Minnesota Intellectual Property Review

Volume 3 | Issue 2 Article 3

2002

Issue Management: A Safety Net for Custom Software Development Projects

Greg L. Smith

Issue Management: A Safety Net for Custom Software Development Projects

*Greg L. Smith**

INTRODUCTION

"Most large-scale systems projects fail."[1] Lawyers, academics, and pundits have failed to make systems development projects successful despite scores of articles, books, and treatises written on the topic of project failure. In the Preface to *Computer Law Handbook*, author David F. Simon stated that by 1989 the pace of development in computer law had "finally slowed."[2] The implication from the high failure rates and Mr. Simon's statement is that computer law has stabilized around and accepted failure.

One reason that systems development projects fail so frequently is that system requirements are constantly changing in response to new and clarified business needs and evolving technology.[3] Traditional development contracts attempt to create a static set of system requirements that can be modified

---

* Greg L. Smith is a second-year law student at the University of Minnesota. Prior to law school, Mr. Smith spent five years working as a consultant with American Management Systems (AMS). Worldwide, AMS is one of the 20 largest international business and information technology consulting firms. *See* http://www.ams.com/AboutAMS/. At AMS Mr. Smith worked extensively with project managers in a effort to help them refine project management tools and processes, including requirements, issues, risks, test cases, schedules, status reports, and incidents/defects. Mr. Smith has an undergraduate degree in Business Management with a special emphasis in Information Systems.

1. Edward M. Roche et al., *The Technical Framework of Information Technology Litigation*, *in* INFORMATION TECHNOLOGY LITIGATION: REPRESENTING YOUR CLIENT IN SOFTWARE PERFORMANCE & SYSTEM FAILURE DISPUTES 7, 11 (Practising Law Inst. ed., 2001). Failure rates for large projects are as high as sixty-five percent. *See id.* at 19.

2. DAVID F. SIMON, COMPUTER LAW HANDBOOK: SOFTWARE PROTECTION, CONTRACTS, LITIGATION, FORMS, ix (1990).

3. *See 13 Ways to Avoid Risks of Buying a Large Accounting System,* MANAGING ACCT. SYS. & TECH., Dec. 1999, at 4, 4; SIMON, *supra* note 2, at 170 (recognizing that change is "necessitated by changing or newly recognized business needs and requirements").

only through a rigid change control process. This traditional approach does not reflect the realities of software development. Software development contracts need to be dynamic, *living* contracts. They should allow the obligations of the parties to evolve in the same way that a system evolves throughout the life of the development project.

Issue management is one process that facilitates the evolution of the system. Issue management is a formal process for resolving issues, which are defined as problems, obstacles, changes, and questions that are disruptive to the progress of the project.[4] Issue management has been labeled "the essence of system management."[5] The purpose of this Note is to encourage the inclusion of an issue management provision in software development contracts. Such a provision has the potential to transform the software development contract into a dynamic, living document that can evolve the legal obligations of the parties in parallel with the evolution of the system.

This Note will show that the traditional legal approaches to software development are inadequate because they largely ignore issue management. Part I of this Note will give an overview of the failure of software development projects and will describe the methods most lawyers currently use to approach project and issue management within software development projects and contracts. Part II will explain why the current approach to software development projects and contracts is inadequate and will propose a means for lawyers and contracts to facilitate issue resolution. The Note concludes that legal processes and contracts that facilitate issue resolution may lead to a decrease in the failure rates of software development projects.

---

4. *See* Robert L. Glass, *Issue Management*, DATABASE FOR ADVANCES IN INFO. SYS., Fall 1998, at 16, 17 (concluding that issues threaten to disrupt and "derail" the project); K.C. Burgess Yakemovic & E. Jeffrey Conklin, *Report on a Development Project Use of an Issue-Based Information System, in* PROCEEDINGS OF THE CONFERENCE ON COMPUTER-SUPPORTED COOPERATIVE WORK, 105, 106 (1990) (defining issues as questions or problems).

5. Glass, *supra* note 55, at 2.

I. THE CURRENT STATE OF SOFTWARE DEVELOPMENT AND ISSUE MANAGEMENT

A. PROJECT FAILURE IS THE RULE RATHER THAN THE EXCEPTION

> It remains a sad statistic that too many software development projects end in failure. Fully [twenty-five] percent of all software projects are cancelled outright. As many as [eighty] percent of all software projects run over their budgets, with the "average" software project exceeding its budget by [fifty] percent. It is estimated that three-fourths of all large systems are "operational failures" because they either do not function as specified or are simply not used.[6]

The cost of failure for an information systems project is frequently far greater than the obvious monetary and resource expenditures.[7] Today, most organizations are heavily, if not completely, dependant on information technology. In fact, one author has observed, "the company is the system."[8] Many businesses rely on their information systems to develop and maintain competitive advantages.[9] Because organizations are so dependant on information systems, "getting it right" may be the truest measure of success.[10] Failed software projects

---

6. Roy Schmidt et al., *Identifying Software Project Risks: An International Delphi Study*, J. MGMT. INFO. SYS., Spring 2001, at 5, 6 (endnotes omitted). The success rate of a project declines rapidly as the project becomes larger. *See, e.g.,* Roche, *supra* note 1, at 19; *See also* Bruce A. Levy, *System Acquisition-Protecting the User*, *in* 19TH ANN. INST. ON COMPUTER LAW 1099, 1101 (Practising Law Inst. ed., 1999) (indicating that only seven percent of studied projects costing between five and ten million dollars were successful and only eighteen percent between one and two million dollars were successful).

7. *See* S. Revelle Gwyn & Alan T. Rogers, *Negotiating and Litigating Computer Law Contracts: Selected Issues,* ALA. LAW., Nov. 1992, at 404, 404 ("Failure of a computer system can damage a business.").

8. *See, e.g.,* Roche, *supra* note 1, at 18 (internal quotation marks omitted).

9. *See* Peter Brown, *Litigating Failed Software Actions*, OCT. 1993 A.B.A. SEC. OF LITIG. 2.

10. *See generally* Robert L. Glass*, Evolving a New Theory of Project Success (Industry Trend or Event)*, COMM. OF THE ACM, Nov. 1999, at 17, 17-19 (previewing a study by Kurt R. Lindberg, *Software Developer Perceptions About Software Project Failure: A Case Study*, J. SYS. & SOFTWARE, Dec. 30, 1999, at 177, finding that what would typically be viewed as a project failure because of budget and schedule over-runs was viewed as a success by developers because the project delivered a quality product that worked as

undermine the success of the entire organization.

