Educating Students in Value-Based Design and Development

Barry Boehm, USC

CSEET 2006 Keynote Address
April 19, 2006

Outline

- Value-based software engineering (VBSE) motivation and definitions
- Initial VBSE theory (with Apurva Jain)
  - Software process implications
  - Application to case study
- Incorporating VBSE into SE courses
  - SE management and economics
  - SE team project course
- Conclusions and references

Software Testing Business Case

- Vendor proposition
  - Our test data generator will cut your test costs in half
  - We'll provide it to you for 30% of your test costs
  - After you run all your tests for 50% of your original cost, you are 20% ahead
- Any concerns with vendor proposition?

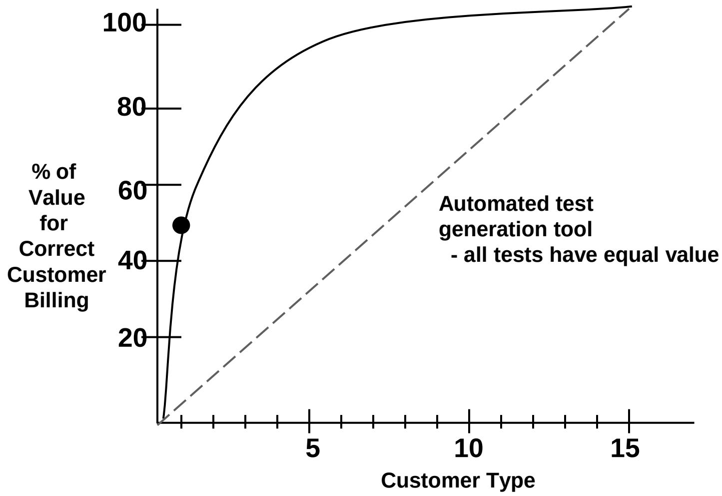
University of Southern California
Center for Software Engineering

Software Testing Business Case

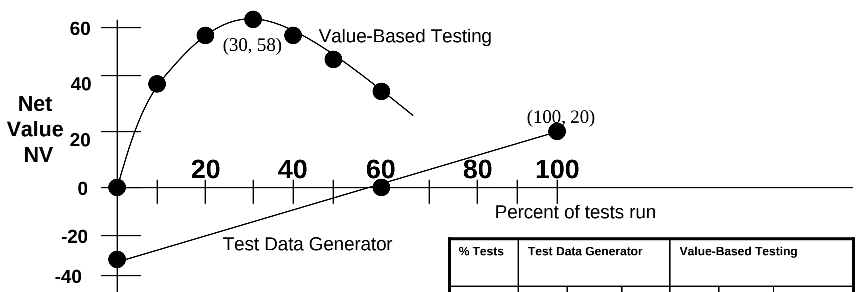
- Vendor proposition
  - Our test data generator will cut your test costs in half
  - We'll provide it to you for 30% of your test costs
  - After you run all your tests for 50% of your original cost, you are 20% ahead
- Any concerns with vendor proposition?
  - Test data generator is value-neutral*
  - Every test case, defect is equally important
  - Usually, 20% of test cases cover 80% of business case

* As are most current software engineering techniques

20% of Features Provide 80% of Value: Focus Testing on These (Bullock, 2000)



Value-Based Testing Provides More Net Value



% Tests	Test Data Generator			Value-Based Testing		
	Cost	Value	NV	Cost	Value	NV
0	30	0	-30	0	0	0
10	35	10	-25	10	50	40
20	40	20	-20	20	75	55
30	45	30	-15	30	88	58
40	50	40	-10	40	94	54
….	….	….	….	….	….	….
100	80	100	+20	100	100	0

Motivation for Value-Based SE

- Current SE methods are basically value-neutral
  - Every requirement, use case, object, test case, and defect is equally important
  - Object oriented development is a logic exercise
  - "Earned Value" Systems don't track business value
  - Separation of concerns: SE's job is to turn requirements into verified code
  - Ethical concerns separated from daily practices
- Value – neutral SE methods are increasingly risky
  - Software decisions increasingly drive system value
  - Corporate adaptability to change achieved via software decisions
  - System value-domain problems are the chief sources of software project failures

The "Separation of Concerns" Legacy

- "The notion of 'user' cannot be precisely defined, and therefore has no place in CS or SE."

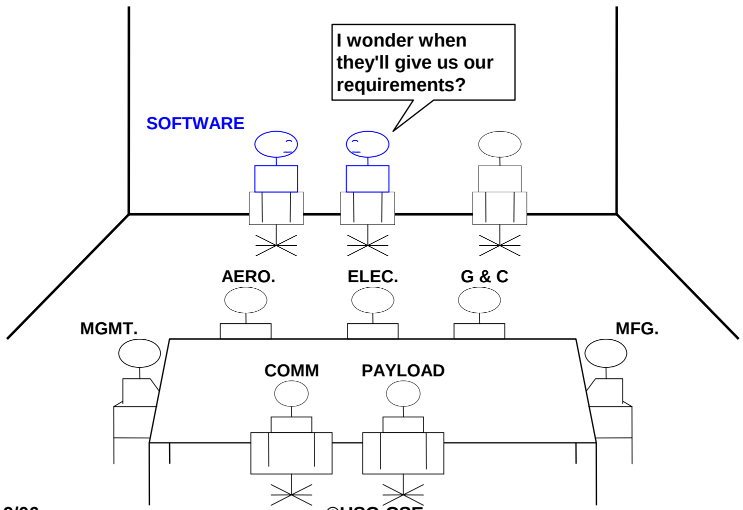
  - Edsger Dijkstra, ICSE 4, 1979

- "Analysis and allocation of the system requirements is not the responsibility of the SE group but is a prerequisite for their work"

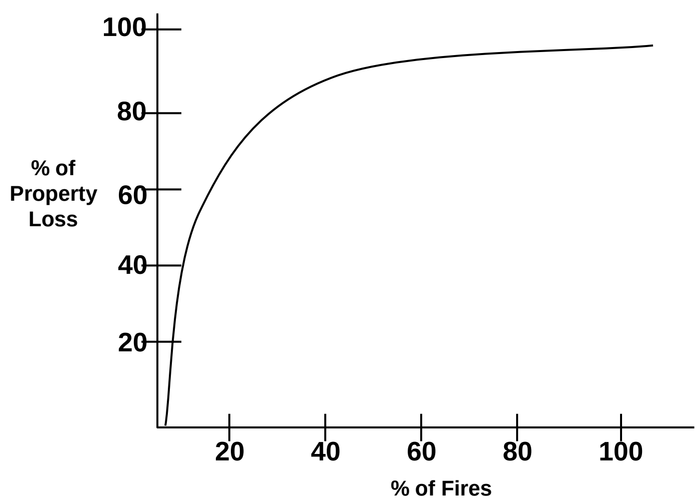
  - Mark Paulk at al., SEI Software CMM* v.1.1, 1993

