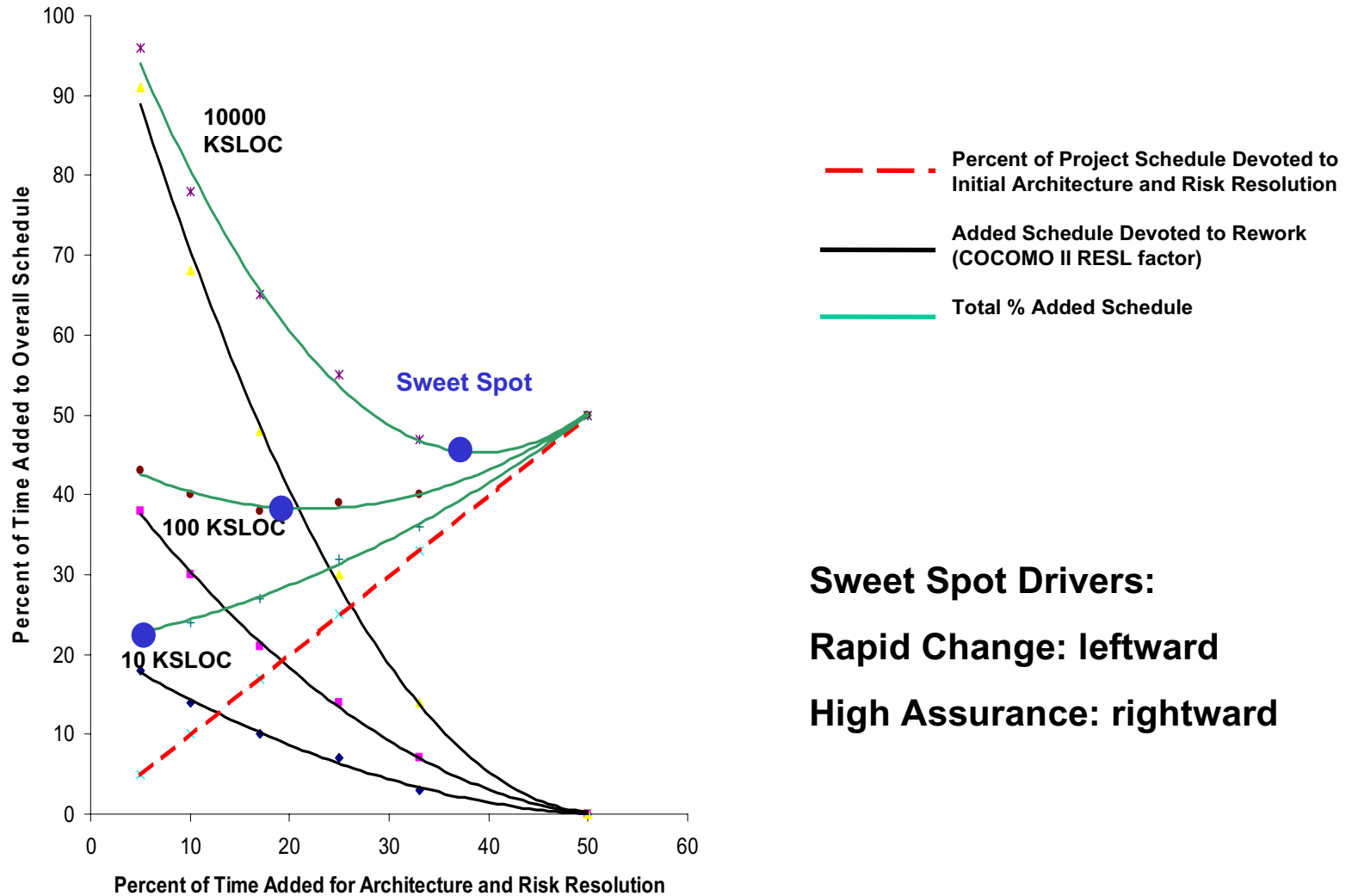
LCA: True for the specific life cycle architecture;

All major risks resolved or covered by a risk management plan



How Much Architecting Is Enough?