A few of the reasons information systems projects fail are:

- Incomplete and changing requirements and specifications[11]
- Poor communication[12]
- Inconsistent decision making[13]
- Poor project planning – including inadequate risk management,[14] budget overruns, and schedule overruns[15]
- Failed business justification for the system[16]
- Lack of top management involvement and support[17]
- Lack of end-user involvement and support[18]
- Use of new and unproven technology[19]
- Inability of vendors to meet commitments[20]

B. A LAWYER'S BIPOLAR ROLE IN THE SOFTWARE DEVELOPMENT PROJECT

Given the high rates of failure for software development projects, one might expect legal counsel to be commonly and thoroughly involved in such projects. One would expect the level of involvement to increase as the cost of the system increases.[21] However, in reality "too little attention is devoted

---

expected).

11. *See 13 Ways to Avoid Risks of Buying a Large Accounting System, supra* note 3, at 4.

12. *See* Michael G. Addario & Lloyd S. Weber, *Why Good Projects Go Bad, Preventing Project Management Meltdown*, MARRIOTT ALUMNI MAG., Fall 2001, at 13, 14.

13. *See id.*

14. Risk management is "[t]he systematic application of management policies, procedures, and practices to the tasks of identifying, analyzing, assessing, treating, and monitoring risk." Ken Doughty & Franke Grieco, *Managing The Risks of Outsourcing Systems Development*, *in* HANDBOOK OF SYS. DEV. 35, 37 (Paul C. Tennirello ed., 1999).

15. *See* Brenda Wittaker, *What Went Wrong? Unsuccessful Information Technology Projects*, 7/1 INFO. MGMT. & COMPUTER SEC. 23, 23 (1999).

16. *See id.*

17. *Id.*

18. *See* David J. Gardner, *How to Avoid IT Project Failures*, CONSULTING TO MGMT., May 2000, at 21, 22; *13 Ways to Avoid Risks of Buying a Large Accounting System*, *supra* note 3, at 4.

19. *See* Wittaker, *supra* note 15, at 23.

20. *See id.*

21. Software development agreements are commonly in the million-dollar

to the legal relationship between [buyer] and vendor . . . This is surprising for transactions of such magnitude."[22]

Lawyers are routinely involved, at least minimally, during the contract negotiation stage of a project. Frequently attorneys are asked to merely rubber stamp a contract that has been negotiated between project managers and the vendor's marketing staff.[23]

After the contract has been signed, lawyers are not typically involved with the development project again until the project is well on the road to failure.[24] Lawyers are not typically involved in the interim phases of a project because they are viewed "as risk identifiers, nit pickers, and deal breakers—not as helpers."[25] A lawyer's tendency to consider the worst case scenario is not consistent with a teamwork oriented, can-do, and success-driven project.[26]

---

range and not uncommonly in the hundreds of millions of dollars range.

22. *13 Ways to Avoid Risks of Buying a Large Accounting System*, *supra* note 3, at 5 (quoting Rauer L. Meyer, Partner in the Technology Department of Thelen, Reid & Pries).

23. *See* Stephen J. Davidson, *Avoiding Pitfalls and Allocating Risk in Major Software Development and Acquisition Contracts*, COMPUTER LAW., May 1997, at 12, 12. "[W]hile your client may be willing to spend a year and a hundred thousand dollars to identify the technical solution and another million dollars to acquire it, he or she often will be loathe to spend a thousand dollars for legal review of the contract." *Id.* at 13.

24. *See* Brown, *supra* note 9, at 5; *but cf. 13 Ways to Avoid Risks of Buying a Large Accounting System*, *supra* note 3, at 6 (advocating legal counsel involvement beyond the contract stage of a project and before problems arise); Mark L. Gordon & Francoise Gilbert, *Contracting for Systems Integration Transactions*, COMPUTER LAW., Dec. 1991, at 13, 19 (suggesting that legal counsel should be involved in reviewing design specifications to ensure that design requirements are clear and sufficiently detailed).

25. Davidson, *supra* note 23, at 12. Project escalation theories may shed some light on why development projects shy away from lawyers who identify risks and nit pick rather than "help." Escalation is basically a "continued commitment to a failing course of action." Mark Keil, *Pulling the Plug: Software Project Management and the Problem of Project Escalation*, MIS QUARTERLY, Dec. 1995, at 421, 422. Escalation is more technically defined as a "continued commitment in the face of negative information about prior resource allocations coupled with 'uncertainty surrounding the likelihood of goal attainment.'" *Id.* at 422 (citing J. Brockner et al., *The Escalation of Commitment to a Failing Course of Action: Toward Theoretical Progress*, ACAD. OF MGMT. R., Jan. 1992, at 39). While many theories have been proposed to explain why projects escalate, a common thread is that managers for various reasons choose to believe that failing projects can turn around and ultimately be successful. *See generally id.* at 422-23 (explaining various project escalation theories). Optimistic managers don't want high-priced lawyers telling them that their half-full glass is really almost empty.

26. *See* THE LAW AND BUSINESS OF COMPUTER SOFTWARE § 14.02[b] (D.C.

Progress in the area of software-development law should be measured by the success rates of software development projects. A lawyer's roll is to assist the client in accomplishing business objectives.[27] The vendor's duty is to supply and the buyer's goal is to obtain a reasonably complete and working computer system.[28] Therefore, the lawyer's purpose is to help the client either supply or obtain a reasonably complete and working computer system. While important secondarily, the lawyer's focus is not to develop a contract that is well-poised for future litigation or that can be used as a weapon against the other side.[29] The software development contract must facilitate the success of the project. A lawyer's role in a failing software project is to bring the project back on track as much as to prepare for potential litigation.[30]

C. OVERVIEW OF SOFTWARE DEVELOPMENT PROJECTS

Software development projects are comprised of multiple phases and complex interactions. While numerous approaches to software project management exist, the process depicted below in

Figure 1 is both typical and traditional.[31] The phases of a software project typically include planning, requirements analysis, systems design, testing, implementation, and support and maintenance. The development contract is usually

---

Toedt III ed., 2001).

27. *See* Cambridge, *Contracting in the Business Environment*, *in* NEGOTIATING COMPUTER CONTRACTS 92, 92 (1985) (indicating that the client's business is the lawyer's purpose and that the contract should help in accomplishing business objectives and is not an end to itself).

