An Initial Theory of Value-Based Software Engineering

Barry Boehm, Apurva Jain
USC-CSE-2005-502, February 2005

Abstract: This chapter presents an initial "4+1" theory of value-based software engineering (VBSE). The engine in the center is the stakeholder win-win Theory W, which addresses the questions of "which values are important?" and "how is success assured?" for a given software engineering enterprise. The four additional theories that it draws upon are utility theory (how important are the values?), decision theory (how do stakeholders' values determine decisions?), dependency theory (how do dependencies affect value realization?), and control theory (how to adapt to change and control value realization?). After discussing the motivation and context for developing a VBSE theory and the criteria for a good theory, the chapter discusses how the theories work together into a process for defining, developing, and evolving software-intensive systems. It also illustrates the application of the theory to a supply chain system example, discusses how well the theory meets the criteria for a good theory, and identifies an agenda for further research.

Keywords: adaptive control, benefits realization, control theory, decision theory, dependency theory, domain theories, game theory, risk/opportunity management, stakeholder win-win, Theory of Justice, Theory W, utility theory, value-based software engineering.

14.1 Introduction

14.1.1 Motivation and Context

The Preface and Chapter 21 provide general motivation for a value-based approach to software engineering. The particular motivation for developing and evolving a VBSE theory includes the following considerations:

- Understanding the "whys" of VBSE, as well as the "whats" and the "hows".
- Serving as an evaluation framework for VBSE practices.

- Providing principles for dealing with new software engineering situations (emergent requirements, rapid unpredictable change, commercial-off-the-shelf products (COTS), systems-of-systems “coopetition”, global cross-cultural software development).
- Providing a unifying framework for stakeholders to reason about software investments in terms of value created.
- Helping to assess the maturity of the VBSE field (and its theory).
- Managing expectations about the universality and formality of theories involving people and unpredictable change.
- Serving as a basis for continuing improvement of VBSE and its theory.

14.1.2 VBSE Theory Context

A VBSE theory needs to address all of the considerations of computer science theory, plus considerations involved in the managerial aspects of software engineering, plus considerations involved in the personal, cultural, and economic values involved in developing and evolving successful software-intensive systems. Although this sounds considerably more complex, we have found that the use of success-critical-stakeholder values to situate and guide technical and managerial decisions has actually made the job easier.

However, the need to deal with people considerations comes at a price with respect to the highly formalized theories of mathematics and science. These are able to be both formal and predictive because they rest on strong universal and timeless assumptions. Examples are that the flow of electricity through a conductor, the flow of air around an airfoil, or the flow of chemicals through a reactor will be the same ten years from now as they are now. However, basing a theory of software engineering on such assumptions as the flow of adoptions for a new software engineering product or the flow of data through an evolving software product being the same ten years from now will lead to theories with very short lifetimes. As a result, a VBSE theory will be less formal and, as we will see, will not be universal or timeless across a wide range of software situations, stakeholders, and products.

On the other hand, formalized theories of software engineering that attempt to abstract out the people factors (e.g., programming calculi [Jones, 1980]) have tremendous difficulties in dealing with situations that are not universal (e.g., skill factors [Juristo et. al., 2005]) or timeless (e.g., Maslow need hierarchies [Maslow, 1954]). A good treatment of these tensions between modernist (written, universal, general, timeless) and postmodern (oral, particular, local, timely) approaches to explanatory theories is provided in Cosmopolis [Toulmin, 1990].

14.1.3 Chapter Objectives, Approach, and Definitions

In this context, the objectives of this Chapter are to: (1) present an initial theory of VBSE; (2) illustrate it via an example; (3) evaluate it with respect to criteria for a good theory; and (4) identify an agenda for further research.

The rest of this Section will provide working definitions for “theory” and “criteria for a good theory”, based on candidate definitions in the literature and our experiences in applying and evolving the stakeholder win-win Theory W [Boehm-Ross, 1989] since 1989.

A Working Definition of “Theory’

There are numerous definitions of “theory” to consider. They range from highly formal definitions such as, “A theory is a system of general laws that are spatially and temporally unrestricted and nonaccidental” [Hempel-Oppenheim, 1960; Danto-Morgenbesser, 1960], to relatively informal definitions such as “A theory is any coherent description or explanation of observed or experienced phenomena” [Gioia-Pitre, 1990]. Our working definition of “theory” here follows [Tarraco, 1994] in attempting to capture the strengths of both formal and informal approaches:

> “A theory is a system for explaining a set of phenomena that specifies the key concepts that are operative in the phenomena and the laws that relate the concepts to each other”.

Our theorems about success criteria for software-intensive enterprises will not be “spatially and temporally unrestricted and nonaccidental”. This is because software-intensive enterprises and their success are subject to multiple concurrent influences, some of which are unpredictable. For example, a project that is poorly requirements-engineered and architected, poorly managed, behind schedule, and over budget can still turn into a great success with the appearance of just the right new COTS product to satisfy stakeholder needs.

The reverse is true as well: “The best laid plans o’ mice an’ men Gang aft agley” through unforeseeable external circumstances [Burns, 1785].

Criteria for a Good Theory

Besides the references on “theory” above, we found good sets of criteria for a good theory in [Patterson, 1983] (importance, preciseness and clarity, parsimony or simplicity, comprehensiveness, operationality, empirical validity or verifiability, fruitfulness, practicality) and [Bacharach, 1989] (falsifiability, utility for explanation and prediction). In comparing these with our previous criteria for evaluating Theory W (simple, general, specific, accurate, analytic, predictive, diagnostic, synthetic, generative), we converged on the following composite list of major criteria:

1. Utility. Particularly in a value-based context, does the theory favor addressal of critical success factors rather than trivia?
2. Generality. Does the theory cover a wide range of situations and concerns (procedural, technical, economic, human)?
3. Practicality. Does the theory help address users’ practical needs with respect to prediction, diagnosis, synthesis of solutions, generation of good practices, and explanation?
4. Preciseness. Does the theory provide situation-specific and accurate guidance?
5. Parsimony. Does the theory avoid excess complexity? Is it simple to understand, learn, and apply?
6. Falsifiability. Is the theory coherent enough to be empirically refuted?

We will address these criteria as we explain the VBSE theory in Section 14.2. After applying it to an example in Section 14.3, we will review how well the criteria were satisfied in Section 14.4. Section 14.5 will summarize our conclusions, and identify areas for further research.

14.2 A “4+1” Theory of Value-Based Software Engineering

Figure 14.1 summarizes the “4+1” theory of VBSE. Credit is due to Philippe Kruchten for originating this model form in the area of software architecture [Kruchten, 1999]. The engine in the center is the success-critical stakeholder (SCS) win-win Theory W, which addresses the questions of “what values are important?” and “how is success assured?” for a given software engineering enterprise. The four additional theories that it draws upon are utility theory (how important are the values?), decision theory (how do stakeholders' values determine decisions?), dependency theory (how do dependencies affect value realization?), and control theory (how to adapt to change and control value realization?).