*Capability Maturity Model

Resulting Project Social Structure



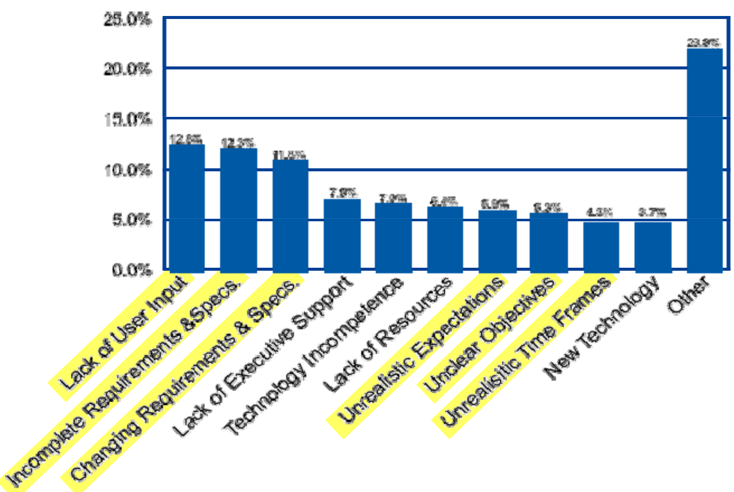
20% of Fires Cause 80% of Property Loss: Focus Fire Dispatching on These?



Penumbra Negotiation Example: Fire Dispatching System

- **Dispatch to minimize value of property loss**
  - **Neglect safety, least-advantaged property owners**
- **English-only dispatcher service**
  - **Neglect least-advantaged immigrants**
- **Minimal recordkeeping**
  - **Reduced accountability**
- **Tight budget; design for nominal case**
  - **Neglect reliability, safety, crisis performance**

Why Software Projects Fail



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

Outline

- Value-based software engineering (VBSE) motivation and definitions
- Initial VBSE theory (with Apurva Jain)
  - Software process implications
  - Application to case study
- Incorporating VBSE into SE courses
  - SE management and economics
  - SE team project course
- Conclusions and references

Initial VBSE Theory: 4+1

- with Apurva Jain

- Engine: Theory W (stakeholder win-win): What values are important?
  - Enterprise Success Theorem
  - Theory of Justice
  - Win-Win Equilibrium and Negotiation
- Four Supporting Theories
  - Utility Theory: How important are the values?
    - Multi-attribute utility; Maslow need hierarchy
  - Decision Theory: How do values determine decisions?
    - Investment theory; game theory; statistical decision theory
  - Dependency Theory: How do dependencies affect value realization?
    - Results chains; value chains; cost/schedule/performance tradeoffs
  - Control Theory: How to monitor and control value realization
    - Feedback control; adaptive control; spiral risk control

Theory W: Enterprise Success Theorem – And informal proof

Theorem: Your enterprise will succeed if and only if it makes winners of your success-critical stakeholders

- Proof of "if":
  Everyone that counts is a winner.
  Nobody significant is left to complain.
- Proof of "only if":
  Nobody wants to lose.
  Prospective losers will refuse to participate, or will counterattack.
  The usual result is lose-lose.

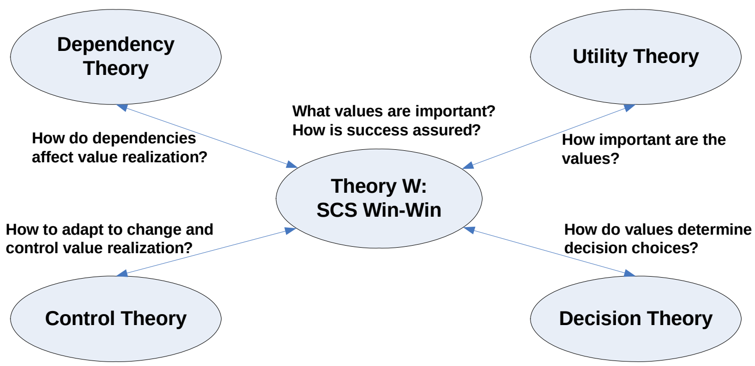
Theory W: WinWin Achievement Theorem

Making winners of your success-critical stakeholders requires:

i. Identifying all of the success-critical stakeholders (SCSs).
ii. Understanding how the SCSs want to win.
iii. Having the SCSs negotiate a win-win set of product and process plans.
iv. Controlling progress toward SCS win-win realization, including adaptation to change.

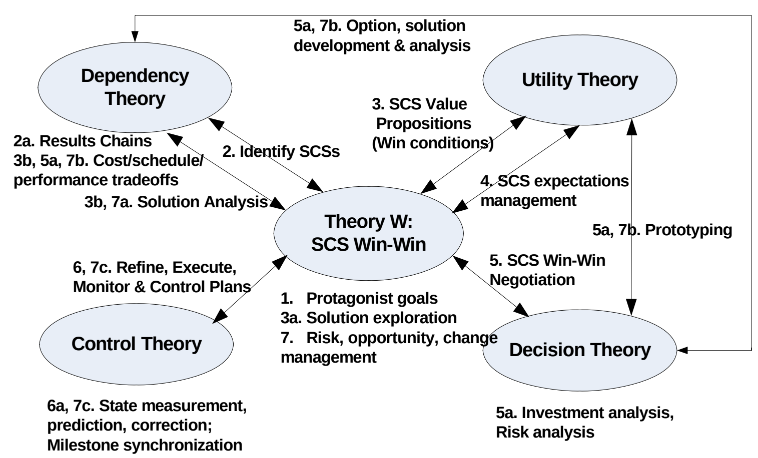
University of Southern California
Center for Software Engineering

VBSE Theory 4+1 Structure



University of Southern California
Center for Software Engineering

Initial VBSE Theory: 4+1 Process
– With a great deal of concurrency and backtracking



SCS: Success-Critical Stakeholder

University of Southern California
Center for Software Engineering

Example Project: Sierra Mountainbikes

- Based on what would have worked on a similar project
- Quality leader in specialty area
- Competitively priced
- Major problems with order processing
  - Delivery delays and mistakes
  - Poor synchronization of order entry, confirmation, fulfillment
  - Disorganized responses to problem situations
  - Excess costs; low distributor satisfaction

University of Southern California
Center for Software Engineering

Order Processing Project Goals

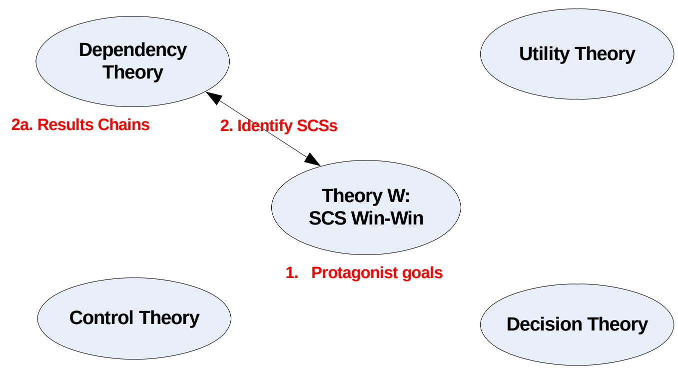
Goals: Improve profits, market share, customer satisfaction via improved order processing

Questions: Current state? Root causes of problems? Keys to improvement?