28. *See* SIMON, *supra* note 2, at § 3.01 ("The user's primary objective is to obtain a reasonably complete and error-free computer system, not to be well-poised for future litigation.").

29. *See id.;* Cambridge, *supra* note 27, at 92.

30. *See* Mark L. Gordon & Steven V. Starr, *Software Development Contracts and Consulting Agreements: A Structure for Enforceability and Practicality*, *in* NEGOTIATING COMPUTER CONTRACTS 142, 173 (1985).

31. Figure 1 is a very simplified and incomplete view of a project, and is intended only to provide context for this article. *See generally* INFO. RESOURCES MGMT., U.S. DEP'T OF JUSTICE, THE DEPARTMENT OF JUSTICE SYSTEMS DEVELOPMENT LIFE-CYCLE GUIDANCE DOCUMENT Sec. § 1.2 (Mar. 2000)(summarizing the system development life cycle and explaining each phase), *available* at http://www.usdoj.gov/jmd/irm/lifecycle/table.htm (last visited February 13, 2002). While Figure 1 depicts only one of many different development approaches, analysis of these other approaches is beyond the scope of and unnecessary for the purposes of this article.

negotiated and agreed to during the planning and requirements analysis phases. Each of the system development life-cycle[32] phases is supported by a common project management component. A few examples of project management components pertinent to this Note are issue management, risk management, change management, schedule management, and status reporting.

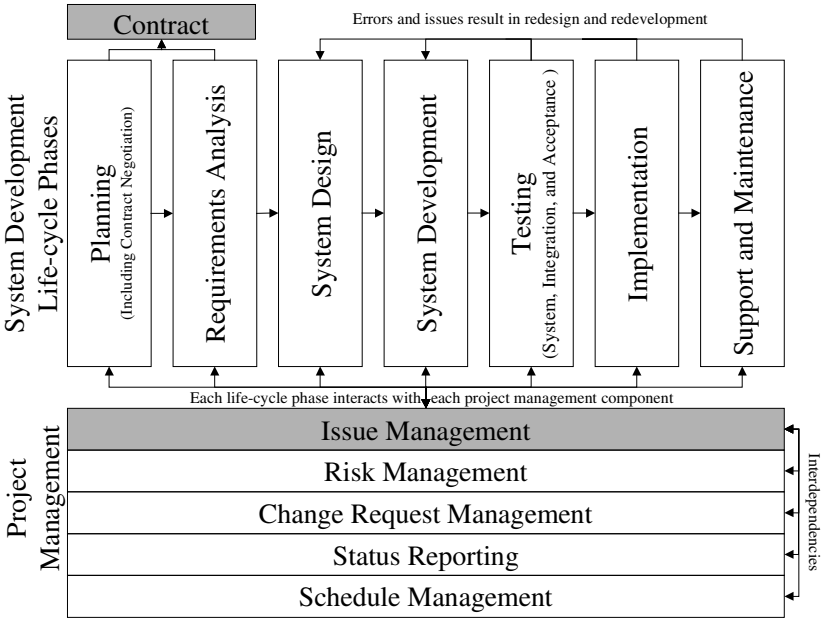


Figure 1: Simplified View of a Typical Software Development Project

In order to understand how and why the issue management process should be included in the contract, a basic legal understanding of Figure 1 is necessary. The next section discusses the planning and requirements analysis phases and the formation of the contract during those phases. Subsequent sections give a brief overview of each of the project management components and discuss how lawyers and contracts typically address each of these components. A final

32. The system development life cycle is the process of developing a system from start to finish. The process repeats itself as the system is upgraded and maintained.

section discusses the interactions between the Issue Management component, the other project management components, and the system development life-cycle phases.

1. Planning, Requirements Analysis, and the Contract

The contract for the design and development of a system is negotiated during the planning and requirements analysis phases of the project. The contract should be finalized after requirements analysis is completed.[33] The contract will define each party's obligations with regard to each of the systems development life-cycle phases and each of the project management components.[34] The planning phase of a systems development project includes the initiation of the project through a preliminary needs assessment, development of a project proposal or systems concept,[35] and completion of an operational and economic feasibility assessment.[36] The requirements analysis phase of the project defines functional user requirements "in terms of data, system performance, security, and maintainability for the system."[37] "All requirements [should be] defined to a level of detail sufficient for systems design to proceed."[38]

In an effort to combat many of the failure factors that have plagued information systems projects,[39] development contracts have focused heavily on the contractual statement of work.[40] A statement of work defines "the services to be performed; the

33. *See* 2 MICHAEL SCOTT, SCOTT ON COMPUTER LAW § 10.03[A] (2002); SIMON, *supra* note 2, § 5.01(b)(8).

34. *See* PETER C. QUITTMEYER, ET AL., COMPUTER SOFTWARE AGREEMENTS: FORMS AND COMMENTARY 5-3 to 5-18 (3d ed. 1998).

35. A systems concept is an overview or high-level description of the proposed computer system.

36. *See* Roche, *supra* note 1, at 11-12.

37. INFO. RESOURCES MGMT., *supra* note 31, § 1.2.3.

38. *Id.*

39. *See supra* notes 9-13 and accompanying text.

40. *See, e.g.,* Diana G. Richard & Michael K. Murphy, *Frequently Litigated Computer Software Contract Clauses: Contract Drafting Advice for the Computer Lawyer*, *in* INFORMATION TECHNOLOGY LITIGATION, REPRESENTING YOUR CLIENT IN SOFTWARE PERFORMANCE & SYSTEM FAILURE DISPUTES 53, 67-68 (Practising Law Inst. ed., 2001) ("[A] well-defined statement of work is of critical importance [to] . . . the successful completion of any engagement."); Davidson, *supra* note 23, at 13 ("[T]he single most important thing both parties can do before committing to the project is to spend the time necessary to develop a sufficiently detailed specification of what will be delivered.").

software to be developed, configured, or implemented; the functional objectives that the software must meet; the deliverables to be provided; milestones, performances and acceptance criteria; and the support, services and staff to be provided by the purchaser."[41] A statement of work that sets forth accurate and detailed requirements grounds the expectations of the parties in a common understanding and minimizes disagreements, thereby increasing the likelihood of project success.[42] However, more often than not, even a carefully bargained and drafted statement of work is not sufficiently detailed and clear.[43] Many software buyers lack the expertise or resources to produce a detailed statement of work and must rely either on outside consultants or the vendor to assist them.[44] When the vendor assumes both the roles of