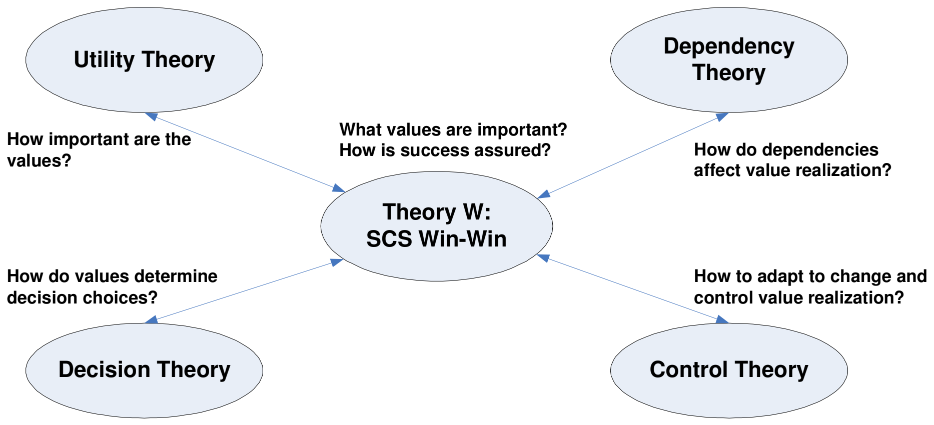

Utility Theory
Dependency Theory
How important are the values?
What values are important? How is success assured?
How do dependencies affect value realization?
Theory W: SCS Win-Win
How do values determine decision choices?
How to adapt to change and control value realization?
Decision Theory
Control Theory

Figure 14.1. The "4+1" Theory of VBSE: Overall Structure

14.2.1 The Central Engine: Theory W

The core of Theory W is the

Enterprise Success Theorem: Your enterprise will succeed
if and only if
It makes winners of your success-critical stakeholders.

An informal proof follows. As discussed in Section 14.1.2, VBSE theorems and proofs are less formal than those in such areas as mathematics and physics.

Proof of "if":

1. Everyone significant is a winner.
2. Nobody significant is left to complain.

Proof of “only if”:

1. Nobody wants to lose.
2. Prospective losers will refuse to participate, or will counterattack.
3. The usual result is lose-lose.

The proof of “if” is reasonably clear. The proof of “only if” may not be so clear, so we illustrate it in three frequently-occurring examples of the primary stakeholders in an enterprise involving a customer contracting with a developer for a software system that will benefit a community of users, as shown in Figure 14.2.

Proposed Solution	“Winner”	Loser
1. Quick, Cheap, Sloppy Product	**Developer & Customer**	**User**
2. Lots of “bells and whistles”	**Developer & User**	**Customer**
3. Driving too hard a bargain	**Customer & User**	**Developer**

Figure 14.2. Win-lose Generally Becomes Lose-Lose

In Case 1, the customer and developer attempt to win at the expense of the user by skimping on effort and quality. When presented with the product, the user refuses to use it, leaving everyone a loser with respect to their expectations.

In Case 2, the developer and user attempt to win at the expense of the customer (usually on a cost-plus contract) by adding numerous low-value “bells and whistles” to the product. When the customer’s budget is exhausted without a resulting value-adding product, again everyone is a loser with respect to their expectations.

In Case 3, the user and customer compile an ambitious set of features to be developed and pressure competing developers to bid low or lose the competition. Once on contract, the surviving bidder will usually counterattack by colluding with the user or customer to convert the project into Case 2 (adding user bells and whistles with funded Engineering Change Proposals) or Case 1 (saying, for example, “The contract specifies user-friendly error messages. For my programmers, a memory dump is a user-friendly error message and thus is a contractually compliant deliverable”). Again, everyone is a loser with respect to their expectations.

14.2.2 Achieving and Maintaining a Win-Win State: The Four Supporting Theories

However, the Enterprise Success Theorem does not tell us how to achieve and maintain a win-win state. This requires the

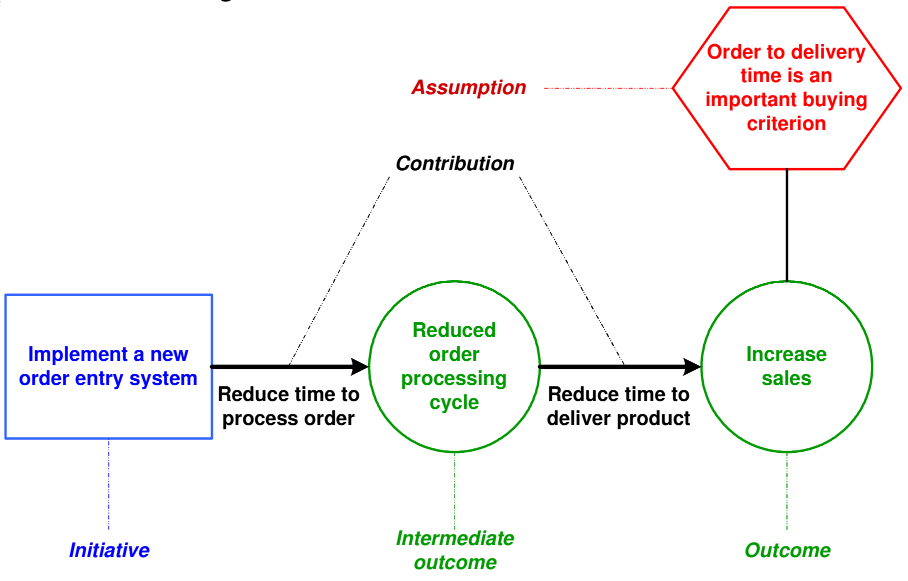
Win-Win Achievement Theorem: Making winners of your success-critical stakeholders requires:

1. ***Identifying all of the success-critical stakeholders (SCSs).***
2. ***Understanding how the SCSs want to win.***
3. ***Having the SCSs negotiate a win-win set of product and process plans.***
4. ***Controlling progress toward SCS win-win realization, including adaptation to change.***

14.2.2.1 Identifying all of the SCSs: Dependency Theory

Identifying all of the SCSs is in the province of dependency theory. A key technique is the Results Chain [Thorp, 1998].

Figure 14.3 shows a simple results chain provided as an example in The Information Paradox [Thorp, 1998]. It establishes a framework linking Initiatives that consume resources (e.g., implement a new order entry system for sales) to Contributions (not delivered systems, but their effects on existing operations) and Outcomes, which may lead either to further contributions or to added value (e.g., increased sales). A particularly important contribution of the Results Chain is the link to Assumptions, which condition the realization of the Outcomes. Thus, in Figure 14.3, if order to delivery time turns out not to be an important buying criterion for the product being sold (e.g., stockable commodities such as soap or pencils), the reduced time to deliver the product will not result in increased sales. The Results Chain provides a valuable framework by which software project members can work with clients to identify additional non-software initiatives that may be needed to realize the potential benefits enable by the software/IT system initiative. These may also identify some additional success-critical stakeholders who need to be represented and "bought into" the shared vision.



