User Involvement in Project Success

Stanley H. Stahl

A basic principle for implementing a sustainable software productivity improvement program is to make everyone a winner. A systems project that comes in on a budget and schedules and meets user requirements make winners out of users, senior management, and the IS department. Making winner of people helps get their commitment and involvement as well as helps them overcome their natural resistance to change. It is a critical component to quality improvement in pioneer W. Edward Deming’s precept of putting to work to accomplish quality improvement.

Theory W is a way to make every one a winner. This theory was developed by Barry Boehm in the late 1980s. At the time, Boehm was chief scientist in TRW’s defense systems group and a professor of computer science at the University of California, Los Angeles. In their paper, “Theory-W Software Project Management: Principles and Examples,” Boehm and Rony Ross presented a unifying theory of software project management that is simultaneously simple, general, and specific. In the introduction of this paper, Boehm and Ross wrote:

“The software project manager’s primary problem is that a software project needs simultaneously satisfy a variety of constituencies: the users, the customers, the development team and management ... Each of these constituencies has its own desires with respect to the software project...These desires create fundamental conflicts when placed together ... These conflicts are at the root of most software project management difficulties—both at the strategic level (e.g., setting goals, establishing major milestones, and responsibilities) and at the tactical level (e.g., resolving day-to-day conflicts, prioritizing assignments, and adapting to changes).”

Theory W is a way to help project managers cope with the difficulty of simultaneously satisfying different constituencies. Theory W has one simple but very far-reaching principle: Make everyone a winner by setting up win-win conditions for everyone.
THEORY W: BACKGROUND AND BASICS

Theory W contrasts with such theories on management as Theory X, Theory Y, and Theory Z. The Theory X-approach to management originated in the work of Frederick Taylor, who was active at the beginning of the century. Taylor contended that the most efficient way to approach work was to organize jobs in an well-orchestrated sequence of efficient and predictable tasks. Management’s responsibility was to keep the system running smoothly; this task was often accomplished by coercing and intimidating workers.

For obvious reasons, Taylor’s Theory X is inappropriate for managing software projects. Theories Y and Z, dating from approximately 1960 and 1980, respectively, were intended as alternatives to Theory X. Theory Y’s perspective is that management must stimulate creativity and initiative, which are both important qualities for a quality software project. The difficulty with Theory Y, however, is that it provides inadequate mechanisms for identifying and resolving conflicts.

Theory Z seeks to improve Theory Y by emphasizing the development of shared values and building consensus. The problem with Theory Z is that consensus may not always be possible or desirable; this can be the case with different constituencies that have their own unique set of individual constraints and requirements.

If the Theory-X manager is an autocrat, the Theory-Y manager a coach, and the Theory-Z manager a facilitator, then the Theory-W manager is a negotiator. The manager in the Theory-W model must proactively seek out win-lose and lose-lose conflicts and negotiate them into win-win situations. Delivering software systems while making winners of all the stakeholders seems, at first glance, to be hopelessly naïve.

Users want systems delivered immediately and they want them with all the bells and whistles imaginable. Management not only want systems delivered on schedule and within budget, they also want a short schedule and a low budget. Developers want technical challenges and opportunities for professional growth, and they often do not want to document their work. Maintainers want well-documented systems with few bugs and the opportunity for a promotion out of maintenance. How can a project manager expect to successfully negotiate the conflicting needs of all constituents?

IMPORTANCE OF NEGOTIATING

Although it may seem like a naïve theory, there is an accumulation of evidence that Theory W works. In fact, Theory W is coming to be seen as fundamental to project success. The reason lies in the character of a win-win
negotiation. The objective in a win-win negotiation is for all parties to recognize each other’s specific needs and to craft a resolution that allows all participant to share in getting their needs met. This is very different from traditional styles of negotiation, which are too often win-lose.

In the absence of an explicit commitment to foster win-win relationships, software projects have the capability of becoming win-lose. For example, building a quick but bug-laden product may represent a low-cost win for an over-pressured development organization but it is a loss for the users. Alternatively, when management and users force developers to add extra features without giving the development organization the time and resources needed to develop the extra features, the result may be a win for users and a loss for developers. Software maintenance personnel often lose as management, developers, and users fail to ensure that software is well-documented and easily maintainable.