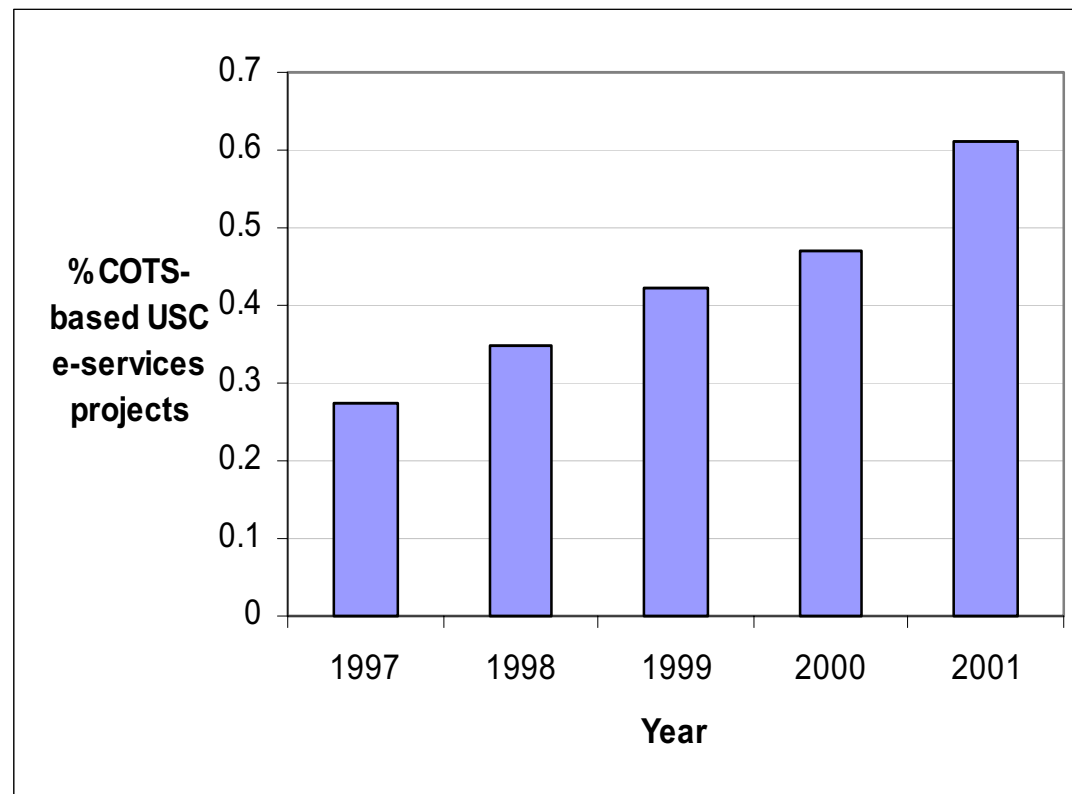
-A COCOMO II Analysis



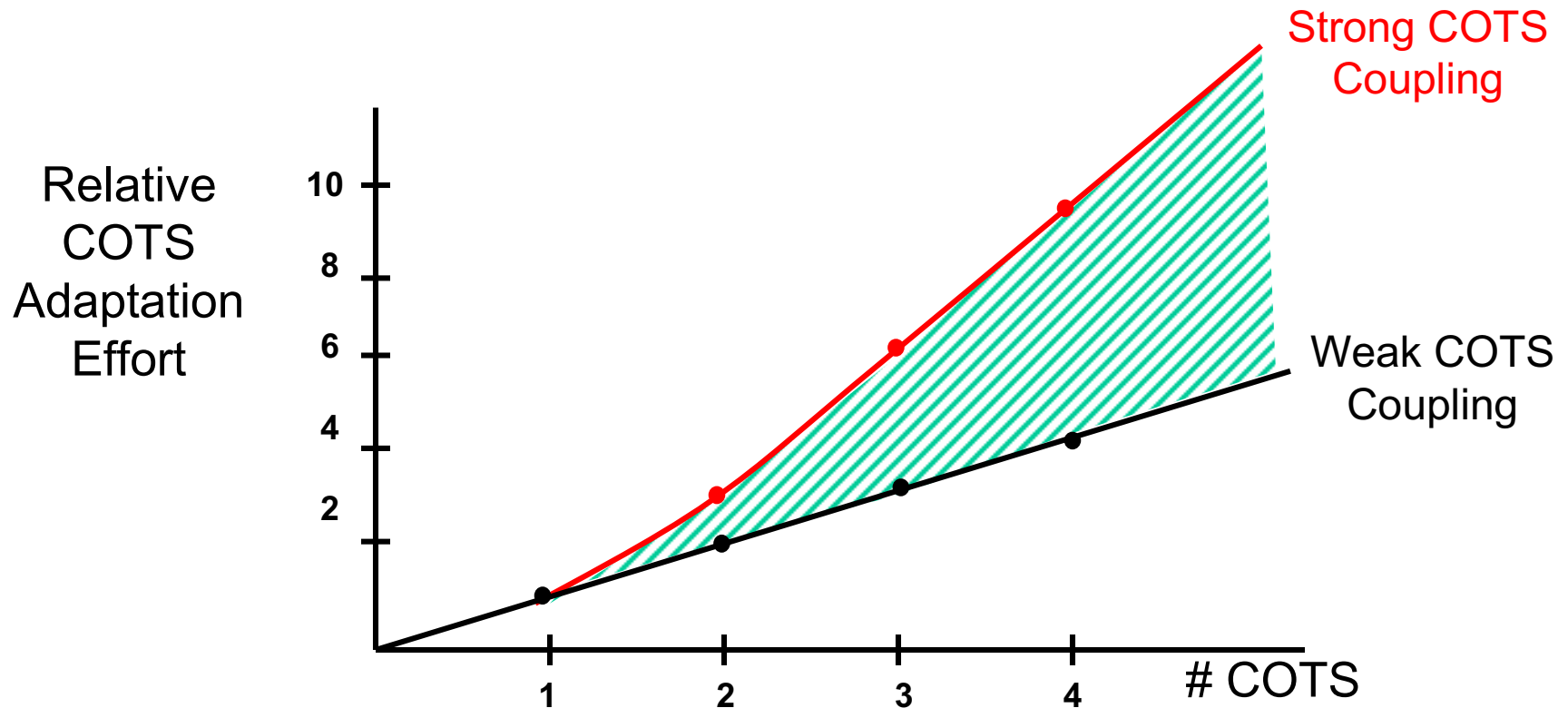


COTS: The Future is Here

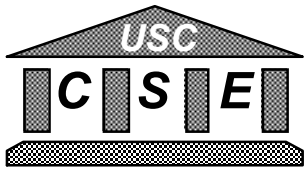
- **Escalate COTS priorities for research, staffing, education**
 - **It's not “all about programming” anymore**



COTS-Based System Effort Scaling

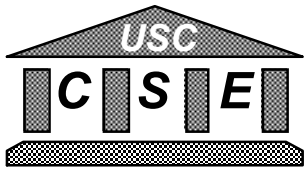


- **Reduce # COTS or Weaken COTS coupling**
 - COTS choices, wrappers, domain architectures, open standards, COTS refresh synchronization



COTS Upgrade Synchronization and Obsolescence

- **Risk #1: Many subcontractors means a proliferation of evolving COTS interfaces**
- **Risk #2: Aggressively-bid subcontracts can lead to delivery of obsolete COTS**
 - **New COTS released every 8-9 months (GSAW)**
 - **COTS unsupported after 3 releases (GSAW)**
 - **An actual delivery: 120 COTS; 46% unsupported**
- **Strategy #1: Emphasize COTS interoperability in source selection process**
- **Strategy #2: Contract provisions ensuring delivery of refreshed COTS products.**



Critical Success Factors: Goal 5

- **Goal 5. Software progress with respect to plans is satisfactory.**
- **5.1 Software cost, schedule, and progress metrics are defined and monitored, and are acceptably consistent with plans.**
- **5.2 Software development, integration, verification and validation, and risk management milestones are being successfully met.**
- **5.3 Progress toward satisfying the software Feasibility Rationale is monitored, and shortfalls are identified as risks to be managed.**
- **5.4 Likely changes in software-related policy, technology, requirements, COTS/NDI components, and interfaces are monitored and analyzed for impact, and are pro-actively being addressed.**
- **5.5 Independent assessments of software aspects are performed periodically.**



Project Top 10 Risk Item List: Satellite Experiment Software

Risk Item	Mo. Ranking			Risk Resolution Progress
	This	Last	#Mo.	
Replacing Sensor-Control Software Developer	1	4	2	Top Replacement Candidate Unavailable