Assumption
Order to delivery time is an important buying criterion
Contribution
Implement a new order entry system
Reduce time to process order
Reduced order processing cycle
Reduce time to deliver product
Increase sales
Initiative
Intermediate outcome
Outcome

Figure 14.3. Results Chain

For example, the initiative to implement a new order entry system may reduce the time required to process orders only if an additional initiative to convince the sales people that the new system will be good for their careers and to train them in how to use the system effectively is pursued. If the order entry system is so efficiency-optimized that it doesn't keep track of sales credits, the sales people will fight using it, so increased sales may also require adding capabilities to keep track of sales credits so sales people will want to use the new system.

Further, the reduced order processing cycle will reduce the time to deliver products only if additional initiatives are pursued to coordinate the order entry system with the order fulfillment system. Some classic cases where this didn't happen were the late deliveries of Hershey's Halloween candy [Carr, 2002] and Toys'R'Us' Christmas toys.

Such additional initiatives need to be added to the Results Chain. Besides increasing its realism, this also identifies additional success-critical stakeholders (sales people and order fulfillment people) who need to be involved in the system definition and development process. The expanded Results Chain involves these stakeholders not just in a stovepipe software project to satisfy some requirements, but in a <u>program</u> of related software and non-software initiatives focused on value-producing end results.

The Hershey's and Toys'R'Us examples show that failing to identify an SCS such as the order-fulfillment organization generally leads to failure. This makes identifying all of the SCSs essentially a necessary condition for WinWin achievement. Here also though, as discussed in Section 14.1.3, our informal theorems do not guarantee failure; neglectful projects can still "get lucky" and succeed if the developer or COTS vendor happens to have the needed additional features at hand. But betting on luck is not a recommended strategy.

Actually, dependency theory covers the full range of theories that help reason about how dependencies affect value realization. These include theories about product dependencies such as physics, computer science, and architectural theories [Alexander, 1979; Rechtin, 1991]; theories about process dependencies, such as scheduling and concurrency theories; theories about stakeholder interdependencies, such as sociology and organization theories [Parsons, 1977; March-Simon, 1958; Argyris, 1978; Rifkin, 2004] and theories about product, process, and stakeholder interdependencies, such as economic, management, and system engineering theories [Simon, 1969; Cyert-March, 1963; Marschak-Radner, 1972; Churchman et. al., 1957; Wymore, 1967; Checkland, 1981]. Other examples will be provided in Section 14.4.

14.2.2.2 Understanding how the SCSs want to win: Utility Theory

<u>Understanding how the SCSs want to win</u> is in the province of <u>utility theory</u> [Dupuit, 1844; Debreu 1959; Fishburn, 1982]. Misunderstanding SCS utility functions does not guarantee failure if an enterprise happens to get lucky. But again, understanding how the SCSs want to win is essentially a necessary condition for WinWin achievement. Utility theory also has several branches such as the satisficing theory of bounded rationality [Simon, 1957], multi-attribute utility theory [Keeney-Raiffa, 1976], and its situation-dependent aspects such as the Maslow need hierarchy [Maslow, 1954] stating that lower-level needs (food and drink; safety and security) have dominant utilities when unsatisfied and negligible utilities when satisfied.

14.2.2.3 Having the SCSs negotiate win-win plans: Decision Theory

Having the SCSs negotiate win-win plans is in the province of decision theory. Decision theory also has many aspects such as negotiation theory [Raiffa, 1982; Fisher-Ury, 1981], game theory [von Neumann-Morgenstern, 1944; Luce-Raiffa, 1957]], multi-attribute decision theory [Keeney-Raiffa, 1976], statistical decision theory and the buying of information to reduce risk [Blackwell-Girshick, 1954], real options theory as discussed in Chapters 11 and 41, and the Theory of Justice [Rawls, 1971] discussed in Chapter 15.

Getting to a Win-Win Decision

Navigating through all of these decision options is rather complex. One aid in the stakeholder win-win negotiation context is the win-win equilibrium theory in [Boehm-Bose, 1994] and [Lee, 1996]. As illustrated in Figure 14.4, the win-win negotiation model begins with the success-critical stakeholders (SCSs) identifying their win conditions (or value propositions) about the system to be developed and evolved. The SCSs can include considerably more classes than users, customers, and developers. Additional SCS classes can include maintainers, administrators, interoperators of co-dependent systems, testers, marketers, venture capitalists, and, as in Section 15.8 on software engineering ethics, representatives of the least-advantaged people whose health, lives, or quality of life may be affected by the system.

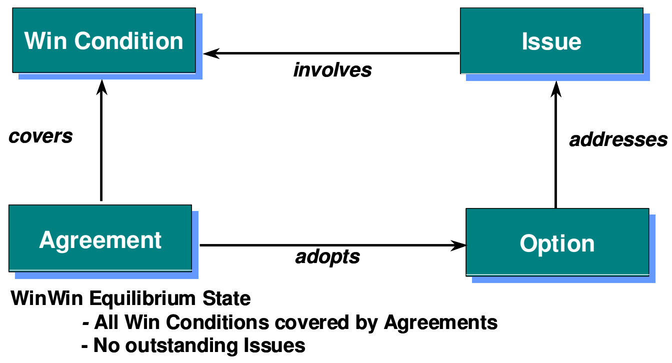


Figure 14.4. WinWin Negotiation Model

Besides Win Conditions, the win-win negotiation model in Figure 14.4 involves Agreements (in which all the SCS agree to adopt a win condition or an option), Issues (in which an SCS can identify a conflict between their and others' win conditions), and Options (proposals for resolving issues by expanding the option space). Agreements can also be reached by having the SCSs agree to adopt an option to resolve an issue.

The WinWin equilibrium state in Figure 14.4 holds when all the win conditions are covered by agreements, and there are no outstanding issues. At the beginning of a negotiation, this is true by default. As soon as a stakeholder enters a win condition, the other stakeholders can all accept it via an agreement, in which case the WinWin equilibrium state still holds, or some stakeholder enters an issue and an associated conflicting win condition. The negotiation then leaves the WinWin equilibrium state, and the stakeholders attempt to formulate options to resolve the issue. For example, if the conflicting win conditions are to have the system run on a Windows platform and a Unix platform, an acceptable option might be to build the system to run on Java Virtual Machine (JVM). The negotiation proceeds until all of the stakeholders' win conditions are entered and the WinWin equilibrium state is achieved, or until the stakeholders agree that the project should be disbanded because some issues are irresolvable. In such situations, it is much preferable to determine this before rather than after developing the system. And in terms of the WinWin Achievement Theorem, this also makes negotiating win-win plans a necessary condition for WinWin achievement.

14.2.2.4. Controlling progress toward SCS win-win realization: Control Theory

Controlling progress toward SCS win-win realization is in the province of control theory. As summarized in [Brogan, 1974] the necessary conditions for successful enterprise control are observability (the ability to observe the current enterprise state), predictability (the ability to predict whether the enterprise is heading toward an unacceptable state), controllability (the ability to redirect the enterprise toward an acceptable near-term state and a successful end state), and stability (the avoidance of positive feedback cycles that cause control systems to overcompensate and become unstable).