At their worst, software projects can become lose-lose situations where no one wins. It is common for management to set unreasonable schedule expectations, and as a result, the development department tries to catch up by adding more and more people to the project. The result is, all too often, a poor product that comes in over cost and over schedule. In this case, everyone loses.

The Cost of Not Negotiating

The following example illustrates how ignoring Theory W affects a project. Although this example is fictional, it is based on actual experience.

A lose–lose project. A growing retailer had just hired a new chief information officer. The new CIO was given the charter to modernize the company’s antiquated information systems but had been explicitly told by the CEO the budgets were extremely limited and that the new systems would have to be implemented as the size of the IS staff was decreased.

The first task was to conduct a user-needs survey, which indicated that, except for the payroll systems, all of the company’s existing systems were inadequate. The inventory control system was barely usable, there was no integration between different systems, and each system served at best, only the limited needs of the department for which it had been designed. The survey also indicated that most users were unaware of the potential productivity boost that up-to-date information systems could give to the company’s business.

After the survey had been analyzed, the following four recommendations were made:
• All existing systems should be replaced by a client/server systems capable of supplying timely and accurate information to both operational personnel and senior management.
• The changeover should be implement in stages. The first system should be a relatively simple, low-risk, standalone application.
• The system should be procured from an outside contractor, preferably a software house that has a package that could be used with minimal modifications.
• Training should be provided to middle managers to enable them to better guide the IS department in implementing the new systems.

**Basics were ignored.** These recommendations were enthusiastically approved. A request for proposal (RFP) for a new inventory distribution management control system, everyone’s favorite candidate to be implemented first, was requested. However, nothing was said about budgets, schedules, or personnel needs, and in their enthusiasm, everyone seemed to forget about training.

**Lack of user input.** The RFP was developed with little input from distribution personnel, so it was rather open-ended and not very explicit. The result was that the IS group received from outside contractors eight responses ranging in price from $70,000 to $625,00. After lengthy negotiations, a contract was awarded to a single company that would provide both hardware and software for the new inventory control system.

**Unclear RFP.** The contractor was a leader in inventory management systems, though its largest account was only half the size of the Retail Company’s. To land the account, the contractor promised to make any necessary modifications to the system free-of-charge. The contractor’s reading the RFP led its developers to believe that there was little technical risk in the promise.

The contractor’s interpretation of the RFP was wrong. Although the contract called for the system to be up and running in six months at a cost of $240,000, a year later the contractor was still working on changing the system to meet the client’s needs.

Neither was the retailer’s IS department experiencing a good year. It was continually at odds with both distribution personnel and the contractor over the capabilities of the new inventory system. The users kept claiming that they system was not powerful enough to meet their needs, while the contractor argued that the system was in use by more than 200 satisfied companies.

**Costly failure.** At the end of the acrimonious year, the contractor and the retailer agreed to cancel the project. In the course of the year, the contractor was paid more that $150,000 and the contractor estimated that its
programming staff had spent more than 10-worker-months modifying the system to meet the clients needs. The retailer estimated that it had invested the time of one senior analyst as well as several hundreds of hours of distribution personnel.

HOW THEORY WOULD HAVE HELPED

Losers and the Consequences
The most apparent source of difficulty on the project was the explicit win–lose contract established between the retailer and the contractor. By requiring the contractor to cover any expenses incurred in modifying the system, the retailer set up a situation in which the contractor would only make the changes reluctantly. This reduced the likelihood that distributors would get the modifications they needed and increased the likelihood that changes would be made in a slap-dash way, with too little attention paid to quality.

The IS department’s relationship with senior management was also win–lose. There was little likelihood that the CIO could emerge victorious. The system had to be brought in on time and within budget, though the user community was inadequately trained to help properly identify its need and requirements. The result was that neither the retailer nor the contractor had an adequate handle on the inventory system’s requirement and were consequently unable to adequately budget or schedule the system’s implementation. The IS department’s situation was made worse because senior management wanted to decrease the size of the IS staff.

The users lost the most. Not only did they not receive the system they needed and had been promised, but they wasted time and money in diverting attention from their primary jobs to help develop the new system. Both the retailer’s and the contractor’s developers lost the time they had invested in a failed project, the ability to grow professionally, and the opportunity to work on a successful project.