Metrics: Balanced Scorecard of benefits realized, proxies

- Customer satisfaction ratings; key elements (ITV: in-transit visibility)
- Overhead cost reduction
- Actual vs. expected benefit and cost flows, ROI

Initial VBSE Theory: 4+1 Process, Steps 1 and 2 – With a great deal of concurrency and backtracking



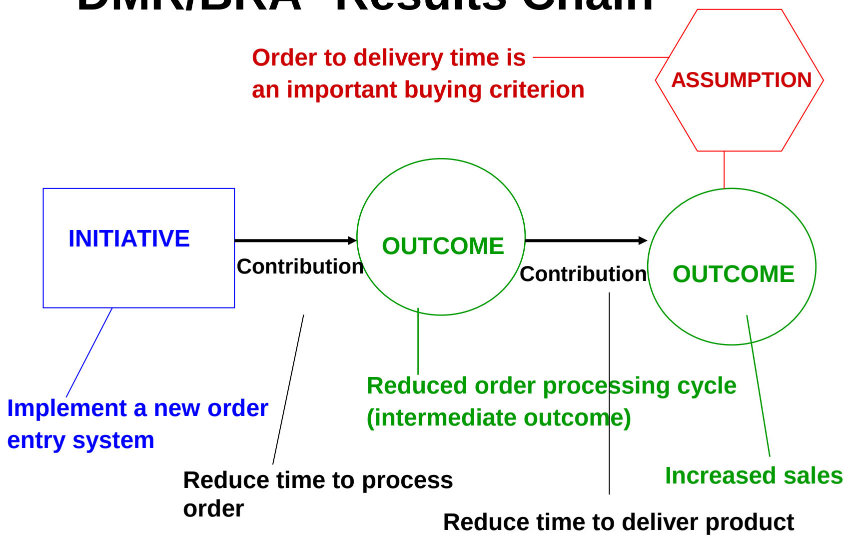
SCS: Success-Critical Stakeholder

Frequent Protagonist Classes

Protagonist Class	*Goals*	*Authority*	*Ideas*	*Resources*
Leader with Goals, Baseline Agenda	X	X	X	X
Leader with Goals, Open Agenda	X	X		X
Entrepreneur with Goals, Baseline Agenda	X		X	X
Entrepreneur with Goals, Open Agenda	X			X
Inventor with Goals, Ideas	X		X	
Consortium with Shared Goals	X	(X)		(X)

- Sierra Moutainbikes: Susan Swanson, new CEO
  - Bicycle champion, MBA, 15 years' experience
  - Leads with goals, open agenda

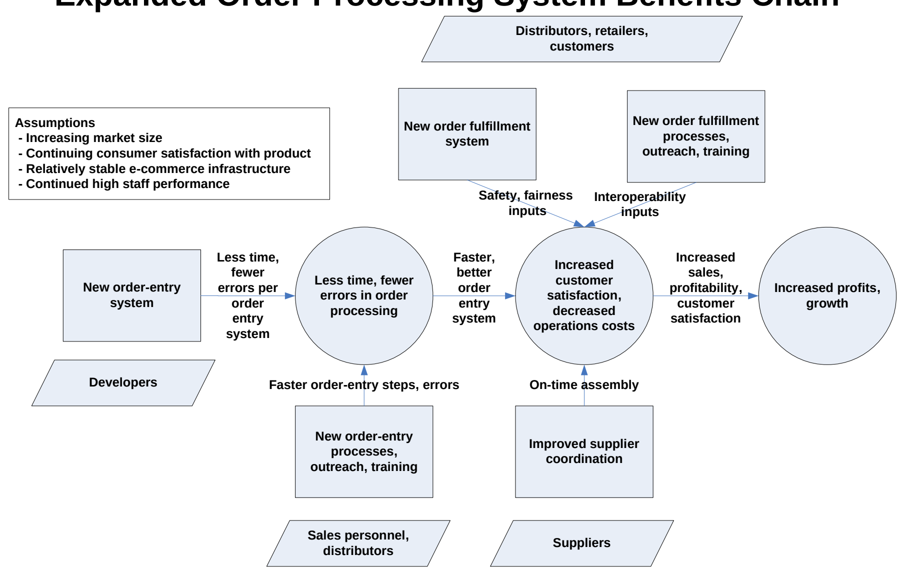
DMR/BRA* Results Chain



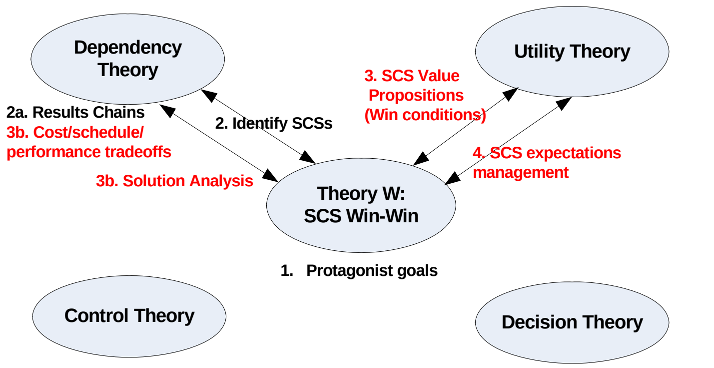
*DMR Consulting Group's Benefits Realization Approach