41. Richard, *supra* note 40, at 68. Some confusion exists with the usage of terms identifying the contractual obligations of the parties. For purposes of this note, scope of work and statement of work will be considered the same. Requirements will be the technical and functional objectives that the software must meet without regard for whether those requirements were included in the statement of work or defined during an early phase of the project. *See* Charles Edison Harris, et. al., *Special Issues Relating to Software Development Contracts*, *in* NEGOTIATING COMPUTER CONTRACTS, *supra* note 30, at 252, 254, 269-73 (indicating that requirements are often defined in an early phase of a multi-phase project and explaining what factors should be considered with this situation). Frequently, computer acquisitions are initiated by requesting select vendors to respond to a Request for Proposal (RFP). *See* RAYMOND T. NIMMER, THE LAW OF COMPUTER TECHNOLOGY: RIGHTS, LICENSES, AND LIABILITIES ¶ 6.04 (3d ed. 1997). An RFP details the systems specifications and objectives of the acquisition. *See id.* However, an RFP does not automatically become part of the agreement between the parties unless specifically incorporated. *See id.* ¶ 6.04[2]. Therefore, an RFP should only be equated with requirements if the contract specifically provides for this. *See id.* Some contracts may document the technical and functional requirements of a system in the system acceptance criteria. *See id.* ¶ 6.05 (indicating that Sha I v. City of San Francisco, 612 F.2d 1215 (9th Cir. 1980) likely held that comprehensive acceptance criteria can fully delineate a seller's contractual obligations). If the seller's system passes all acceptance tests, then the seller has likely met its contractual obligations even if the system does not meet all of the buyer's needs. *See id.* Acceptance test criteria are part of acceptance testing conducted in the testing phase of the project. *See* Figure 1.

42. *See* Davidson *supra* note 23, at 13.

43. *See, e.g.,* SIMON, *supra* note 2, § 5.01(b)(2); QUITTMEYER, *supra* note 34, at 4-39; NIMMER, *supra* note 41, ¶ 6.01. For an unfortunate example of the disaster that results when the contract fails to adequately define the scope of work, see *Clay Bernard Systems International, Ltd. v. United States*, 22 Cl. Ct. 804, 807-16 (1991).

44. *Cf.* NIMMER, *supra* note 41, ¶ 6.04 (indicating that some buyers do not have the resources or expertise to conduct a needs analysis or acquire a vendor without outside assistance). If detailed requirements cannot be included in

design consultant and developer, the vendor takes on heightened obligations, even to the point of creating a "warranty as to the suitability of the product designed."[45]

In order to limit system warranties and obligations, vendors frequently attempt to integrate the contract to include only what is defined in the contractual statement of work.[46] An integrated contract is a complete and whole contract that should be interpreted only according to the terms within the contract and not according to prior oral or written agreements.[47] Integration is a frequently litigated issue because vendors often make broad oral and written statements during the proposal and bidding stages in order to win the contract.[48] Buyers rely on these vendor representations despite disclaimers in the contract.[49] Similarly, vendors rely on buyers to define system requirements that will ultimately satisfy the buyer's needs and expectations.[50] While some vendors make

---

the original contract, then either two contracts should be developed—one for detailing the requirements and one for development—or the contract should strictly forbid development activities from beginning prior to the definition and acceptance of the requirements. *See* SCOTT, *supra* note 33, at § 10.03[A]; SIMON, *supra* note 2, § 5.01(b)(8). If the vendor chooses to start development with incomplete specifications, then the vendor may "assume[] the risk[s] of inaccurate predictions as to the cost and time involved." NIMMER, *supra* note 41, ¶ 6.27[2].

45. NIMMER, *supra* note 41, ¶ 6.27[2]. In *NAPSCO Int'l, Inc. v. Tymshare, Inc.*, the vendor took on both the roles of design expert and developer and, therefore, "had a duty to alert [the buyer] to the areas in which [the buyer] or [the system being purchased] was lacking." 556 F. Supp. 654, 660 (E.D. La. 1983). The court further held that "'Sales-puffing' and silence are not defenses, especially not for the party with more information about the proposed system." *Id.* at 661 (footnote omitted).

46. *See generally* Richard, *supra* note 40, at 131-139 (detailing the issues surrounding software development integration clauses); SCOTT *supra* note 33, § 7.38 (discussing computer integration clauses).

47. *See* SCOTT, *supra* note 33, § 7.38.

48. *See 13 Ways to Avoid Risks of Buying a Large Accounting System*, *supra* note 3, at 5; Richard, *supra* note 40, at 73 & n.8 (citing as support *Cummings v. HPG Int'l, Inc*., 244 F.3d 16 (1st Cir. 2001); *APLications, Inc. v. Hewlett-Packard Co.*, 501 F. Supp. 129 (S.D.N.Y. 1980); and *Sound Techs., Inc. v. Hoffman*, 737 N.E.2d 920, 924 (Mass. App. Ct. 2000)).

49. *See* Brown, *supra* note 9, at 7 (acknowledging that buyers frequently have to rely on vendor representations about the vendor's product because the buyers lack computer knowledge).

50. *See* NIMMER, *supra* note 41, ¶ 6.27[2], at 6-121 to 6-122, ¶ 9.18 at 9-58 to 9-59 (explaining that the vendor cannot design software without understanding the needs of the end user and the buyer has an obligation to provide such information); *see also* Davidson, *supra* note 23, at 13 ("The customer should understand that if it receives only what is specified and no

representations unethically, many are just ignorant or misinformed about either the capabilities of their own systems or the true needs of the buyer.[51] All representations upon which the buyer relies should be detailed in the contract,[52] no matter how positive and trusting the relationship with the vendor may be,[53] because these representations become express warranties when included in the contract.[54] If the project ends

---

more, it will have gotten all that it bargained for and will have no basis to complain if it turns out that what it bargained for is not really what it needed."). Contracts often include "force majeure" clauses that force the buyer to accept responsibility for failing to providing information to the vendor. *See* QUITTMEYER, *supra* note 34, at 4-8 ("Consultant shall not be liable to Customer for any failure or delay caused by events beyond Consultant's control, including, without limitation, Customer's failure to furnish necessary information, . . . failures or substitutions of equipment, . . . shortages of labor, fuel, raw materials or equipment, or technical failures.").