The application of these necessary conditions to people-intensive software enterprises does not permit the use of observability and controllability equations as precise as those in aerospace and electrical engineering, but they capture most of the wisdom provided by software management thought leaders. Examples are “You can’t control what you can’t measure” [DeMarco, 1982]; “If you don’t know where you’re going, a map won’t help” [Humphrey, 1989]; and “Giving people rewards for finding bugs is more likely to increase bug production than to increase software quality” [Adams, 1995].

Particularly for VBSE, it is more important to apply control theory principles to the expected value being realized by the project rather than just to project progress with respect to plans. Traditional “earned value” systems have their uses, but they need to be complemented by business-value and mission-value achievement monitoring and control systems as discussed in Chapter 12 and [Boehm-Huang, 2003]. These involve the use of risk management; adaptive control functions such as market watch and plan renegotiation; and multi-criteria control mechanisms such as BTOPP [Scott Morton, 1991; Thorp, 1998] and balanced scorecards [Kaplan-Norton, 1996]. Particularly in an era of increasing rates of change, this makes both traditional and adaptive control [Highsmith, 2000] necessary conditions for software enterprise success in terms of the WinWin Achievement Theorem.

14.3 Using and Testing the 4+1 VBSE Theory: Process Framework and Example

In this Section, we present in Figure 14.5 a seven-step process-oriented expansion of the 4+1 VBSE theory framework shown in Figure 14.1, and will then apply it to a supply chain management system development example. In Section 14.4, we will use the results to evaluate how well it addresses the criteria for a good theory presented in Section 14.1.3.

Step 1 of the process starts with a protagonist or change agent who provides the motivating force to get a new project, initiative, or enterprise started. As shown in Table 14.1, protagonists can be organization leaders with goals, authority, and resources, entrepreneurs with goals and resources, inventors with goals and ideas, or consortia with shared goals and distributed leadership and resources.

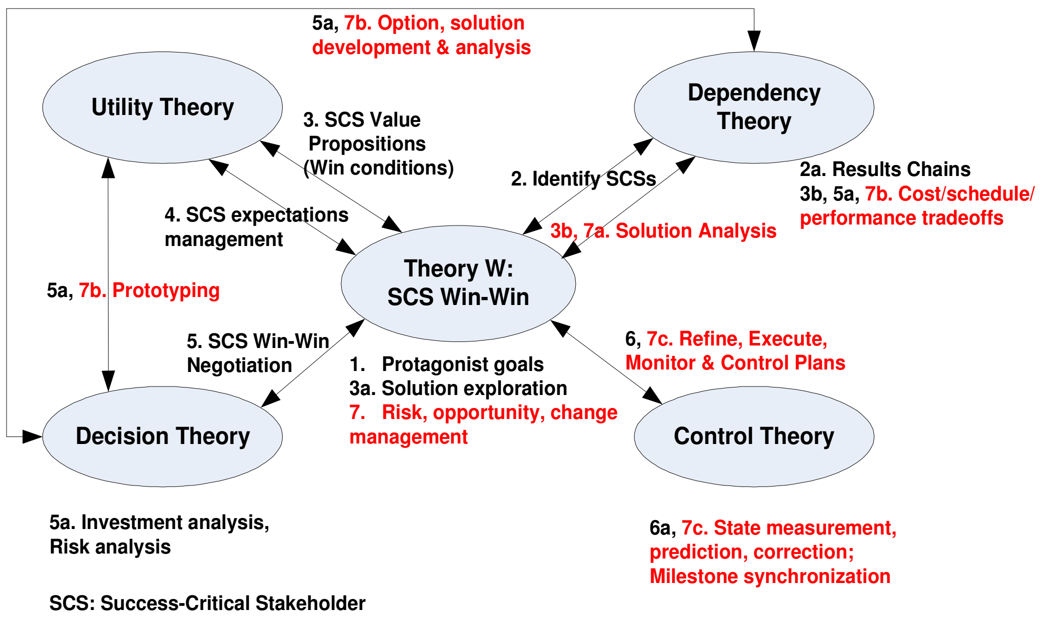

5a, 7b. Option, solution development & analysis
Utility Theory
3. SCS Value Propositions (Win conditions)
Dependency Theory
2. Identify SCSs
2a. Results Chains
3b, 5a, 7b. Cost/schedule/ performance tradeoffs
4. SCS expectations management
3b, 7a. Solution Analysis
Theory W: SCS Win-Win
5a, 7b. Prototyping
5. SCS Win-Win Negotiation
1. Protagonist goals
3a. Solution exploration
7. Risk, opportunity, change management
6, 7c. Refine, Execute, Monitor & Control Plans
Decision Theory
Control Theory
5a. Investment analysis, Risk analysis
6a, 7c. State measurement, prediction, correction; Milestone synchronization
SCS: Success-Critical Stakeholder

Figure 14.5. Process-Oriented Expansion of 4+1 VBSE Theory Framework

Table 14.1 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)

Each class of protagonist will take a somewhat different approach in visiting the seven main steps in Figure 14.5 to create and sustain a win-win combination of SCSs to achieve their goals. In this Section, we will trace the approach taken by a leader whose goals involve a combination of opportunities and problems, who has the authority and resources to address the goals, and who is open to different ideas for addressing them. She is Susan Swanson, an experienced MBA-holding executive, former bicycling champion, and newly-hired CEO of Sierra Mountainbikes, Inc. (a fictitious company representative of two similar companies with less successful projects).

14.3.1 Sierra Mountainbikes Opportunities and Problems

Susan began by convening her management and technology leaders, along with a couple of external consultants, to develop a constructive shared vision of Sierra Mountainbikes' primary opportunities and problems. The results determined a significant opportunity for growth, as Sierra's bicycles were considered top quality and competitively priced. The major problem area was in Sierra's old manual order processing system. Distributors, retailers, and customers were very frustrated with the high rates of late or wrong deliveries; poor synchronization between order entry, confirmation, and fulfillment; and disorganized responses to problem situations. As sales volumes increased, the problems and overhead expenses continued to escalate.

In considering solution options, Susan and her Sierra team concluded that since their primary core competence was in bicycles rather than software, their best strategy would be to outsource the development of a new order processing system, but to do it in a way that gave the external developers a share in the system's success. As a result, to address these problems, Sierra entered into a strategic partnership with eServices Inc. for join development of a new order processing and fulfillment system. eServices was a growing innovator in the development of supply chain management systems (in terms of Table 14.1, an inventor with ideas looking for leaders with compatible goals and resources to apply their ideas).

14.3.2 Step 2: Identifying the Success-Critical Stakeholders (SCSs)

Step 2 in the process version of the VBSE theory shown in Figure 14.5 involves identifying all of the success-critical stakeholders involved in achieving project's goals. As seen in Figure 14.6, the Step 2a Results Chain jointly determined by Sierra and eServices, this includes not only the sales personnel, distributors, retailers, and customers involved in order processing, but also the suppliers involved in timely delivery of Sierra's bicycle components.

