Prerequisites for Success: Why Process Improvement Programs Fail

David Cottengim
Defense Finance and Accounting Service

Why do system development programs so often fail to meet their objectives? Why do efforts to implement software process improvement methodologies fail to yield promised results? Regardless of the improvement methodology chosen, there are fundamental prerequisites to success that must be present. Absent these prerequisites, any attempt to implement a structured approach to quality improvement will fail. It is not enough to infuse the tools of improvement into an organization. The fundamental nature of an organization must support the core business changes required to successfully implement an improvement program.

System development programs continue to fail at an alarming rate. Failure is defined as: a) coming in late, b) going over budget, or c) not delivering what was required [1]. Failure often comes only after millions of dollars in scarce resources have been invested in the doomed venture. In spite of all the existing research and lessons learned, system development programs remain recalcitrant. There is clearly a need to improve the quality of system development efforts [2].

The failure to accurately and completely identify the problem to be solved (system requirements) is a root cause of system development failures [3]. The following definitions will provide a base of understanding for this article:

- What is a system requirement? System requirements encompass a broad spectrum of capabilities and attributes that a business system must possess. To properly address the risks associated with requirement gathering, a sufficiently inclusive definition must be adopted. For the purposes of this discussion we use the following definition of system requirement:

  > Any function, capability, characteristic, constraint, or purpose the software system in question must directly or indirectly address or satisfy for any stakeholder [4].

- What is a system requirement risk? There are many formal definitions of risk. I choose to apply the following definition provided by the Project Management Institute:

  > A risk is an uncertain condition that, if it occurs, has a positive or negative effect on a project objective [5].

Where Do Requirements Originate?

System requirements should emanate from a business need [6]. Numerous methodologies exist for the sole purpose of requirement elicitation, gathering, and documentation. The impediments to successful requirement elicitation are numerous [7]. Multiple business variables converge on any effort to gather the requirement for a perceived business need.

> *"The seeds of system failure are often sown at this point in the requirements elicitation process. Many organizations lack the ability to consolidate and reconcile multiple stakeholder viewpoints ..."*

Business problems must be evaluated in the context of strategic planning beyond the system solution being developed for any individual business need. Shortsighted solutions to immediate business needs might cause considerable long-term harm to the organization [8].

Stakeholder Viewpoints

Most software systems in today's business environment have stakeholders with divergent and conflicting points of view about the nature of the business problem, let alone how it should be solved. These varied points of view manifest in the requirement elicitation process and must be considered before a final system solution to a problem can be defined [9]. Figure 1 describes a basic requirement elicitation process.

Senior management should approach a business need from a strategic point of view. How does this problem fit into the organization's larger mission? Can a link be drawn between solving a particular business need and a larger organizational performance goal?

Middle management might also have a strategic slant to their point of view, but will generally also bring a near-term tactical point of view to what needs to be done about a business problem. Is this problem preventing the accomplishment of mission-critical functions? Will solving the perceived problem provide any benefits to their level of the organization? How will any proposed system solution impact their way of doing business?

Task level action officers bring yet another point of view to the requirement elicitation process. At the task execution level of an organization, there are very real concerns about how a new software application will impact day-to-day operations. Software systems addressing larger organizational objectives will often place additional workload on those at the task execution level. Job security, job satisfaction, advancement, need for training, skill set requirement, and retention of institutional knowledge are all areas of concern at the task execution level of the organization.

Defining the Final System Requirement

Documenting the different stakeholder points of view is not the end of the requirement definition process. The requirements generated by all the different points of view must be synthesized into a

single system requirement. The skills required to consolidate divergent requirements and to maneuver stakeholders to agreement are elusive.

The seeds of system failure are often sown at this point in the requirements elicitation process. Many organizations lack the ability to consolidate and reconcile multiple stakeholder viewpoints or to resolve conflicting requirements.

The pivotal transition for organizations wishing to reduce system requirement risks is the implementation of an underlying culture that facilitates the migration from current business practices and points of view to a new world view and mode of thought in support of the new system.