University of Southern California
Center for Software Engineering

Expanded Order Processing System Benefits Chain



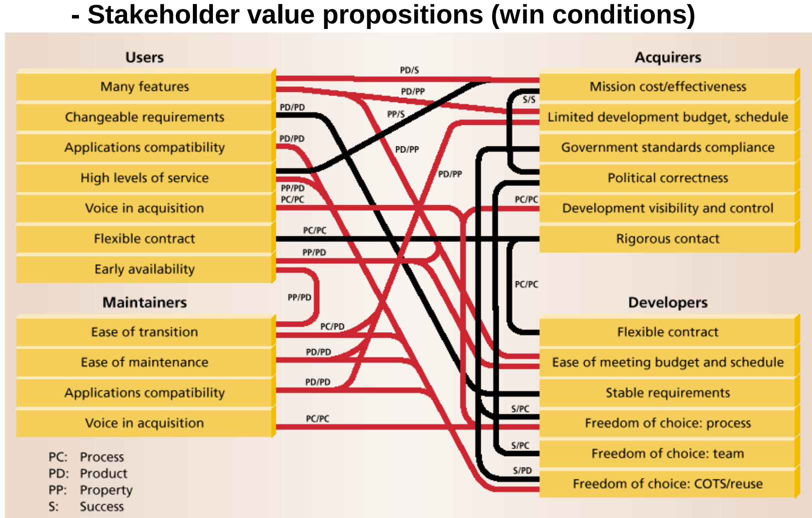
Initial VBSE Theory: 4+1 Process, Steps 3 and 4
– With a great deal of concurrency and backtracking



SCS: Success-Critical Stakeholder

The Model-Clash Spider Web: Master Net

- Stakeholder value propositions (win conditions)



EasyWinWin OnLine Negotiation Steps

Review and Expand Negotiation Topics (Group Outliner)
Jointly review and define the scope of the negotiation. Identify the negotiation topics for your EasyWinWin activity.

Brainstorm Stakeholder Interests (Electronic Brainstorming)
Collect ideas about Win Conditions for your EasyWinWin activity

Converge on Win Conditions (Categorizer)
Jointly craft and organize a succinct list of win conditions.

Capture Glossary of Terms (Topic Commenter)
Define important terms of the domain.

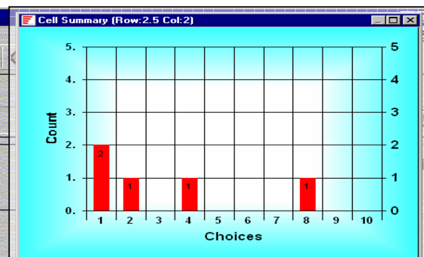
Prioritize Win Conditions (Alternative Analysis)
Determine the business importance and the ease of implementation of all win conditions. Reveal issues and constraints.

WinWin Tree (Group Outliner)
Identify Issues and Options. Negotiate Agreements.

Organize Negotiation Results (Categorizer)
Categorize the results using the negotiation topics.

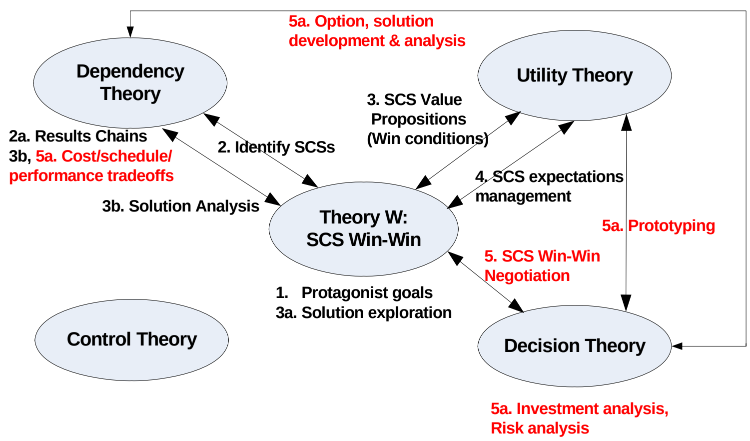
Red cells indicate lack of consensus.

Oral discussion of cell graph reveals unshared information, unnoticed assumptions, hidden issues, constraints, etc.



	Features	Importance	Ease of Implementation	Total	Mean
2.	Application Capabilities				
2.1	W2 Integrate banner ads with email and chat	10.00	6.50	16.50	8.25
2.2	W3 The banner will provide a link to the universit	10.00	10.00	20.00	10.00
2.3	W4 Interface for advertisers to select their sched	8.67	3.00	11.67	5.83
2.4	W5 Default banner of bookstore if no other events	8.00	10.00	18.00	9.00
2.5	W6 The site management must have a website which	10.00	10.00	20.00	10.00
2.6	W7 Different kinds of advertising, including sales	10.00	10.00	20.00	10.00
2.7	W8 Flexible text on banners	10.00	5.00	15.00	7.50
2.8	W9 Display address of the bookstore, a map of it a	4.00	7.50	11.50	5.75
2.9	W10 Ads must be hyperlinked so that users can clic	7.33	6.00	13.33	6.67
2.10	W11 Link to bookstore site (incl book's prices)	9.33	10.00	19.33	9.67
2.11	W12 Web statistics tracking to determine number of	8.00	4.00	12.00	6.00
2.12	W13 Input of banner contents to admin via email	5.50	10.00	15.50	7.75

Initial VBSE Theory: 4+1 Process, Step 5 – With a great deal of concurrency and backtracking



SCS: Success-Critical Stakeholder

Project Strategy and Partnerships

- Partner with eServices, Inc. for order processing and fulfillment system
  - Profit sharing using jointly-developed business case
- Partner with key distributors to provide user feedback
  - Evaluate prototypes, beta-test early versions, provide satisfaction ratings
- Incremental development using MBASE/RUP anchor points
  - Life Cycle Objectives; Architecture (LCO; LCA)
  - Core Capability Drivethrough (CCD)
  - Initial; Full Operational Capability (IOC; FOC)
- Architect for later supply chain extensions

Business Case Analysis

- Estimate costs and schedules
  - COCOMO II and/or alternative for software
  - PRICE H or alternative for hardware
  - COSYSMO for systems engineering
- Estimate financial benefits
  - Increased profits
  - Reduced operating costs
- Compute Return on Investment
  - ROI = (Benefits – Costs) / Costs
  - Normalized to present value
- Identify quantitative metrics for other goals
  - Customer satisfaction ratings
    - Ease of use; In-transit visibility; overall
  - Late delivery percentage

Order Processing System Schedules and Budgets

Milestone	Due Date	Budget ($K)	Cumulative Budget ($K)
Inception Readiness	1/1/2004	0	0
Life Cycle Objectives	1/31/2004	120	120
Life Cycle Architecture	3/31/2004	280	400
Core Capability Drivethrough	7/31/2004	650	1050
Initial Oper. Capability: SW	9/30/2004	350	1400
Initial Oper. Capability: HW	9/30/2004	2100	3500
Developed IOC	12/31/2004	500	4000
Responsive IOC	3/31/2005	500	4500
Full Oper. Cap'y CCD	7/31/2005	700	5200
FOC Beta	9/30/2005	400	5600
FOC Deployed	12/31/2005	400	6000
Annual Oper. & Maintenance		3800	
Annual O&M; Old System		7600	