The Results Chain includes initiatives to integrate the new system with an upgrade of Sierra's supplier, financial, production, and human resource management information systems. The Sierra-eServices strategic partnership is organized around both the system's results chain and business case, so that both parties share in the responsibilities and rewards of realizing the system's benefits. Thus, both parties share a motivation to understand and accommodate each other's value propositions or win conditions and to use value-based feedback control to manage the program of initiatives.

This illustrates the "only if" part of the Enterprise Success Theorem. If Susan had been a traditional cost-cutting, short-horizon executive, Sierra would have contracted for a lowest-bidder order processing system using Case 3 in Figure 14.2, and would have ended up with a buggy, unmaintainable stovepipe order processing system and many downstream order-fulfillment and supplier problems to plague its future.

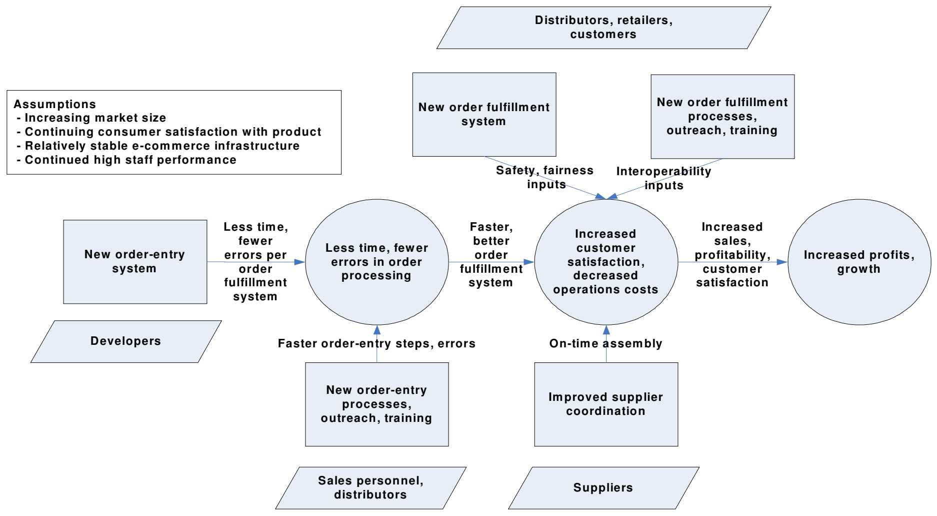

Distributors, retailers, customers
Assumptions
- Increasing market size
- Continuing consumer satisfaction with product
- Relatively stable e-commerce infrastructure
- Continued high staff performance
New order fulfillment system
New order fulfillment processes, outreach, training
Safety, fairness inputs
Interoperability inputs
New order-entry system
Less time, fewer errors per order fulfillment system
Less time, fewer errors in order processing
Faster, better order fulfillment system
Increased customer satisfaction, decreased operations costs
Increased sales, profitability, customer satisfaction
Increased profits, growth
Developers
Faster order-entry steps, errors
On-time assembly
New order-entry processes, outreach, training
Improved supplier coordination
Sales personnel, distributors
Suppliers

Figure 14.6. Results Chain for Sierra Supply Chain Management

In terms of the VBSE process in Figure 14.5, however, Sierra and eServices used the Results Chain form of Dependency Theory to identify additional SCSs (sales personnel, distributors, retailers, customers, suppliers) who also need to be brought into the SCS WinWin equilibrium state (fortunately, pollution and public safety are not major issues with bicycles, so a representative of the general public is not necessary).

14.3.3 Steps 3 and 4: Understanding SCS Value Propositions; Managing Expectations

As shown in Figure 14.7 (the first four steps in Figure 14.5), Step 3 (understanding all of the SCSs' value propositions or win conditions) primarily involves utility theory. But it also involves Theory W in reconciling SCS win conditions with achievable solutions (Step 3a), and various forms of dependency theory in conducting cost/schedule/performance solution tradeoff, and sensitivity analyses (Step 3b).

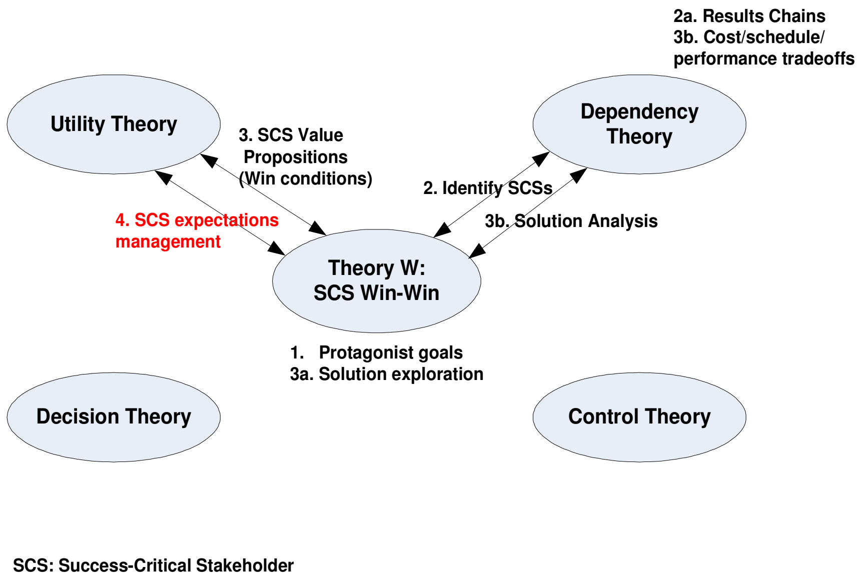


Figure 14.7. Process-Oriented Expansion of 4+1 VBSE Theory Framework (Steps 1 through 4)

For example, the suppliers and distributors may identify some complex exception reporting, trend analysis, and customer relations management features they would like to have in the system's Initial Operational Capability (IOC) in early 2005. However, the use of forms of dependency theory such as software cost and schedule estimation models may show that the dependency of IOC delivery schedule on IOC software size makes it unrealistic to try to develop the full desired feature set by the IOC date. In such a case, Sierra and eServices will have to revisit the SCSs' utility functions in Step 4 by showing them the cost and schedule model credentials and results, and asking them to expand their utility functions by prioritizing their desired features and participating in further solution exploration (a go-back to Step 3a) to achieve a win-win consensus on the top-priority subset of features to include in the IOC.

It may be in some cases that the SCSs' IOC needs are irreconcilable with the IOC schedule. If so, the SCSs may need to live with a later IOC, or to declare that a SCS win-win state is unachievable and to abort the project. Again, it is better to do this earlier rather than later. The particular considerations are discussed in more detail in a paper on the Schedule as Independent Variable (SAIV) process [Boehm et. al., 2002].