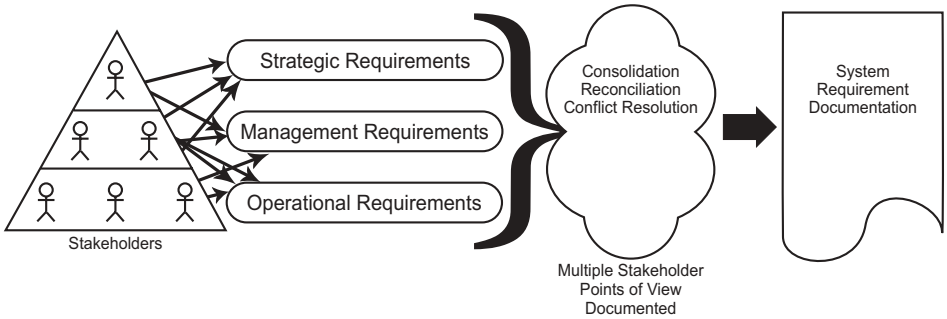


Figure 1: *Requirements Elicitation Process*

Requirements Elicitation Obstacles

Figure 2 presents the typical environmental obstacles all stakeholders must face as they try to define business requirements.

All stakeholders are impacted by four primary sources of system requirement risks. Each stakeholder group will build their own personalized amalgamation of these factors as they present what they perceive to be the business need and the correct system solution to address that need.

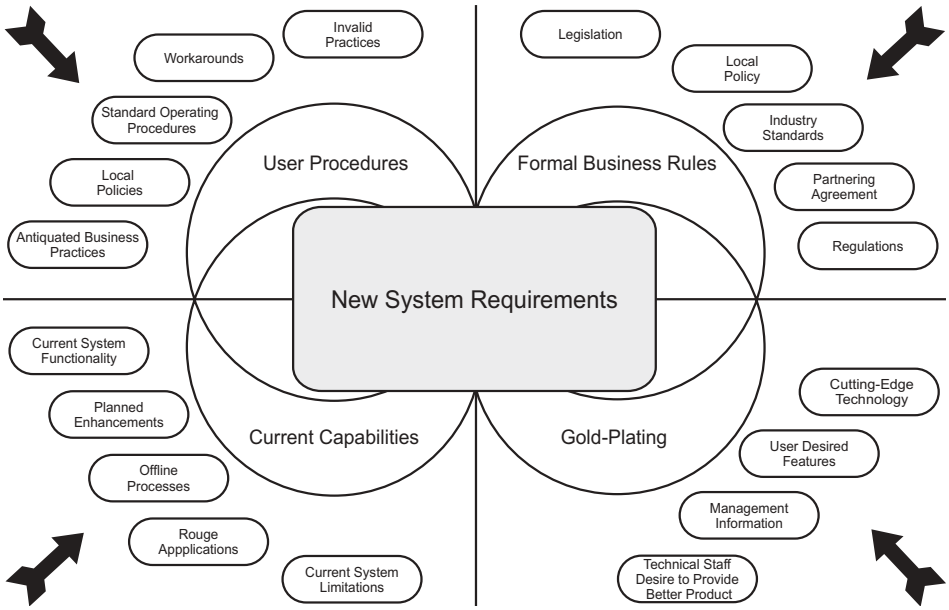
User Procedures

- Invalid Practices: Invalid day-to-day practices are often discovered during requirement elicitation exercises. Over time the end users will follow the path of least resistance to balance demands on their time and resources. This will often result in long-standing practices that are in direct violation of formal policy, guidelines, regulations, or legislation.
- Workarounds: Limitations of the current system solution almost always require the end users to develop workaround procedures to accomplish mission critical activities. These workarounds may not be documented or even formally acknowledged by the management of the business area. Workarounds may go on for years after the original need has vaporized.
- Standard Operating Procedures (SOPs): SOPs are often documented for a business activity. SOPs vary in degree of relevance and accuracy for current or future business needs. Depending on the degree of correlation between the SOPs, the real business need, and how things are really accomplished, use of the SOPs might not be a valid approach to defining system requirements for a new system.
- Local Policies: Tradition and local preference influence business practices. Local policies regarding sharing authority, delegation of duties, customized methods, and adherence to formal business rules will vary.
- Antiquated Business Practices: As technology and business practices evolve, it is not uncommon for personnel to hold on to old and familiar ways of doing business. Filtering outdated business practices out of the requirements for a new system solution can often be in conflict with the desire to make the system user friendly.