51. *See 13 Ways to Avoid Risks of Buying a Large Accounting System*, *supra* note 3, at 4 (indicating that deceit isn't usually the problem, but more commonly just an "'honest disconnect' between what the user truly needs and expects and what the vendor can truly deliver.") (quoting Rauer L. Meyer, Partner in the Technology Department of Thelen, Reid & Pries); *see also* NAPSCO Int'l, Inc. v. Tymshare, Inc., 556 F. Supp. 654, 660 (E.D. La. 1983) (observing that the sales and implementation representatives dealing directly with the buyer were never aware of the precise capabilities of the system being purchased).

52. SCOTT, *supra* note 33, § 9.03[D].

53. *See* THE LAW AND BUSINESS OF COMPUTER SOFTWARE, *supra* note 26, at § 14.02[b] (arguing that despite synergetic attitudes, contracting parties should always assume that a "Mack Truck" will hit the key synergetic players and that the replacement players will hate each other).

> It is a sure sign of a potential problem if the vendor refuses to agree to reasonable requests for warranty protection or to put prior written or oral commitments into the final agreement. At the other extreme, a vendor willing to guarantee just about anything probably has little to lose and should be dealt with accordingly.

SIMON, *supra* note 2, § 5.01(b)(5) at 136.

54. *See* NIMMER, *supra* note 41, ¶ 6.07[1] at 6-35 to 6-38 ("Representations in the written agreement concerning product specifications are often described as express warranties. They are enforceable according to their terms despite general language elsewhere in the contract that disclaims and excludes 'warranties.'") (citing as examples *Consolidated Data Terminal Co. v. Applied Digital Sys., Inc.*, 708 F.2d 385 (9th Cir. 1983) and *Fargo Mach. & Tool Co. v Kearney & Trecker Corp.*, 428 F. Supp. 364 (ED Mich. 1977)). An express warranty is an affirmation of fact or a promise, including a description, sample, and model, about the software made by the software vendor that became a basis for the bargain between the parties. U.C.C. 2-313. In addition to express warranties, software contracts may also carry implied warranties of merchantability and implied warranties of fitness for a particular purpose. U.C.C. §§ 2-314, 2-315; *see generally* Richard, *supra* note 40, at 116–39 (discussing the use of implied and express warranties in software development contracts). Another key reason the statement of work should be defined in

in litigation, earlier levels of trust between the parties will be relatively worthless compared to documented express warranties.

2. Schedule Management

Schedule management is the traditional means by which projects have been managed.[55] Schedule management includes management by task plans, PERT charts, and other CASE tools.[56] The goal is to complete the project within the anticipated timeframe. Unfortunately schedules are too frequently unrealistic.[57] By pressuring a project to meet an unrealistic schedule, the project is forced to fail when the schedule is exceeded, when budgets are exceeded in an effort to stay on schedule, or when the quality of the system is compromised in order to stay on schedule.[58] The contract needs

---

detail before the contract is signed is because prior to signing, the buyer has negotiating leverage to convince the vendor to agree to changes without having those changes impact scope, price, or schedule. *See* SIMON, *supra* note 2, § 5.01(b)(3) at 132.

55. *See* Robert L. Glass, *The "Date Wars" and Management by Issue*, 48 J. SYS. & SOFTWARE, 1999, at 1, 2. *See generally* James A. Ward, *Productivity Through Project Management*, *in* HANDBOOK OF SYSTEMS DEVELOPMENT, *supra* note 14, at 11, 11-20 (discussing techniques for managing by schedule).

56. *See* Glass, *supra* note 55, at 2; Steven Alter & Michael Ginzberg, *Managing Uncertainty in MIS Implementation*, SLOAN MGMT. R., Fall 1978, at 23, 23.

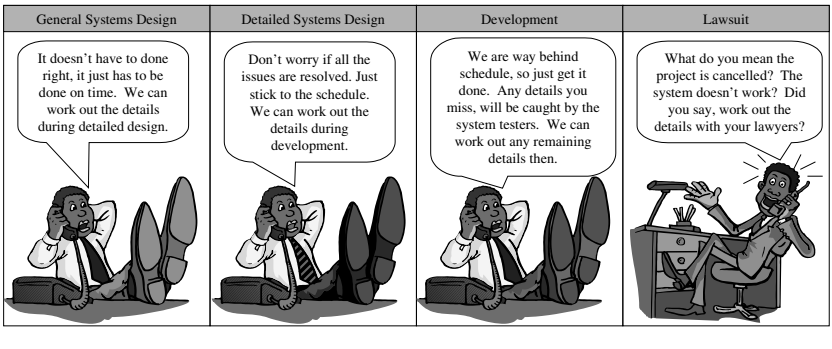
57. *See* Glass, *supra* note 55, at 2; *see generally id.* (criticizing the technique of managing by schedule). The year 2000 problem exemplified the inability of schedule management to define success. *See id.* During the year 2000 problem, project managers set unrealistic schedules the same as they always had done. *See id.* However, not only was the deadline for year 2000 projects inflexible, but so was the quality of the solution. *See id.* No artificial management schedule could change the level of effort that was needed to effectively address this problem. *See id.* "It takes nine months to make a baby, and no amount of management pressure will change that." *Id.* In the end, significant, unanticipated resources had to be dedicated to solving the year 2000 problem because schedule and cost had to give way to quality. *See id.*

58. *Cf.* Addario, *supra* note 12, at 13-14 (arguing that if schedule is the most important element of a project then cost and performance, including quality, will suffer). The dynamics of a project are depicted in the following diagram:

to include a set schedule. However, the schedule should include time buffers for unanticipated issues.[59] The project must be able to extend the schedule by agreement of the parties.[60]

Management Elements				
Maximize	Constrain	Accept		
		X	Performance	Effort Elements
	X		Cost	
X			Schedule	

*Id.*at 13. As shown by the diagram, only one element of a project can be maximized, constrained, and accepted. *See id.* If schedule is the most important part of a project, then cost and performance have to suffer. *See id.* Projects have to align their objectives with the priorities they are placing on each of the management elements. *See id.* It is impossible for a project to maximize performance without being willing to pay for that performance in terms of time or money. *Id.* The following cartoon, based on the authors first systems project, illustrates the negative impacts of emphasizing schedule at the expense of quality.



59. *See* THE LAW AND BUSINESS OF COMPUTER SOFTWARE, *supra* note 26, at § 14.02[c] (urging that the contract anticipate delays and built them into the agreed schedule).