14.3.3 Step 5: SCSs Negotiate a WinWin Decision

Actually, the previous paragraph anticipates the content of Step 5, in which the SCSs negotiate a win-win decision to commit themselves to go forward. Once the SCSs have identified and calibrated their win conditions in Steps 3 and 4, the process of identifying conflicts or Issues among win conditions; inventing and exploring Options to resolve Issues; and converging on Agreements to adopt win conditions or Options proceeds as described in Section 14.2.2.3 and Chapter 36.

In a situation such as the Sierra supply chain project, the number of SCSs and the variety of their win conditions (cost, schedule, personnel, functionality, performance, usability, interoperability, etc.) means that multi-attribute decision theory will be involved as well as negotiation theory. Susan will also be concerned with investment theory or business case analysis to assure her stakeholders that the supply chain initiative will generate a strong return on investment. As many of the decisions will involve uncertainties (market trends, COTS product compatibilities, user interface choices), forms of statistical decision theory such as buying information to reduce risk will be involved as well.

User interface prototypes are actually ways of better understanding SCS utility functions, as indicated in Figure 14.5 by the arrow between decision theory and utility theory. The other components of Step 5a in Figure 14.5 involve other aspects of dependency theory, such as performance analysis, business case analysis, or critical-path schedule analysis. As also show in Figure 14.5, these analyses will often proceed at increasing levels of detail in supporting steps 3a, 5a, and 7a as the project proceeds into detailed design, development, integration, and test. Chapters 22, 24, 31, 32, 35, and 44 provide further detailed examples.

Figure 14.8 summarizes the business case analysis for the Sierra project, Dollar value are all in millions of 2004 dollars ($M) for simplicity. The analysis compares the expected sales and profits for the current system (columns 4, 5) and the new system (columns 7, 8) between 2004 and 2008, the cumulative increase in profits, investment cost, and resulting return on investment (columns 11-13), and expected improvements in other dimensions such as late delivery and customer satisfaction (columns 14-17). The bottom line is a strong 2.97 ROI, plus good expected outcomes in the customer satisfaction dimensions. More detail can be found in Chapter 12 and [Boehm-Huang, 2003].

		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

Figure 14.8. Expected Benefits and Business Case

The negotiations converge on a number of win-win agreements, such as involving the suppliers and distributors in reviews, prototype exercising, and beta-testing; having Sierra provide eServices with two of their staff members to work on the software development team; and agreeing on compatible data definitions for product and financial interchange. At one point in the negotiation, an unfortunate go-back is necessary when an Agreement on a product definition standard is reversed by the management of one of the distributors, who disclose that they are now committed to an emerging international standard. After some renegotiation, the other SCSs agree to this at some additional cost. But it brings up another necessary condition for successful win-win negotiations (and other collaborative vestures such as agile methods): that the stakeholder representatives be CRACK (collaborative, representative, authorized, committed, and knowledgeable) participants [Boehm-Turner, 2004]. Some other perspectives on win-win management are in [Waitley, 1985] and [Covey, 1989].

14.3.5 Steps 6 and 7: Planning, Executing, Monitoring, Adapting, and Controlling

As with the dependency analyses, project planning, executing, monitoring, adapting, and controlling proceed incrementally in increasing amounts of details, generally following a risk-driven spiral process. Questions such as "how much is enough planning, specifying, prototyping, COTS evaluation, business case analysis, architecting, documenting, verifying, validating etc.?" are best resolved by balancing the risk exposures of doing too little or too much. As Risk Exposure = Probability (Loss) * Value (Loss) is a value-based concept, risk balancing is integral to VBSE. See [Boehm-Turner, 2004] and [Port-Chen, 2004] for example "how much is enough?' analyses.

Value-based planning and control differs most significantly from traditional project planning and control in its emphasis on monitoring progress toward value realization rather than towards project completion. Particularly in an era of increasing rates of change in market, technology, organizational, and environmental conditions, there is an increasing probability that managing to a fixed initial set of plans and specifications will produce systems that are out of step and non-competitive with projects managing adaptively toward evolving value realization.

Perhaps the most provocative example is the traditional technique of "earned value management". It assigns "value" to the completion of project tasks and helps track progress with respect to planned budgets and schedules, but has no way of telling whether completing these tasks will add to or subtract from the business value or mission value of the enterprise. Example failure modes from this approach are systems that had to be 95% redeveloped on delivery because they failed to track evolving requirements [Boehm, 1973], and startup companies that fail to track closure of market windows.

If an organization has used steps 1-5 to identify SCSs, determine their value propositions, and develop business cases, it has developed the framework to monitor expected value realization, adjust plans, and control progress toward real SCS value achievement. Figure 14.9 shows how this is done for the Sierra project, based on the initial budgets, schedules, and business case in Figure 14.8. Value-based monitoring and control for Sierra requires additional effort in terms of technology watch and market watch, but these help Sierra to discover early that their in-transit-visibility (ITV) COTS vendor was changing direction away from Sierra's needs.

This enabled Sierra to adapt by producing a timely fallback plan, and to proactively identify and approach other likely ITV COTS vendors. The results, as shown in the ITV column and explained in the Risks/Opportunities column of Figure 14.9, was an initial dip in achieved ITV rating relative to plans, but a recovery to close to the originally planned value. The Risks/Opportunities column also shows a "new hardware competitor" opportunity found by market watch activities that results in a $200K hardware cost savings that mostly compensated for the added software costs of the ITV fallback. The use of prioritized requirements to drive value-based Pareto- and risk-based inspection and testing, as discussed in Chapter 21 and [Gerrard-Thompson, 2002], is another source of software cost savings.

The bottom-line results are a good example of multi-attribute quantitative/qualitative balanced-scorecard methods of value-based monitoring, adaptation, and control. They are also a good example of use of the necessary conditions for value-based control based on control theory. A traditional value-neutral "earned value" management system would fail on the criteria of business-value observability, predictability, and controllability, because its plans, measurements, and controls deal only with internal-project progress and not with external business-value observables and controllables. They also show the value of adaptive control in changing plans to address new risks and opportunities, along with the associated go-backs to revisit previous analyses and revise previous plans in Steps 7a, 7b, and 7c.

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 3/31/04	400 427		20 20	72 72	7.0 7.0			12.4 12.4	1.7 1.7	1.0 1.0	1.8 1.8	(1)
Core Capability Demo (CCD)	7/31/04 7/20/04	1050 1096								2.4*	1.0*	2.7*	(2)
Software Init. Op. Cap. (IOC)	9/30/04 9/30/04	1400 1532								2.7*	1.4*	2.8*	
Hardware IOC	9/30/04 10/11/04	3500 3432											(3)
Deployed IOC	12/31/04 12/20/04	4000 4041		20 22	80 88	8.0 8.6	0.0 0.6	-1.0 -.85	11.4 10.8	3.0 2.8	2.5 1.6	3.0 3.2	(4)
Responsive IOC	3/31/05 3/30/05	4500 4604	300 324						9.0 7.4	3.5 3.3	3.0 1.6	3.5 3.8	
Full Op. Cap. CCD	7/31/05 7/28/05	5200 5328	1000 946							3.5*	2.5*	3.8*	(5)
Full Op. Cap. Beta	9/30/05 9/30/05	5600 5689	1700 1851							3.8*	3.1*	4.1*	
Full Op. Cap. Deployed Release 2.1	12/31/05 12/20/05 6/30/06	6000 5977 6250	2200 2483	22 24	106 115	12.2 13.5	3.2 5.1	-.47 -.15	7.0 4.8	4.0 4.1	3.5 3.3	4.0 4.2	

(1) Increased COTS ITV risk, fallback identified.
(2) Using COTS ITV fallback; new HW competitor; renegotiating HW
(3) $200K savings from renegotiated HW.
(4) New COTS ITV source identified, being prototyped.
(5) New COTS ITV source initially integrated.
* Interim ratings based on trial use

Figure 14.9 Value-Based Expected/Actual Outcome Tracking

14.4 VBSE Theory Evaluation With Respect to Goodness Criteria

The Sierra example in Section 14.3 provides an opportunity to evaluate the VBSE theory with respect to the criteria for a good theory presented in Section 14.1.3.