Target Hardware Delivery Delays	2	5	2	Procurement Procedural Delays
Sensor Data Formats Undefined	3	3	3	Action Items to Software, Sensor Teams; Due Next Month
Staffing of Design V&V Team	4	2	3	Key Reviewers Committed; Need Fault-Tolerance Reviewer
Software Fault-Tolerance May Compromise Performance	5	1	3	Fault Tolerance Prototype Successful
Accommodate Changes in Data Bus Design	6	-	1	Meeting Scheduled With Data Bus Designers
Testbed Interface Definitions	7	8	3	Some Delays in Action Items; Review Meeting Scheduled
User Interface Uncertainties	8	6	3	User Interface Prototype Successful
TBDs In Experiment Operational Concept	-	7	3	TBDs Resolved
Uncertainties In Reusable Monitoring Software	-	9	3	Required Design Changes Small, Successfully Made



Independent Assessments

- **Help programs and broaden understanding**
 - Fight PM tunnel vision
 - Identify issues across programs
 - Capture lessons learned
- **Example: Tri-Service Assessment Initiative**
 - Over 50 assessments
 - Direct impact on programs (e.g. FCS, Comanche)
 - Systemic findings from macro-analysis
 - Common causes of problems
 - Chains of events that may signify risks
 - Unintended consequences from policies
 - A major input to this Goal/CSF framework



Example Assessment Summary

	Previous actual	Current projected	Current actual	Next projected	Comments
1. System and software objectives and constraints	Y	G	Y	G	New interoperability O&Cs
1.1 Functionality and performance objectives	G	G	G	G	
1.2 System boundary, environment, interfaces	Y	G	Y	G	Interoperability more complex
1.3 Flexibility and evolvability objectives	Y	G	G	G	
1.4 Environment, resource, policy constraints	G	G	Y	G	Interoperability more complex
1.5 Objectives validated with respect to constraints	Y	G	Y	G	New objectives to validate



Challenge to Community

- **Goal/CSF Framework improvement suggestions**
- **Techniques for avoiding problem situations**
- **Experimental application and feedback**