Current Capabilities

- Current System Functionality: The easiest trap to fall into is allowing current system capabilities to dictate future system capabilities [10]. The connection between current system capabilities and real business requirement is dubious at best. The program manager must maintain a focus on business needs.
- Planned Enhancements: Many current business applications have been in a production environment for decades. User requested enhancements are almost certainly planned and backlogged. Any effort to develop a new system will be saddled with a list of enhancements the old system was going to satisfy "any day now."
- Offline Processes: Completely mapping current business practices is the key to finding offline processes. End users usually developed offline processes to compensate for legacy system deficiencies. If an offline process cannot be linked to a current business need, it should not be a source of system requirements for a new system.
- Rogue Applications: Advances in desktop application capabilities have given managers and end users in all areas of operations the ability to build their own unique applications to meet their business needs [11]. The new system solution might need to interface with these rogue systems or absorb their functionality.
- Current System Limitations: Business practices are constrained by the limitations of legacy systems. Relying on

Figure 2: *Environmental Obstacles to Requirements Elicitation*



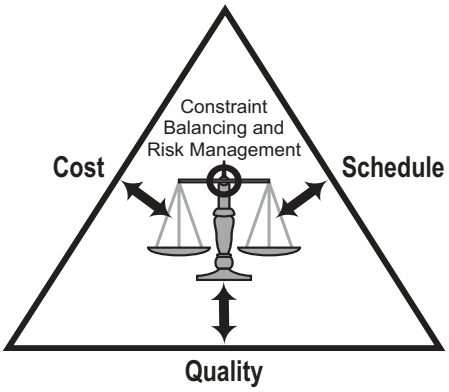


Figure 3: *Primary Constraints*

current business practices as the sole source of requirements is not the best technique for defining current business needs. Current business practices might have been overly tailored to accommodate material limitations in the current legacy systems.

Formal Business Rules

- Legislation: Deliberative bodies at all levels of government have attempted to address the performance of system acquisition programs. Numerous legislative requirements influence system solutions for government entities.
- Local Policy: Lack of standardized implementation methods for formal business rules and the resulting variation in business requirements will overwhelm attempts to develop a single system solution to a common business need. Peer-level organizations may lobby to have their way of doing business included as a mandatory feature of a new system solution to a business need.
- Industry Standards: Industry groups define standards for conducting business. These standards often influence requirements for system solutions to business needs. The integration and interdependence of business systems across organizational boundaries generate the need to stay current with applicable industry standards to assess their impact on planned system solutions.
- Partnering Agreements: Beyond the requirements of industry standards, there are usually requirements between organizations regarding how industry standards will be implemented or how other common business practices will be integrated. Standards only provide templates and guidelines. Additional work beyond what the standard provides is normally required before a working relationship between two organizations can be finalized.
- Regulations: Business rules for government-related activities often manifest in the form of published regulations. Regulations provide further guidance on implementation of policy or legislative requirements. Inconsistent interpretation of regulations is another source of variation in business processes for identical business needs.

Gold-Plating

Adding features or functionality to a system that are not required to satisfy the minimum operational requirements is referred to as "gold-plating" [12].

- Cutting-Edge Technology: Rapid advances in technological capabilities can entice businesses to introduce unnecessary capabilities into system requirement documentation. The desire to work with the latest and greatest technology is compelling.
- User Desired Features: Technologically savvy users will attempt to dictate the technical solution to their business need. Organizational level strategic goals and objectives must supercede user-level requests for specific technical solutions to business needs.
- Management Information: The quality and accuracy of management information is a top priority for system solutions to business needs. The presentation of the information can be a source of gold-plating. Users prefer information be presented in familiar formats. This might unnecessarily increase development costs. This is especially true when an activity is attempting to implement a commercial off-the-shelf package without making significant modifications to the core product.
- Technical Staff Desire to Provide Better Product: System engineers are often ingenious and will seek new and better technical solutions for business needs. Organizations must balance the three primary constraints of any project effort as depicted in Figure 3. As schedule and cost constraints become fixed, it is often necessary to compromise on quality and accept "good enough" system solutions [13].

Overcoming the Obstacles

What can an organization do to reduce system requirement risks and overcome these impediments to successful system development?