Utility: Addressing Critical Success Factors. The Results Chain method in Step 2 identified missing success-critical initiatives and stakeholders that were the downfall of supply chain initiatives at Hershey's and Toys'R'Us. The risk-driven inspection and test approaches in Step 6 avoid wasting inspection and test time on trivial-value aspects of the system.

Generality: Covering procedural, technical, economic, and human concerns; covering small and large systems. The 7-step process and its ability to accommodate parallel activities and go-backs were sufficient to cover the Sierra project's procedural needs. Technical and economic concerns are addressed in the use of dependency theory for cost, schedule, performance, and business case analyses in Steps 3a, 5a, and 7b. Human concerns are the essence of Theory W and utility theory, and of the SCS negotiations in Step 5. The steps in the VBSE theory have worked well for several mid-sized supply chain and customer relations management systems similar to Sierra; for over 100 small real-client e-services projects at USC; and as a framework for addressing very large systems of systems in such areas as defense and air traffic control.

Practicality: Supporting practical needs for prediction, diagnosis, solution synthesis, good-practice generation, and explanation. The theory draws on a wide-variety of dependency models (e.g. cost, schedule, performance, quality) to predict outcomes. In a stable, well-understood environment, managing to the predictions usually produces a self-fulfilling prophecy. In less stable and less familiar situations such as the Sierra case study, dependency theory was able to diagnose risks such as missing stakeholders in Step 2, Theory W was able to support synthesis of SCS win-win solutions in Steps 3-5, and adaptive control theory was able to generate good value-achievement monitoring practices to support in-process diagnosis and re-synthesis in Steps 6-7. The control theory necessary conditions of observability and controllability were able to explain why traditional earned value systems would not have addressed and resolved these value-domain problems.

Preciseness: Providing situation-specific and accurate guidance. The theory is no more (and no less) accurate than its constituent theories in predicting outcomes of unprecedented situations, but it is able to provide situation-specific guidance, as shown in its application to the Sierra supply-chain project. Also, several examples were provided in Section 14.3 of how the theory would have generated different guidance in different situations, such as with the distributor management's reversal of a win-win agreement on a product definition standard in Step 5, and with the ITV COTS vendor's change of direction in Steps 6 and 7.

Parsimony: Avoiding excess complexity; ease of learning and application. The theory's use of risk management to determine "how much is enough" planning, specifying, testing, etc. helps avoid excess complexity and to make "everything as simple as possible, but no simpler" [Einstein, 1879-1955]. Its ease of learning and use has been tested mainly on USC's over 100 e-services projects. These are developed by teams of 5-6 MS-level students who learn the technologies as they go, and have a 92% success rate of on-time, satisfied-customer delivery [Boehm et al., 1998].

Flexibility: Ability to be empirically refuted. The case study identified a particular situation in which application of the theory could not produce a win-win solution, leading to a timely decision to cancel the project. This involved incompatible and non-negotiable SCS win conditions about Initial Operational Capability content and schedule in Steps 3 and 4. A similar outcome could have resulted from the distributor management change of direction in Step 5.

Actually, there are several other classes of situations in which our experience has shown that the win-win approach may not succeed. These are:

- People may disguise their true win conditions. In one situation, a stakeholder rejected a COTS product for being too expensive. When the price was lowered, the stakeholder said that some essential features were missing. When the vendor offered to supply the features at no extra cost, the true reason came out: the stakeholder had had bad dealings with the COTS vendor in the past.

- Some people like to win by making others losers. It is best to seek other partners when you encounter such people.

- You can't make omelets without breaking eggs. Many large-scale dams that benefited millions of people had to drown some other people's homes and villages. Generous payment can reduce the loss, but generally not eliminate it.

- Some situations have only one winner. A good example involves political elections, in which political parties are motivated to discredit and demonize candidates and platforms of other parties.

However, many apparent only-one-winner or zero-sum-game situations can be turned into win-win situations by expanding the option space. A good example is provided in Getting to yes [Fisher-Ury, 1981], in which a boundary-line location stalemate on ownership of the Sinai Desert between Egypt and Israel was resolved by creating a new option: the land was given back to Egypt, satisfying its territorial win condition, but it was turned into a demilitarized zone, satisfying Israel's security win condition. Other examples are provided in [Boehm-Ross, 1989].

14.5 Conclusions and Areas for Further Research

The VBSE theory presented above has been shown to apply well to a reasonably complex supply chain application. In other situations, versions of the theory have been successfully applied to over 100 small e-services applications, and to some very large software-intensive systems of systems.

The VBSE theory satisfies the main criteria for a good theory (utility, generality, practicality, preciseness, parsimony, and falsifiability) reasonably well, particularly when compared to other theories involving explanations of human behavior.

The theory identifies several fruitful areas for further research, Some, such as elaborations of aspects of utility theory, decision theory, and dependency theory to address particular VBSE issues are discussed in other chapters in this book. Others are identified in the VBSE agenda but not covered in the book, such as extensions of the theory to cover such areas as programming methodology, agile methods, quality assurance, COTS-based applications, software maintenance, and combinations of these and the other areas covered.

Another area we are exploring is the extension of the current theory to provide a theory of value-based systems engineering. The systems engineering field is inherently value-based, and shares many of the same challenges as software engineering, but also brings additional considerations of hardware phenomenology and hardware-software-peopleware tradeoffs into the arena.

Finally, as with all theories, the initial VBSE theory needs many more tests. The easiest tests to start with are tests of its ability to explain differences between success and failure on completed projects. Other tests that can be done right away are tests of its ability to generate good software engineering practices; an early example is in [Boehm-Ross, 1989].

Further analyses can be performed on its consistency with other theories, such as the chaos-type theories underlying agile and adaptive software development [Highsmith, 2000] or the theories underlying formal software development [Jones, 1980] and generative programming approaches [Czarnecki-Eisenecker, 2000].

Tests of utility, generality, practicality, preciseness, and parsimony basically involve trying to apply the theory in different situations, observing its successes and shortfalls, and generating improvements in the theory that improve its capability in different situations or uncover unstated assumptions that should be made explicit to limit its domain of dependable applicability. We hope that this initial presentation of the theory will be sufficiently attractive for people to give this option a try.