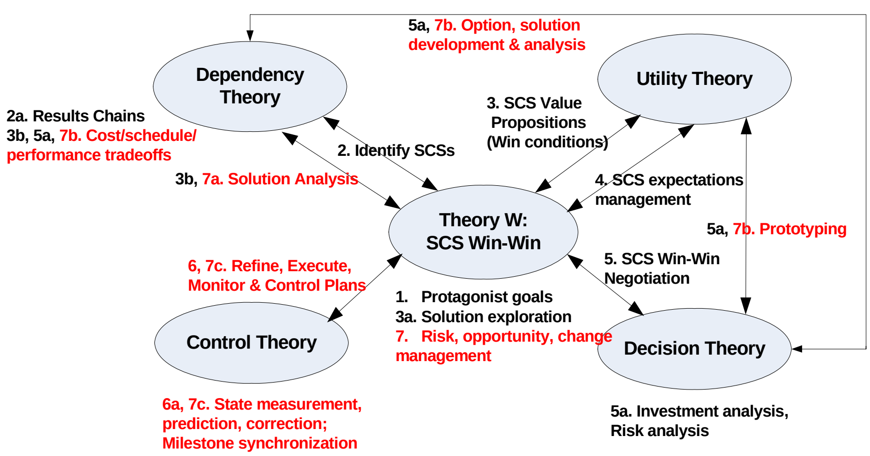
University of Southern California
Center for Software Engineering

Order Processing System: Expected Benefits and Business Case

		Current System			New System											
					Financial								Customers			
Date	Market Size ($M)	Market Share %	Sales	Profits	Market Share %	Sales	Profits	Cost Savings	Change in Profits	Cum. Change in Profits	Cum. Cost	ROI	Late Delivery %	Customer Satisfaction (0-5)	In-Transit Visibility (0-5)	Ease of Use (0-5)
12/31/03	360	20	72	7	20	72	7	0	0	0	0	**0**	12.4	**1.7**	1.0	1.8
12/31/04	400	20	80	8	20	80	8	0	0	0	4	**-1**	11.4	**3.0**	2.5	3.0
12/31/05	440	20	88	9	22	97	10	2.2	3.2	3.2	6	**-.47**	7.0	**4.0**	3.5	4.0
12/31/06	480	20	96	10	25	120	13	3.2	6.2	9.4	6.5	**.45**	4.0	**4.3**	4.0	4.3
12/31/07	520	20	104	11	28	146	16	4.0	9.0	18.4	7	**1.63**	3.0	**4.5**	4.3	4.5
12/31/08	560	20	112	12	30	168	19	4.4	11.4	29.8	7.5	**2.97**	2.5	**4.6**	4.6	4.6

University of Southern California
Center for Software Engineering

Initial VBSE Theory: 4+1 Process, Steps 6 and 7 – With a great deal of concurrency and backtracking



SCS: Success-Critical Stakeholder

Value-Based Expected/Actual Outcome Tracking Capability

Milestone	Schedule	Cost ($K)	Op'l Cost Savings	Market Share %	Annual Sales ($M)	Annual Profits ($M)	Cum. Profits	ROI	Late Delivery %	Customer Satisfaction	ITV	Ease of Use	Risks/Opportunities
Life Cycle Architecture	3/31/04	400		20	72	7.0			12.4	1.7	1.0	1.8	Increased COTS ITV risk, fallback identified.
	3/31/04	427		20	72	7.0			12.4	1.7	1.0	1.8	
Core Capability Demo (CCD)	7/31/04	1050											Using COTS ITV fallback; new HW competitor; renegotiating HW.
	7/20/04	1096								2.4*	1.0*	2.7*	
Software Initial Op'l Capability (IOC)	9/30/04	1400											
	9/30/04	1532								2.7*	1.4*	2.8*	
Hardware IOC	9/30/04	3500											$200K savings from renegotiated HW.
	10/11/04	3432											
Deployed IOC	12/31/04	4000		20	80	8.0	0.0	-1.0	11.4	3.0	2.5	3.0	New COTS ITV source identified, being prototyped.
	12/20/04	4041		22	88	8.6	0.6	-.85	10.8	2.8	1.6	3.2	
Responsive IOC	3/31/05	4500	300						9.0	3.5	3.0	3.5	
	3/30/05	4604	324						7.4	3.3	1.6	3.8	
Full Op'l Capability CCD	7/31/05	5200	1000							3.5*	2.5*	3.8*	New COTS ITV source initially integrated.
	7/28/05	5328	946										
Full Op'l Capability Beta	9/30/05	5600	1700							3.8*	3.1*	4.1*	
	9/30/05	5689	1851										
Full Op'l Capability Deployed Release 2.1	12/31/05	6000	2200	22	106	12.2	3.2	-.47	7.0	4.0	3.5	4.0	
	12/20/05	5977	2483	24	115	13.5	5.1	-.15	4.8	4.1	3.3	4.2	
	6/30/06	6250											

Outline

- Value-based software engineering (VBSE) motivation and definitions
- Initial VBSE theory (with Apurva Jain)
  - Software process implications
  - Application to case study
- Incorporating VBSE into SE courses
  - SE management and economics (CS 510)
  - SE team project course (CS 577)
- Conclusions and references

Comparison of CS 510 and CS 577a

CS 510

- VBSE Theory, Practice
- COCOMO II Extensions
- Microeconomics
  - Decision Theory
- Agile and Rapid Development
- People Management
- 2 Midterms, Final

CS 510 and CS 577a (overlap)

- VBSE Framework
- MBASE
- WinWin Spiral
  - Risk Management
- Planning & Control
  - COCOMO II
- Business Case Analysis

CS 577a

- S/W - System Architecting
- Operational Concept & Rqts. Definition
  - WinWin System
  - Prototyping
- OO Analysis & Design
  - Rational Rose
- Team Project (DEN: IV&V)

CS 577 Learning Objectives

" Software Engineering:" The disciplines which distinguish the coding of a computer program from the development of a software product.