From a Process Perspective

Fault lies in both contracting process and the systems requirement management process by which the retailer and the contractor defined and managed systems requirements. These front-end processes are often the source of the project management difficulties, but problems were aggravated in this case by the IS department’s inability to identify the real needs of stakeholders and to negotiate an appropriate win-win package. Although the IS group was neglectful, it is not to blame-the problem lies with the process.
STEPS TO IMPROVE THE PROCESS

To improve these processes, Theory-W principles of software management can be used. The following three steps, which are adapted from “Theory-W Software Project Management,” can be used to implement Theory-W software management:

1. Establishing a set of win-win preconditions by performing the following:
   — Understanding what it is that people want to win.
   — Establishing an explicit set of win-win objectives based on reasonable expectations that match participants’ objectives to their win conditions.
   — Providing an environment that supports win-win negotiations.

2. Structuring a win-win development process by accomplishing the following:
   — Establishing a realistic plan that highlights potential win-lose and lose-lose risk items.
   — Involving all affected parties.
   — Resolving win-lose situations.

3. Structuring a win-win software product that matches the following:
   — The users’ and maintainers’ win conditions.
   — Management’s and supplier’s financial and scheduling win conditions.

APPLICATION OF THEORY W

There are several actions the retailer’s IS department could have taken to increase the project’s probability of success. It could have trained distribution personnel on the key role they have in properly identifying and articulating their needs. Following training, IS staff could have worked with these users to draft a more thorough RFP. After receiving RFP responses, the IS department could have involved senior management in identifying limits on the resources and schedules. It could have foreseen the difficulties the contractor would have if significant program modifications were needed.

By identifying constituent win conditions, the IS department would then have been in a position to negotiate a fair contract that would have explicitly taken into account all the win conditions. Having done these critical up-front tasks, the IS department would then have been in a position to structure both a win-win development process and a software product. Unfortunately, the IS department never set win-win preconditions.

In the absence of an explicit philosophy to make everyone a winner and an explicit process for accomplishing this, the IS department lacked the
necessary support to identify stakeholder’s needs and negotiate a reason-
able set of win-win objectives. Thus, it was a matter of time until incompat-
ible and unobtainable win conditions destroyed the project.

INTEGRATING THEORY W INTO THE DEVELOPMENT LIFE-CYCLE

Theory-W easily integrates into the classical waterfall life-cycle. During
requirements definition, for example, it can be used to ensure that manage-
ment, users, and developers have their individual needs met in a way that
supports the timely and cost-effective development of the system. During
design phases, developers and maintenance personnel can work together
to ensure that the system is able to be efficiently maintained. And through-
out all phase, Theory-W is a natural mechanism for ensuring that all stake-
holder are able to effectively articulate their individual win conditions.

Boehm and his colleague have recently extended the integration of The-
ory-W into the system life-cycle by integrating Theory-W into Boehm’s Spi-
ral Model of system development. As originally conceived in the late 1980s,
the Spiral Model was a four-step iterative model for system development
that provided a better mirror of actual real-world development practices
than the classical waterfall model, particularly in situations where it is dif-
ficult to pin down systems specifications in advance. As such, the model
shares similarities to other evolutionary models of system development,
like, for example, the one developed by Tom Gilb.

In extending the Spiral Model to explicitly incorporate Theory-W, Boehm
and his colleagues have produced the following iterative seven-step model
(Exhibit 1):

1. Identify next-level stakeholders.
2. Identify stakeholders’ win conditions.
3. Reconcile stakeholders’ win conditions. Establish next level object-
tives, constraints, and alternatives.
5. Define next-level product and process-including systems-wide parti-
tions.
6. Validate product and process definitions.

As a part of the integration of Theory-W into the Spiral Model, Boehm
and his colleagues have prototyped an interactive tool in which constitu-
ents can either win their own conditions and all stakeholders can then
simultaneously analyze and negotiate a combined set of win conditions.
They have used the tool to support the successful acquisition of a commer-
cial off-the-shelf system for the Air Force.
CONCLUSION

Theory-W has been shown to be successful—both in terms of the paradigm it offers and in terms of its ability to help the managers explicate and simultaneously manage the win conditions of all constituents and stakeholders. Theory-W has also proven itself to be of value for organizations embarking on a systematic program to improve performance—whether of more effective system deployment, improved productivity, better customer service, total quality management, or more effective processes. Indeed, to be successful performance improvement programs require the full and complete support of all stakeholders. Theory-W offers a robust theory, an effective process, and a practice model, along with a prototype tool, for getting and keeping this needed support.