60. *See id.* (urging that the contract provide means by which the parties can extend the schedule with and without penalties and ultimately provide for cancellation of the contract in the event of unacceptable schedule delays).

3. Status Reporting

Status reports are periodic summaries of the progress being made on the project along with summaries of unresolved problems and plans for addressing the problems.[61] Status reporting is the means of communicating status reports to managers and to others throughout the project. The contract should require status reporting and status meetings because the complex nature of system development projects "requires that the progress of the work be tracked, that . . . problems be documented and resolved, and that the continuing performance of all parties be monitored."[62]

4. Change Request Management

Even if a contract is successfully integrated and includes a detailed statement of work, changes are inevitable[63] and change has to be anticipated within the contract.[64] In fact, change is a necessary and desirable part of the project.[65] Contracts may include provisions detailing a formal, written change order process.[66] Change orders are the formal mechanism by which the parties amend the contract.[67] The contract should be drafted so that a change order does not require a renegotiation

61. *See id.* § 14.03[d] ¶ 215.5, at 14-16; SIMON, *supra* note 2, § 5.01(b)(25)(D), at150.

62. Gordon, *supra* note 24, at 18.

63. *See* Addario, *supra* note 12, at 14 (indicating that changing requirements are one of the top problems facing projects and are an expected part of any project). Technology is advancing at such a rapid pace that needs change before the project can be completed. *See* SIMON, *supra* note 2, at 170 (recognizing that change is "necessitated by changing or newly recognized business needs and requirements"). Technology not only changes the needs of the business for hardware and software (e.g. speed, performance, and tools), but technology also enables new business processes and methods, thus changing the functional business needs.

64. *See, e.g.,* SCOTT, *supra* note 33, § 10.04[B] (explaining the need for detailing a change order process in the parties' agreement).

65. *See* QUITTMEYER, *supra* note 34, at 5-5.

66. *See* SCOTT, *supra* note 33, § 10.04[B]*;* NIMMER, *supra* note 41, ¶ 6.28, at 6-123, ¶ 9.20 at 9-61 (indicating that because designs and specifications are frequently modified, the contract should specify how to deal with these modifications). However, in practice, contracts frequently do not specify how to deal with modifications. *See id.* ¶ 6.28, at 6-123

67. *See* Peter Vogel, *System Acquisition: Protecting the User, in* 19TH ANNUAL INST. ON COMPUTER LAW 1083, 1092 (Practising Law. Inst. ed. 1999).

of the entire contract.[68] By anticipating formal contractual changes, contract provisions, such as warranties and system acceptance requirements, should already encompass and account for future changes.[69]

Given the frequency and certainty of change in a software development project, change orders are a significant element of the original contract. For example, due to changing business needs and technological advances, a five-year, multi-million dollar development project is not likely to implement the system originally anticipated in the contract. In order not to delay the progress of the project, the contract should require prompt decisions on proposed change orders.[70] The contract should make clear who has authority for each party to propose, modify, approve, or reject proposed change orders.[71] Because change orders are changes to the contract, written approval of these changes by both parties should be required.[72] Written approval by the appropriate managers encourages proper evaluation of all changes and a consideration of the impacts of such changes on the project.[73]

5. Risk Management

Information systems engagements typically manage uncertainty by managing project risks. Risk management is "[t]he systematic application of management policies, procedures, and practices to the tasks of identifying, analyzing, assessing, treating, and monitoring risk."[74] Risks are a "measure of the probability and severity of adverse effects" upon objectives.[75] Examples of software development project risks include:

---

68. *See id.*

69. *See id*.

70. *See* Gordon, *supra* note 24, at 13, 20.

71. *See* Vogel, *supra* note 67, at 1087.

72. *See* THE LAW AND BUSINESS OF COMPUTER SOFTWARE, *supra* note 26, § 14.03[e].

73. *See* SCOTT, *supra* note 33, § 10.04[B].

74. Doughty, *supra* note 14, at 37.

75. *See* Yacov Y. Haimes, *Risk Analysis, Systems Analysis, and Covey's Seven Habits*, 21 RISK ANALYSIS: INT'L J. 217, 220 n.4 (Apr. 2001); Doughty, *supra* note 14,at 37.

- "The system cannot be developed on time or within budget."[76]
- "The system fails to meet current or future needs of users because . . . the environment [has] change[d] so that it is no longer functionally appropriate."[77]
- "Interorganizational factors hinder progress as a result of perceived system threats."[78]
- "The system will not generate the forecasted returns on investment."[79]
- "Development attempts to go beyond what is technologically feasible."[80]

Risks are mitigated so that their adverse affects either never materialize or are minimized.[81] One of the main roles of the lawyer is to anticipate risks and to mitigate them in the contract.[82]

6. Issue Management

Projects frequently encounter two types of changes: changes that impact the scope of work agreed to in the contract, namely schedule and price; and changes that are "within the estimating and performance risks undertaken by the vendor."[83] Changes that alter the contractual scope of work are dealt with as change orders.[84] Changes that do not alter the contractual scope of work are issues.[85] Both contractual and non-

76. SUSAN A. SHERER, SOFTWARE FAILURE RISK: MEASUREMENT AND MANAGEMENT 27 (1992).

77. *Id.* at 28.

78. *Id.*

79. *Id.*

80. *Id.*

81. *See* Alter, *supra* note 56, at 23, 28. *See generally id.* (discussing risk management theories); Paul Cule *et al.*, *Strategies for Heading Off IS Project Failure*, INFO. SYS. MGMT., Spring 2000, at 65 (discussing risk management through categorization of risks).

82. *Compare* Doughty, *supra* note 14, at 49-68 (listing common projects risks, risk impacts, and risk treatments) *with* QUITTMEYER, *supra* note 34, at 5-3 to 5-18 (explaining a sample software development contract) (the comparison will show that many of the risks perceived by information systems professions are addressed by the exemplified legal agreement).

83. SIMON, *supra* note 2, at 149.

84. *See supra* notes 63-73 and accompanying text (discussing change orders).