Stages / Issues	Requirements, Architecture	Design, Code	**Test, Implement, Maintain**
Computer Science		CS Focus	
User Applications			
Economics			
People			

- **Prepare you for software leadership careers through the 2040's**
  - **Agility , discipline, COTS/OSS, scalable spirals, service-based systems**
- **Integrate all theses considerations**
  - **Via value-based, model – driven software engineering (VBSE, MBASE) project experience**

e-Services Projects Overview

- Clients identify prospective projects
  - Operational capabilities or feasibility explorations
  - Fall: 12 weeks to prototype, analyze, design, plan, validate
  - Spring: 12 weeks to develop, test, transition
  - MS-level, 5-6 person, CS 577 project course
- Clients, CSE, ISD negotiate workable projects
  - Useful results within time constraints
  - Operationally supportable as appropriate
- Clients work with teams to define, steer, evaluate projects
  - Exercise prototypes, negotiate requirements, review progress
  - Mutual learning most critical success factor

Stakeholder Win-Win Approach

Stakeholders	Win Conditions
•Students, Employers	•Full range of SW Engr. skills •Real-client project experience •Non-outsourceable skills •Advanced SW tech. experience
•Project clients	•Useful applications •Advanced SW tech. understanding •Moderate time requirements
•Faculty, Profession	•Educate future SW Engr. leaders •Better SW Engr. technology •Applied on real-client projects

Software Engineering Project Course (CS 577)

- Fall: Develop Life Cycle Architecture Packages
  - Ops. Concept, Requirements, Prototype, Architecture, Plan
  - Feasibility Rationale, including business case
  - Results chain linking project results to desired outcomes
  - 20 projects; 120 students; about 20 clients
- Spring: Develop Initial Operational Capability
  - 6-10 projects; 30-50 students; 6-10 clients
  - Software, personnel, and facilities preparation
  - 2-week transition period
  - then the student teams disappear
- Tools and techniques: EasyWinWin; Results Chain Rational Rose, Clear Case; USC COCOMO II; MS Project; USC MBASE method
  - Reworked annually based on student & client feedback

Win Win Spiral Anchor Points

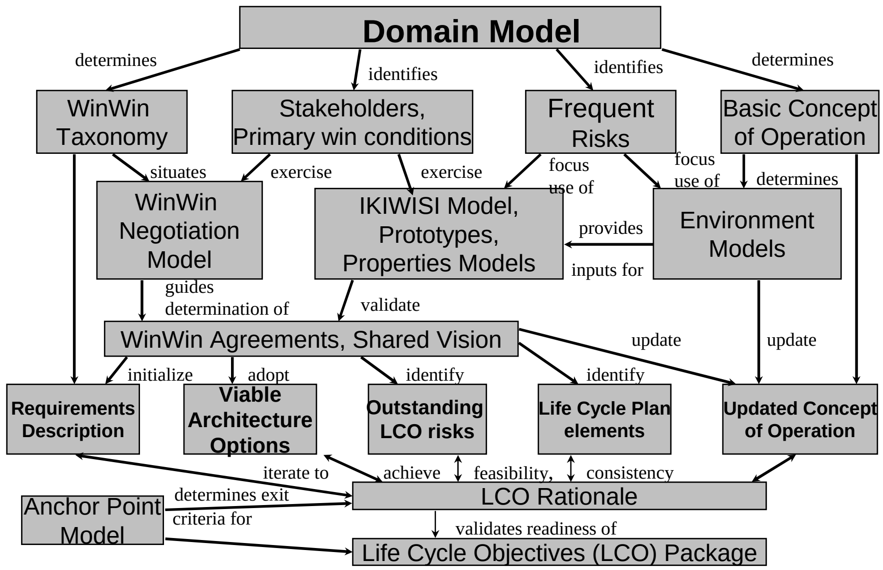
(Risk-driven level of detail for each element)

Milestone Element	Life Cycle Objectives (LCO)	Life Cycle Architecture (LCA)
Definition of Operational Concept	• Top-level system objectives and scope - System boundary - Environment parameters and assumptions - Evolution parameters • Operational concept - Operations and maintenance scenarios and parameters - Organizational life-cycle responsibilities (stakeholders)	• Elaboration of system objectives and scope of increment • Elaboration of operational concept by increment
System Prototype(s)	• Exercise key usage scenarios • Resolve critical risks	• Exercise range of usage scenarios • Resolve major outstanding risks
Definition of System Requirements	• Top-level functions, interfaces, quality attribute levels, including: - Growth vectors and priorities - Prototypes • Stakeholders' concurrence on essentials	• Elaboration of functions, interfaces, quality attributes, and prototypes by increment - Identification of TBD's((to-be-determined items) • Stakeholders' concurrence on their priority concerns
Definition of System and Software Architecture	• Top-level definition of at least one feasible architecture - Physical and logical elements and relationships - Choices of COTS and reusable software elements • Identification of infeasible architecture options	• Choice of architecture and elaboration by increment - Physical and logical components, connectors, configurations, constraints - COTS, reuse choices - Domain-architecture and architectural style choices • Architecture evolution parameters
Definition of Life-Cycle Plan	• Identification of life-cycle stakeholders - Users, customers, developers, maintainers, interoperators, general public, others • Identification of life-cycle process model - Top-level stages, increments • Top-level WWWWWHH* by stage	• Elaboration of WWWWWHH* for Initial Operational Capability (IOC) - Partial elaboration, identification of key TBD's for later increments
Feasibility Rationale	• Assurance of consistency among elements above - via analysis, measurement, prototyping, simulation, etc. - Business case analysis for requirements, feasible architectures	• Assurance of consistency among elements above • All major risks resolved or covered by risk management plan

*WWWWWHH: Why, What, When, Who, Where, How, How Much

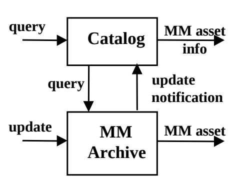
University of Southern California
Center for Software Engineering

MBASE Model Integration: LCO Stage



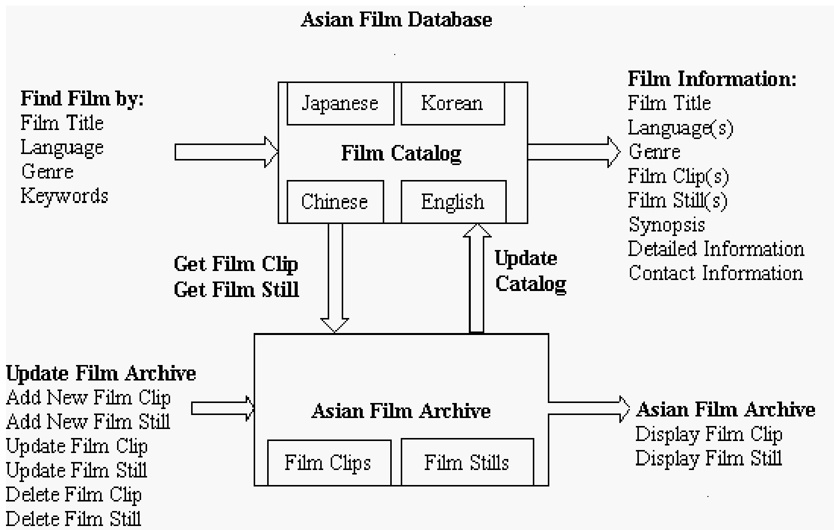
University of Southern California
Center for Software Engineering

S&C Subdomain (General)

Type of Application	Simple Block Diagram	Examples (project nos.)	Deveoper Simplifiers	Developer Complicators
Multimedia Archive		1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 20, 31, 32, 35, 36, 37, 39	• Use standard query languages • Use standard or COTS search engine • Uniform media formats	• Natural language processing • Automated cataloging or indexing • Digitizing large archives • Digitizing complex or fragile artifacts • Rapid access to large Archives • Access to heterogeneous media collections • Automated annotation/description/ or meanings to digital assets • Integration of legacy systems