Contemporary Wisdom

Contemporary wisdom is to implement some version of the various process improvement programs as a method for reducing system requirement risks. Several software process improvement methodologies have risen to the forefront. These methodologies include the following:

1. The Capability Maturity Model® <www.sei.cmu.edu>.
2. International standards such as ISO/IEC 15504 (SPICE) <www.sei.cmu.edu/iso-15504> and ISO 9001 <www.iso.org/isoen/iso9000-14000/tour/ magical.html>.
3. Joint Application Development (JAD) [14].
4. Rapid Application Development (RAD) [15].
5. Quality Function Deployment (QFD) <www.qfdi.org>.
6. Six-Sigma <www.6-sigma.com>.

While each of these methodologies and techniques has shown positive results in specific implementations, none of them can claim to be the silver bullet of software process improvement [16].

Implementing these methodologies alone will not burrow down far enough into the core competencies of the organization. While these methodologies do present better techniques for reconciling and consolidating requirements and resolving conflicting requirements, they all share the same fundamental prerequisites for success.

Organizations should therefore first focus on the foundational prerequisites for success. Only after the prerequisite conditions have been secured can a methodology or technique significantly reduce system requirement risks.

Building the Foundation for Success

The environment surrounding system development will not become less complex. Organizations must adapt to the complexity of their environment. Implementing new requirements gathering methodologies without the attendant examination of the organization's underlying characteristics is too often the approach taken to address environmental complexity. The key to successfully adapting to increased environmental complexity is to focus management's attention on the characteristics of how the organization engages complexity.

Failure to create the prerequisite organizational character to foster the success of process improvement or risk management programs will cause the implementation of any methodology to be superficial and doom it to failure. The specifics of the process adopted are not nearly as critical as the philosophical change required to tran-

sition to the new paradigm [17].

Therefore management's focus should shift away from a particular methodology and toward the creation of an environment that meets the prerequisites for success under any of the possible methodologies. The success of any action to manage system requirement risks will depend on the dominating presence of several key prerequisites, as shown in Figure 4 [18].

Prerequisites for Success

1. Leadership: The cornerstone to any successful process improvement or risk management plan is leadership focused on clearly defined goals and objectives. This is very difficult to find even in small organizations and woefully lacking in large institutions. Leadership ambiguity will confuse and frustrate personnel and misdirect resources. Organizational leaders must remove obstacles and move the organization toward the selected goals and objectives [19]. Leadership vacuum will guarantee failure of any effort to materially reduce system requirement risks.
2. Commitment: Talk is cheap. Rhetoric is damaging. Management must be prepared to show firm commitment to risk management policies and process improvement methods in their own actions and in the actions they require from personnel. Commitment is needed from every member of the organization. The level of commitment among staff will vary but management must be committed to building teams of individuals that are fundamentally behind the improvement program. Team members must be supportive of the efforts to implement new initiatives [20]. Retention of key personnel who refuse to transition to the New World view required to support a system or methodology demonstrates a lack of management's commitment and will undermine risk reduction and process improvement efforts.
3. Honesty: Many organizations will not face the truth of their environment. Failures are often spun into successes. Yardsticks of success are shortened to declare victory when any objective evaluation would return a failure verdict. Organizations that cannot be honest about their shortcomings and failures will be powerless to take action for improvement. Stakeholders need to evaluate the results of their performance against honest objectives to determine their cause and interrelationship [21]. Pretending projects have gone well will breed cynicism. Rewarding failure de-motivates personnel. An environment of dishonesty cannot take the action required to foster improvement.
4. Training (a.k.a. education): Once a direction is chosen, the organization must be mobilized to support it. Training in the chosen methodology will show personnel the behavior that will be rewarded. Adopting behavior that supports the organization's risk management objectives is the critical path to successful process improvement. Education does not teach behavior. Organizations must take care to differentiate between education on a subject and training in a specific behavior [22]. The organization must then be honest when evaluating performance relative to the trained behavior.
5. Standardization: An organization cannot afford multiple methods for achieving the same outcome. This will be the stumbling block for many system implementations. Non-standard business methods will cripple system solutions to business needs. Stakeholders must insist the organization migrate toward standard processes and overcome resistance against transitioning to fewer optimized business processes [23]. Permitting multiple processes for accomplishing the same task will add complexity to system requirements that cannot be overcome by improved processes or risk management programs.
6. Professionalism: Requirement elicitation, cost estimating, system engineering, system testing, and project management have become formal professions. Each of these professions has a body of knowledge for practitioners to acquire and master. Professional certifications exist to help measure practitioner skill level. Adequately trained and skilled personnel should be hired and placed in these professional positions. A team of professionals committed to following proven methods in their work will produce consistent and predictable results that will demonstrate sustained improvement [24]. An honest assessment of skills will usually reveal significant skill-set deficiencies in personnel holding key leadership positions. Poor decisions by unskilled personnel will produce cascading impacts throughout a program. Hard work after the damage is done can rarely recover the opportunity costs of poor decisions.