85. The distinction between change orders and issues is important. If too many changes impacting the system price or project schedule are

contractual changes need to be controlled because both have the potential of disrupting the project.[86] However, issues are much more than changes. Issues were defined above as obstacles, questions, or problems that arise during the project and that threaten to disrupt the progress of the project.[87]

---

mischaracterized as issues instead of change orders, then the buyer could be in danger of breach of contract. *See* NIMMER, *supra* note 41, ¶ 6.28 (indicating that recurring changes without compensatory adjustments hinders the vendor's ability to perform and would be an adequate basis for the vendor to cancel the contract). The timing of a change may affect the characterization of the change as a change order or as an issue. During the design phase of the project, a change that modifies the functional contractual requirements may be incorporated without an impact on the price of the system or on the project schedule. *See id.* (indicating that changes made during the design phase of the project "are within the contemplation of the parties and involve no added costs"). However, this same change, if requested during the development or testing phase of the project, may significantly impact price and schedule due to the rework that may be required. *See id.* An issue may result in a decrease or increase of scope – frequently without generating a change order. *Cf.* SIMON, *supra* note 2, at 149 (indicating that some changes are "within the estimating and performance risks undertaken by the vendor."). For example, the parties may have had a misunderstanding about one of the contractual requirements. Once the issue is resolved the parties determine that the correct understanding is already reflected in the contract, even though this results in a greater project scope of work. As another example, during the design phase, technical issues frequently arise that do not change the scope of contract requirements but do significantly affect the functionality of the system.

86. *Cf.* Thomas Fleishman, *Change Control and Problem Tracking Systems, in* HANDBOOK OF SYSTEMS DEVELOPMENT, *supra* note 14, at 803, 804-05 (discussing the fact that large and small changes have to be controlled during the support and maintenance phase of a project because both have the potential for disrupting service levels). "The misconception that only major changes . . . should be formally controlled prevents the fostering of effective change control management." *Id.* at 804.

87. *See* Robert L. Glass, *Issue Management*, DATABASE FOR ADVANCES IN INFO. SYS., Fall 1998, at 16, 17 (concluding that issues threaten to disrupt and "derail" the project); K.C. Burgess Yakemovic & E. Jeffrey Conklin, *Report on a Development Project Use of an Issue-Based Information System, in* PROCEEDINGS OF THE CONF. ON COMPUTER-SUPPORTED COOPERATIVE WORK, 105, 106 (1990) (defining issues as questions or problems). Issues can frequently be categorized as schedule/progress, resource/cost, growth/stability, product quality, development performance, and technical adequacy. *See* JOHN MCGARRY, ET AL., JOINT LOGISTICS COMMANDERS, PRACTICAL SOFTWARE MEASUREMENT: A GUIDE TO OBJECTIVE PROGRAM INSIGHT 26 (Version 2.1, Mar. 27, 1996). In one sense, risks are just issues that have been foreseen by management. *See* Glass, *supra,* at 16 ("If all issues could be identified in advance, they would probably be addressed as risks."). While technically correct, to equate issues and risks is inadequate. Many issues would only be considered identified risks in the abstract or in the aggregate. For example, an issue might be as simple as that a developer cannot proceed unless he knows the data values for a status field on a customer invoice and the

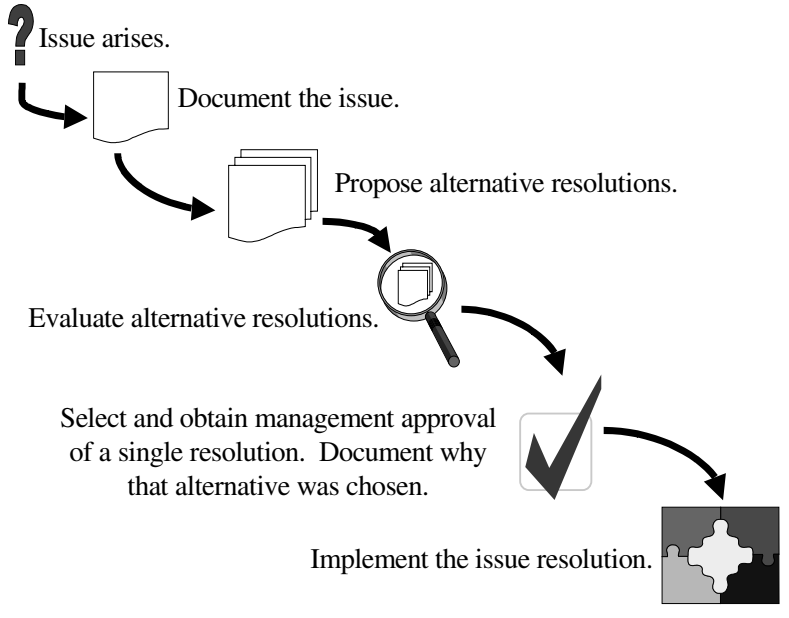


Figure 2: Issue Management Process Overview

Figure 2 above presents an overview of the issue resolution process. Once an issue arises, the issue must be documented and communicated to the appropriate decision-makers.[88] Alternative resolutions to the issue are proposed and evaluated.[89] A single alternative is chosen as the resolution to that issue.[90] Management approves the issue resolution.[91] The

customer does not know what the values should be. This simple issue would only be considered a risk in the generic sense that the requirement specifications may be defined too inadequately for design and development to proceed. Conversely some issues would have been identified as risks if anticipated. For example, an issue might be that two mission-critical systems cannot interface as required due to technical constraints. This same issue could have been anticipated as a risk. Issues are not limited to obstacles that management should have anticipated. Project managers must manage both risks and issues. *See* Glass, *supra,* at 18.

88. *See* Yakemovic*, supra* note 87, at 106, 108, 113.

89. *See id.* at 106, 113, 115; QUITTMEYER, *supra* note 34, at 5-11 (exemplifying a contract where developers were contractually required to evaluate issues and change orders at no additional cost). Analysis of the various alternatives should include an estimate of the cost for implementing the issue resolution and the impact that the resolution will have on the project. *See id.*

90. *See* Yakemovic*, supra* note 87, at 106. One study found that the team

issue resolution documents why that alternative was selected.[92] Finally, the issue resolution is implemented and woven into the rest of the project. Figure 3 below presents the various levels of approval that an issue may have to go through depending on the issue's impact on the project.[93]