University of Southern California
Center for Software Engineering

S&C Subdomain (Specialized to Project)



University of Southern California
Center for Software Engineering

S&C Developer-Side Simplifiers

Simplifiers	Risks and Trade-offs
Generic Uniform Media Formats **Specific** All video clips are stored using an open file format for video/audio (e.g., MPEG). All film stills are stored using an open image file format (e.g., JPEG). The inverse complicator is to store film clips using streaming video technologies	This means that we may have to convert existing digital assets or digitize the original media, which may be costly. A unique file format limits the user base to those who have viewers for that particular file format The chosen file format may not be the most efficient for the various types of media (in terms of compression rates, quality, etc...)
Generic Use Standard Query Languages **Specific** Organize catalog and archive relationally so that queries will be limited to standard search formats,: match exactly by value on any of the fields with or without using boolean combinations (AND, OR, NOT, etc...), or using pattern matching (SQL *LIKE* keyword)	May not be as effective for "discovering" assets in the archive: users must know what they're looking for, in order to search for it
Generic Use Standard COTS **Specific** Use a standard Relational Database Management System (RDBMS) that supports storing multi-media assets	A Relational Database Management System may not be most suited for archival of multi-media assets. A Relational Database Management System may have a high initial cost, high implementation, and high administration cost (requires specialized knowledge skills)

Team Structure

- Six-person teams
  - Each artifact should have a lead producer and a co-producer
- Project Manager generally the lead for Feasibility Rationale
  1. Ensures consistency among the team members' artifacts (and documents this in the Rationale).
  2. Leads the team's development of plans for achieving the project results, and ensures that project performance tracks the plans.

  Teams formed by Wednesday, Sept. 7
  - Web questionnaires should help in team formation
- Start forming teams now!
  - What are your skills? What roles would you prefer?
  - What skills does your team need? Who does them?
  - What projects does your team prefer?

Major Class Project Milestones

September 7	--	All teams formed
September 16	--	Initial Shared Vision, Scenarios
September 26	--	Easy WinWin Results, Prototypes
October 10	--	LCO Drafts on Web Site
October 17- 21	--	LCO Architecture Reviews
October 24	--	LCO Package Due
November 21	--	LCA Drafts on Web Site
Nov.28 – Dec.2	--	LCA Architecture Reviews
December 5	--	LCA Package Due
December 7	--	Individual Critiques Due

Cognitive Demands Analysis

Project Tasks	Risk Management Skills - Skill-building activities
• Select projects; form teams	• Project risk identification • Staffing risk assessment and resolution - Readings, lectures, homework, case study, guidelines
• Plan early phases	• Schedule/budget risk assessment, planning • Risk–driven processes (spiral, MBASE) - Readings, lectures, homework, guidelines, planning and estimating tools
• Achieve stakeholders' shared vision	• Simplifier/complicator analysis • Prototyping as buying information to reduce risk - Readings, lectures, homework, prototype, WinWin tool
• Formulate, validate concept of operation	• Risk-driven level of detail - Readings, lecture, guidelines, project
• Manage to plans	• Risk monitoring and control - Readings, lecture, guidelines, project
• Develop, validate LCO* package	• Risk assessment and prioritization - Readings, lecture, guidelines, project
• LCO Architecture Review	• Risk-driven review process • Review of top-N project risks -Readings, lecture, case studies, review

From Data Mining the Library Catalogue's LCA

ROI Analysis Example (Part I)

Inception and Elaboration Time Invested (CS577a)	
Meetings with Full Team & Individual Members (10% time for 12 weeks)	48 Hours
Email time (1.5% time for 12 weeks)	7 Hours
Architecture Review Board(s)	6 Hours
Total (Inception and Elaboration Time)	61 Hours
Construction and Transition Time Invested (CS577b)	
Meetings with Full Team & Individual Members (7% time for 12 weeks)	34 Hours
Email time (1% time for 12 weeks)	5 Hours
Architecture Review Board(s)	6 Hours
Transition Setup (rough estimate)	10 Hours
Total (Construction/Transition Time)	54 Hours
Semester Maintenance	
Maintenance Time (disk cleanup @ 2.5% time for 16 week semester)	16 Hours
Work w/maintenance team personnel on updates (1/5 Inception/Elaboration time)	12 Hours
Total (Semester Maintenance Time)	28 Hours

ROI Example (Part II)

Using the previous numbers as the Investment Costs, and calculating hours saved for one person as the time it takes to review an original sized report compared to a SURG filtered report of 1/3 the original Unicorn size (See Section 2.1.5.1), the Return On Investment for this project is shown in the table and chart below:

1/3 Year Semesters	Fall '98	Spr '99	Sum'99	Fall'99	Spr'00	Sum'00
Hours Time Saved Per Month (1 person - Using 1/3 report size reduction)		5	19	19	19	19
Reports per Semester		19	78	78	78	78
Time Saved In Hours		19	78	78	78	78
Cumulative Hours		19	97	175	252	330
Time Invested in Hours	61	54	28	28	28	28
Cumulative Hours	61	116	144	172	200	229
Return On Investment		0.17	0.67	1.01	1.26	1.44

Spring Schedule (2006)

Jan. 18- Feb. 14: Work with teams:
- **Rebaseline prototype, prioritize requirements**
- **Plan for CS 577b specifics, including transition strategy, key risk items**
- **Participate in ARB review**

Feb 15 - Apr 11: Scheduled Weekly Meetings with Teams to:
- **Discuss status and plans**
- **Provide access to key transition people for strategy and readiness discussions**

Mar 8 - 27: Core Capability Drivethroughs

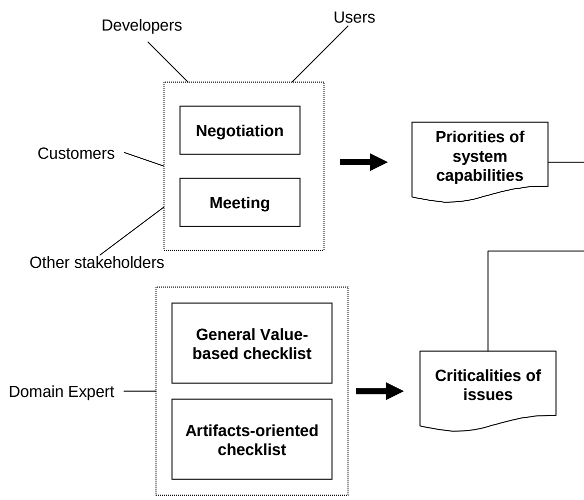
Apr 13 - Apr 14: Project Transition Readiness ARB Reviews

Apr 15: Installation and Transition
- **Install Product**
- **Execute Transition Plan**

May 1 - 2: Release Readiness Review for Initial Operational Capability

May 3: Client Evaluations

Value-Based Review Process (II)



		Priority		
		High	Medium	Low
Criticality	High	1	4	6
	Medium	2	5	optional
	Low	3	optional	optional

Number indicates the usual ordering of review*

* May be more cost-effective to review highly-coupled mixed-priority artifacts.