Implications

If you choose to agree that the prerequisites for success must be present before any software process improvement or risk management program can succeed, then the implications of that belief are quite severe.

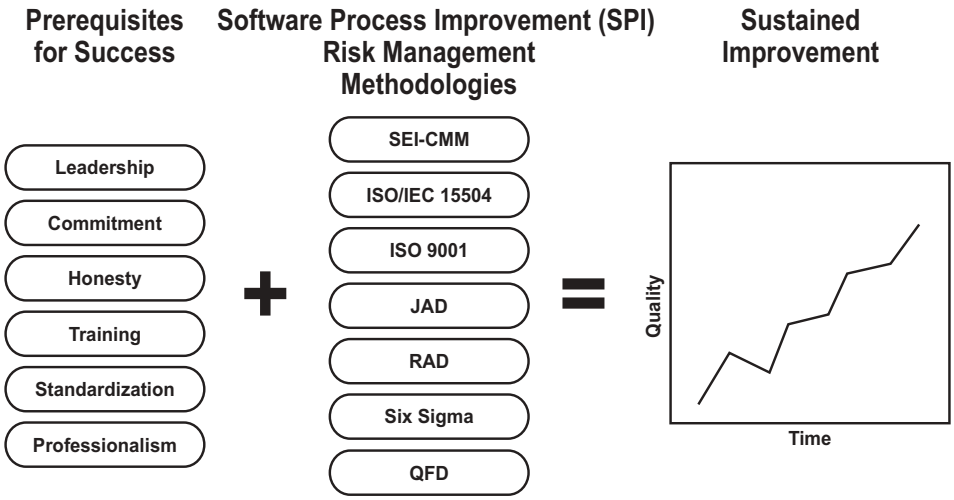
The mentality that suggests the definition of a good program manager is someone who can achieve success in an environment devoid of the prerequisites for success must be abandoned. That modality of leadership simply cannot overcome the obstacles confronting improvement programs.

If you agree that the prerequisites I have identified (see Figure 4) are required for success, then the conclusion follows naturally that some organizations are not ready to implement software improvement programs and should not be developing complex software systems.

Software improvement or risk reduction programs absent the prerequisites for success will continue to experience dismal performance in their attempts to develop software intensive systems [25]. This leaves you with the difficult task of introspection.

Before you invest the stakeholder

Figure 4: *Prerequisites for Success and SPI Risk Management Methodologies*



CROSSTALK
The Journal of Defense Software Engineering

Get Your Free Subscription

Fill out and send us this form.

OO-ALC/TISE
7278 FOURTH STREET
HILL AFB, UT 84056-5205
FAX: (801) 777-8069 DSN: 777-8069
PHONE: (801) 775-5555 DSN: 775-5555

Or request online at www.stsc.hill.af.mil

NAME:_____

RANK/GRADE:_____

POSITION/TITLE:_____

ORGANIZATION:_____

ADDRESS:_____

_____

BASE/CITY:_____

STATE:_____ZIP:_____

PHONE:(____)_____

FAX:(____)_____

E-MAIL:_____@_____

CHECK BOX(ES) TO REQUEST BACK ISSUES:

FEB2000 ☐ RISK MANAGEMENT

MAY2000 ☐ THE F-22

JUN2000 ☐ PSP & TSP

APR2001 ☐ WEB-BASED APPS

JUL2001 ☐ TESTING & CM

AUG2001 ☐ SW AROUND THE WORLD

SEP2001 ☐ AVIONICS MODERNIZATION

DEC2001 ☐ SW LEGACY SYSTEMS

JAN2002 ☐ TOP 5 PROJECTS

MAR2002 ☐ SOFTWARE BY NUMBERS

resources entrusted to you in any type of improvement program, a difficult decision awaits you. If you know your organization does not foster the environment required for success, then you have a fiduciary duty to not waste the resources you have been provided.