14.6 References

S. Adams, Dilbert Comic Strips, 1995.

C. Alexander, The Timeless Way of Building, Oxford University Press, 1979.

C. Argyris, Organizational Learning, Addison-Wesley, 1978.

B. Boehm, Software and Its Impact: A Quantitative Assessment, Datamation, May 1973, pp. 48-59.

B. Boehm and P. Bose, A Collaborative Spiral Software Process Model Based on Theory W, Proceedings, ICSP 3, IEEE, October 1994.

B. Boehm and L. Huang, Value-Based Software Engineering: A Case Study, IEEE Computer, March 2003, pp. 21-29.

B. Boehm and R. Turner, Balancing Agility and Discipline, Addison Wesley, 2004.

B. Boehm, and R. Ross, Theory-W Software Project Management: Principles and Examples, IEEE Trans. SW Engineering., July 1989, pp. 902-916.

B. Boehm, D. Port, L. Huang, and W. Brown, Using the Spiral Model and MBASE to Generate New Acquisition Process Models: SAIV, CAIV, and SCQAIV, CrossTalk, January 2002, pp. 20-25.

B. Boehm, A. Egyed, J. Kwan, D. Port, A. Shah, and R. Madachy, Using the WinWin Spiral Model: A Case Study, IEEE Computer, July 1998, pp. 33-44.

D. Blackwell and M. Girshick, Theory of Games and Statistical Decisions, Wiley, 1954.

W. Brogan, Modern Control Theory, Prentice Hall, 1974 (3rd ed., 1991).

R. Burns, To a Mouse, November 1785.

D. Carr, "Sweet Victory", Baseline, December 2002.

P. Checkland, Systems Thinking, Systems Practice, Wiley, 1981.

C. W. Churchman, R. Ackoff, and E. Arnoff, An Introduction to Operations Research, Wiley, 1957.

S. Covey, The Seven Habits of Highly Successful People, Fireside/Simon & Schuster, 1989.

R. M. Cyert and J.G. March, A Behavioral Theory of the Firm, Prentice Hall, 1963.

K. Czarnecki and U. Eisenecker, Generative Programming: Methods, Tools, and Applications, Addison-Wesley, 2000

A. Danto and S. Morgenbesser (eds.), Philosophy of Science, Meridian Books, 1960.

G. Debreu, Theory of Value, Wiley, 1959.

T. DeMarco, Controlling Software Projects, Yourdon Press, 1982.

J. Dupuit, On the Measurement of the Utility of Public Works, Translated by R. H. Barback, International Economic Papers 2:83-110, 1844 (1952).

P. C. Fishburn, The Foundations of Expected Utility, Dordrecht, 1982.

R. Fisher and W. Ury, Getting To Yes: Negotiating Agreement Without Giving In, Houghton Mifflin, 1981.

P. Gerrard and N. Thompson, Risk-Based E-Business Testing, Artech House, 2002.

D. A. Gioia and E. Pitre, Multi-paradigm perspectives on theory building, Academy of Management Review, 15, pp. 584-602, 1990.

C. G. Hempel and P. Oppenheim, Problems of the Concept of General Law, in (eds.) A. Danto and S. Mogenbesser, Philosophy of Science, Meridian Books, 1960.

J. Highsmith, Adaptive Software Development, Dorset House, 2000.

J. Highsmith, Agile Software Development Ecosystems, Addison Wesley, 2002.

W. S. Humphrey, Managing the Software Process, Addison-Wesley, 1989.

C. B. Jones, Software development: A rigorous approach, Prentice Hall, 1980.

N. Juristo, A. Moreno, and S. Acuna, A Software Process Model Handbook for Incorporating People's Capabilities, Kluwer, 2005.

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

R. L. Keeney and H. Raiffa, Decisions with Multiple Objectives: Preferences and Value Tradeoffs, Cambridge University Press, 1976.

P. Kruchten, The Rational Unified Process: An Introduction, Addison Wesley, 1999.

M. J. Lee, Foundations of the WinWin Requirements Negotiation System, Ph.D. dissertation, University of Southern California, 1996.

R. D. Luce and H. Raiffa, Games and Decisions, John Wiley, 1957.

J. March and H. Simon, Organization*s*, Wiley, 1958.

J. Marschak and R. Radner, Economic Theory of Teams, Yale University Press, 1972.

A. Maslow, Motivation and Personality, Harper, 1954

T. Parsons, Social Systems and the Evolution of Action Theory, The Free Press, 1977.

C. H. Patterson, Theories of counseling and psychotherapy, Harper and Row, 1983.

Daniel Port and Scott Chen, Assessing COTS Assessment: How Much Is Enough?, ICCBSS 2004 Proceedings, Springer, 2004.

H. Raiffa, The Art and Science of Negotiation, Belknap/Harvard U. Press, 1982.

J. Rawls, A Theory of Justice, Belknap/Harvard U. Press, 1971, 1999.

E. Rechtin, Systems Architecting: Creating and Building Complex Systems, Prentice-Hall, 1991.

S. Rifkin, The Parsons Game: The First Simulation of Talcott Parsons' Theory of Action, Ph.D. dissertation, George Washington University, 2004.

M. Scott Morton, The Corporation of the 1990s: Information Technology and Organization Transformation, Oxford University Press, 1991.

H. Simon, The Science of the Artificial, MIT Press, 1969.

H. Simon, Models of Man, Wiley, 1957.

J. Thorp and DMR's Center for Strategic Leadership, The Information Paradox: Realizing the Benefits of Information Technology, McGraw-Hill, 1998.

R. J. Torraco, Theory-building research methods, in R. A. Swanson & E. F. Holton III (eds.), Human resource development handbook: Linking research and practice pp. 114–137, Berrett-Koehler, 1997.

S. Toulmin, Cosmopolis: The Hidden Agenda of Modernity, U. of Chicago Press, 1992 reprint edition.

J. von Neumann and O. Morgenstern, Theory of Games and Economic Behavior, Princeton University Press, 1944.

D. Waitley, The Double Win, Berkley, 1985.

A. W. Wymore, A Mathematical Theory of Systems Engineering: The Elements, Wiley, New York, 1967.

Author Biography

Barry Boehm is the TRW Professor of Software Engineering and Director of the Center for Software Engineering at USC. His current research interests include software process modeling, software requirements engineering, software architectures, software metrics and cost models, software engineering environments, and value-based software engineering. His contributions to the field include the Constructive Cost Model (COCOMO), the Spiral Model of the software process, and the Theory W (win-win) approach to software management and requirements determination. He is a Fellow of the primary professional societies in computing (ACM), aerospace (AIAA), electronics (IEEE), and systems engineering (INCOSE), and a member of the US National Academy of Engineering.

Apurva Jain is a candidate Ph.D. student in the Computer Science department at the University of Southern California. His research interests include software management and economics, software architecture, and value-based software engineering.