Value-Based Checklist (I) <General Value-Based Checklist>

	High-Criticality Issues	Medium-Criticality Issues	Low-Criticality Issues
Completeness	•**Critical missing elements: backup/ recovery, external interfaces, success-critical stakeholders; critical exception handling, missing priorities** •**Critical missing processes and tools; planning and preparation for major downstream tasks (development, integration, test, transition)** •**Critical missing project assumptions (client responsiveness, COTS adequacy, needed resources)**	•**Medium-criticality missing elements, processes and tools: maintenance and diagnostic support; user help** •**Medium-criticality exceptions and off-nominal conditions; smaller tasks (review, client demos), missing desired growth capabilities, workload characterization**	•**Easily-deferrable, low-impact missing elements: straightforward error messages, help messages, GUI details doable via GUI builder, project task sequence details**
Consistency/ Feasibility	•**Critical elements in OCD, SSRD, SSAD, LCP not traceable to each other** •**Critical inter-artifact inconsistencies: priorities, assumptions, input/output, preconditions/post-conditions** •**Missing evidence of critical consistency/feasibility assurance in FRD**	•**Medium-criticality shortfalls in traceability, inter-artifact inconsistencies, evidence of consistency/feasibility in FRD**	•**Easily-deferrable, low-impact inconsistencies or inexplicit traceability: GUI details, report details, error messages, help messages, grammatical errors**
Ambiguity	•**Vaguely defined critical dependability capabilities: fault tolerance, graceful degradation, interoperability, safety, security, survivability** •**Critical misleading ambiguities: stakeholder intent, acceptance criteria, critical user decision support, terminology**	•**Vaguely defined medium-criticality capabilities, test criteria** •**Medium-criticality misleading ambiguities**	•**Non-misleading, easily deferrable, low-impact ambiguities: GUI details, report details, error messages, help messages, grammatical errors**
Conformance	•**Lack of conformance with critical operational standards, external interfaces**	•**Lack of conformance with medium-criticality operational standards, external interfaces** •**Misleading lack of conformance with document formatting standards, method and tool conventions**	•**Non-misleading lack of conformance with document formatting standards, method and tool conventions, optional or low-impact operational standards**
Risk	•**Missing FRD evidence of critical capability feasibility: high-priority features, levels of service, budgets and schedules** •**Critical risks in top-10 risk checklist: personnel, budgets and schedules, requirements, COTS, architecture, technology**	•**Missing FRD evidence of mitigation strategies for low-probability high-impact or high-probability, low-impact risks: unlikely disasters, off-line service delays, missing but easily-available information**	•**Missing FRD evidence of mitigation strategies for low-probability, low-impact risks**

Value-Based Reading (VBR) Experiment

— Keun Lee, ISESE 2005

By Number	P-value	% Gr A higher	By Impact	P-value	% Gr A higher
Average of Concerns	0.202	34	Average Impact of Concerns	0.049	65
Average of Problems	0.056	51	Average Impact of Problems	0.012	89
Average of Concerns per hour	0.026	55	Average Cost Effectiveness of Concerns	0.004	105
Average of Problems per hour	0.023	61	Average Cost Effectiveness of Problems	0.007	108

- Group A: 15 IV&V personnel using VBR procedures and checklists
- Group B 13 IV&V personnel using previous value-neutral checklists
  - Significantly higher numbers of trivial typo and grammar faults

Experiment

2005-06: Finished Transition Readiness Reviews

- On-schedule with satisfied customers
  - Physics education support (USC)
  - Data mining PubMed results (USC, UCLA)
  - USC football recruiting database (USC)
  - Web-based XML editing (USC)
  - Intelligent, diff-ing CodeCount (Aerospace, NGC)
  - Code Count product line with XML (Aerospace, NGC)
  - Rule-based editor for science data (JPL)
  - eBay notification system (Klappholz)
  - Template-based code generator (Sophoi)

Conclusions

- Current SE methods are basically value-neutral
- Value-neutral SE methods are increasingly risky
- VBSE agenda making progress, but major challenges remain
  - Evolving VBSE theory
  - Creating VB counterparts for value-neutral SE methods
- VBSE helps student team projects succeed

References - I

C. Baldwin & K. Clark, Design Rules: The Power of Modularity, MIT Press, 1999.

S. Biffl, A. Aurum, B. Boehm, H. Erdogmus, and P. Gruenbacher (eds.), Value-Based Software Engineering, Springer, 2005.

B. Boehm, "Value-Based Software Engineering," ACM Software Engineering Notes, March 2003.

B. Boehm, C. Abts, A.W. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece, Software Cost Estimation with COCOMO II, Prentice Hall, 2000.

B. Boehm and L. Huang, "Value-Based Software Engineering: A Case Study, Computer, March 2003, pp. 33-41.

B. Boehm & K. Sullivan, "Software Economics: A Roadmap," The Future of Software Economics, A. Finkelstein (ed.), ACM Press, 2000.

B. Boehm and R. Turner, Balancing Agility and Discipline: A Guide for the Perplexed, Addison Wesley, 2003 (to appear).

B. Boehm, L. Huang, A. Jain. R. Madachy, " The ROI of Software Dependability: The iDAVE Model", IEEE Software Special Issue on Return on Investment, May/June 2004.

M. Denne and J. Cleland-Huang, Software by Numbers, Prentice Hall, 2004.

References - II

S. Faulk, D. Harmon, and D. Raffo, “Value-Based Software Engineering (VBSE): A Value-Driven Approach to Product-Line Engineering,” Proceedings, First Intl. Conf. On SW Product Line Engineering, August 2000.

R. Kaplan & D. Norton, The Balanced Scorecard: Translating Strategy into Action, Harvard Business School Press, 1996.

D. Reifer, Making the Software Business Case, Addison Wesley, 2002.

K. Sullivan, Y. Cai, B. Hallen, and W. Griswold, “The Structure and Value of Modularity in Software Design,” Proceedings, ESEC/FSE, 2001, ACM Press, pp. 99-108.

J. Thorp and DMR, The Information Paradox, McGraw Hill, 1998.

S. Tockey, Return on Software, Addison Wesley, 2004.

Economics-Driven Software Engineering Research (EDSER) web site: www.edser.org

MBASE web site : sunset.usc.edu/research/MBASE