You must first take action to transform your organizational environment to one that provides the prerequisites for success, or you will simply be adding to the list of failed improvement efforts and cancelled system programs.◆

References

1. O'Connell, Fergus. How To Run Successful High-Tech Project-Based Organizations. Boston: Artech House, 1999. xvii.
2. Humphrey, Watts S. Managing the Software Process. New York: Addison-Wesley, 1989. 13.
3. Sommerville, Ian, and Pete Sawyer. Requirements Engineering: A Good Practice Guide. New York: Wiley, 1998. 64.
4. Young, Ralph R. Effective Requirements Practices. Boston: Addison-Wesley, 2001. 9.
5. Project Management Institute. A Guide to the Project Management Body of Knowledge. PMI Publishing, 2000. 127.
6. Wiley, Bill. Essential System Requirements: A Practical Guide to Event-Driven Methods. Reading, Mass.: Addison-Wesley, 2000. 15.
7. Gilb, Tom. Software Inspection. Addison-Wesley, 1993. 253.
8. Weinberg, Gerald M. Quality Software Management, Vol. 1, Systems Thinking. New York: Dorset House, 1992. 155.
9. Metzger, Philip, and John Boddie. Managing A Programming Project. New Jersey: Prentice Hall, 1996. 19.
10. Kotonya, Gerald, and Ian Sommerville. Requirements Engineering Processes and Techniques. New York, 1998. 171.
11. Jones, Capers. Assessment and Control of Software Risks. N.J.: Yourdon, 1994. 37.
12. Wiegers, Karl E. Software Requirements. Microsoft Press, 1999. 13.
13. Yourdon, Edward. Death March. N.J.: Prentice Hall, 1997. 147.
14. Wood, Jane, and Denise Silver. Joint Application Development. New York: John Wiley & Sons Inc., 1995.
15. McConnel, Steve. Rapid Development. Microsoft Press, 1996. 2.
16. Kemerer, Chris F. Software Project Management Readings and Cases. Boston: McGraw-Hill, 1997. 591, 600.
17. Horch, John W. Practical Guide to Software Quality Management. Boston: Artech House, 1996. 192.
18. Schulmeyer, Gordon G., ed., and James I. McManus. Handbook of Software Quality Assurance, 3rd ed. N.J.: Prentice Hall, 1999. 61.
19. Smith, Perry M. Rules & Tools for Leaders: How to Run an Organization Successfully. New York: Avery, 1998. 33.
20. IEE Computer Society. Software Engineering Project Management. IEE Computer Society, 1997. 380.
21. Aurelius, Marcus. The Meditations Of Marcus Aurelius, Book 12. Trans. A.S.L. Farquharson. Knopf, 1944. Chapters 10 and 29.
22. Beer, Michael. The Critical Path to Corporate Renewal. Harvard Business School, 1990.
23. Harry, Mikel, and Richard Schroeder. Six Sigma. New York: Currency, 2000. 134.
24. Humphrey, Watts S. A Discipline For Software Engineering. Mass.: Addison-Wesley, 1995. 474.
25. Hall, Elaine M. Managing Risk, Methods For Software Systems Development. Addison-Wesley, 1998. 152.

About the Author

David Cottengim is a financial analyst at the Defense Finance and Accounting Service, Indianapolis. He has more than 10 years experience in system development for Department of Defense activities. He is certified as a Project Management Professional, Certified Software Test Engineer, and Certified Government Financial Manager. He completed his undergraduate education in finance and economics and his graduate education in finance and management information systems at the Indiana University Kelly School of Business.

Defense Finance and Accounting Service - Indianapolis
Building #1, Column 230F
8899 East 56th Street
Indianapolis, IN 46249
Phone: (317) 510-3121
Fax: (317) 510-3174
E-mail: david.cottengim@dfas.mil