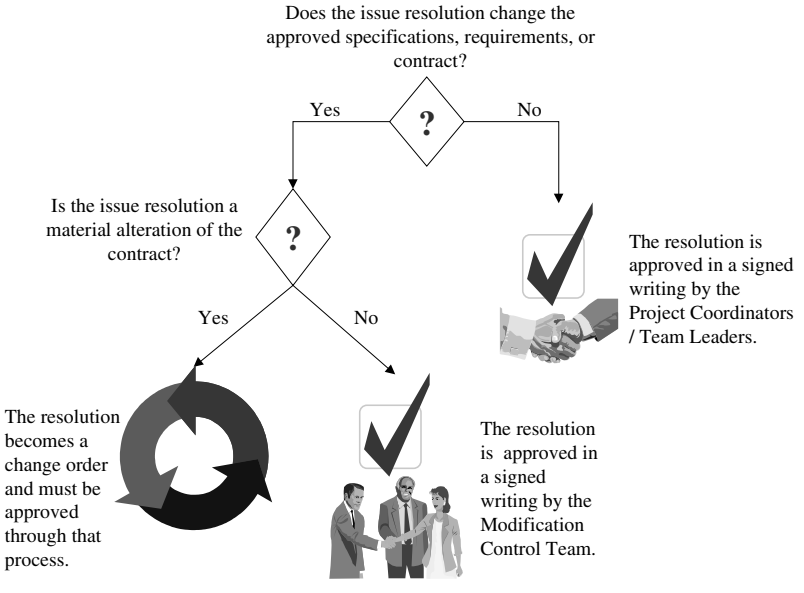


Figure 3: Various Levels of Issue Approval

Issue resolutions follow one of three paths:

1. *Project Coordinators.* If the issue resolution is more in the nature of a clarification than a substantive change to

---

using the issue management process had little difficulty picking resolutions and more difficulty defining alternative resolutions. *See id.* at 115. However, the issue management process aids and encourages the exploration of various alternatives. *See id.*

91. QUITTMEYER, *supra* note 34, at 5-11.

92. *See* Yakemovic*, supra* note 87, at 112.

93. QUITTMEYER, *supra* note 34, at 5-11 (laying out the process shown in as a sample contract).

refers to "issues" whereas the Quittmeyer treatise treated this approval process as part of a larger change control process rather than as separate issue and change management processes. *See id.* at 5-10 to 5-11.

contractual requirements, then a vendor and a buyer project coordinator approve the issue resolution.[94] Project coordinators are typically low- to mid-level managers, such as team or group leaders.

2. *Modification Control Team.* If the issue resolution results in a change to the approved project designs, specifications, or requirements but does not result in a material change to the contract, then the modification control team, consisting of at least one buyer representative and an equal number of vendor representatives,[95] approves the issue resolution.[96]

3. *Change Order.* If the issue resolution results in a material change to the contract, such as a change to the project's schedule or scope, then the issue resolution becomes a change order and must be approved independently through the change order process.[97]

Issues must be resolved in order for the project, or some part of it, to continue progressing. Resolving issues is "the essence of system management."[98] Occasionally, in an effort to avoid conflict, parties ignore or fail to identify issues.[99] Managers are often so busy managing risks (potential issues) that they fail to manage issues immediately before them. However, selective ignorance of issues leads to increased conflict and management difficulties as problems escalate and become more difficult to address.

The management of issues is "not simple."[100] Managers cannot resolve issues by any singular formula or methodology because they are "extremely project-specific."[101] Managers must be "nimble"[102] and technologically savvy in order to effectively resolve issues.[103] In order for the issue process to be successful, project personnel should feel free to bring issues to the attention of management and management must be competent

---

94. *See id.* at 5-11.

95. *See id.* The buyer and vendor shall each have one undivided vote, regardless of the size of the team. *See id.*

96. *See id*.

97. *See id.* at 5-11 to 5-12.

98. *See* Glass, *supra* note 55, at 2.

99. *See* Brown, *supra* note 9, at 3.

100. Glass, *supra* note 87, at 17.

101. *Id.*

102. *Id.*

103. *See* Robert L. Glass, *The "Date Wars" and Management by Issue*, 48 J. SYS. & SOFTWARE, 1999, at 3.

at resolving these issues.[104] Issues must be managed from the outset of the project for the process to be successful.[105] Further, management should prioritize issues according to their level of organizational importance.[106]

Documenting issues and having the relevant parties sign-off on the issue resolution has the benefits of:

- *Capturing the why.* The issue management process preserves the rationale behind a decision for future reference by developers, users, managers, and lawyers.[107] "[T]here is a need during development to capture the rationale – the why that underlies the what – behind large and complex computer systems. More precisely, there is a growing appreciation of the cost of failing to capture this information."[108]
- *Avoiding rehash.* An understanding of why decisions were originally made avoids wasting resources and rehashing decisions because no one recalls how decisions were previously resolved.[109]
- *Improving system maintenance.* "[T]he maintainers of large systems can not reliably make changes to the code without understanding the reasoning, or plan, that was used by the system developers."[110]
- *Saving money through early problem detection.* Projects that successfully resolve problems and remove obstacles at each stage of the project increase the likelihood of success.[111] The issue management process results in more problems being identified and dealt with early in

104. *See id.* at 2-3.

105. *See* Yakemovic*, supra* note 87, at 111.

106. *See* Glass, *supra* note 87, at 17.

107. Jeff Conklin & Ed Yourdon, *GroupWare for the New Organization*, AM. PROGRAMMER, Sep. 1993, at 5 (summarizing that a formal issue management process creates an "organizational memory" for the background and rationale behind decisions).

108. Yakemovic*, supra* note 87, at 105 (internal citations omitted). Part of the reason for memorializing the *why* is that key people with this knowledge frequently leave the project and, without written documentation of decisions, this knowledge leaves with them. *See id.* at 112. Another reason for documenting issue resolutions is that future changes may necessitate a reversal of a prior decision and the parties will need to understand *why* this earlier decision was made so that hidden dependencies are not ignored. *See id.*

109. *See id.* at 105.

110. *Id.* (citation omitted).

111. *See* Alter, *supra* note 56, at 26.

the development cycle when they are less costly to deal with.[112]

- *Improving the timing and quality of problem resolutions.* Documenting issues in written form helps decision-makers to understand the issue they are trying to solve more quickly than if the issue were not documented.[113] Written alternatives and evaluations of those alternatives aid in understanding the alternatives, identifying key assumptions, identifying weak or missing supporting arguments, identifying unique angles that might otherwise be overlooked, and, ultimately, making more timely and effective decisions.[114]

---

112. *See* Yakemovic*, supra* note 87, at 105, 113.

> The results of this study suggest that if design rationale is documented in an [issue management process], the process of performing the review and update of this information may pay for itself, by allowing more problems to be found earlier in the development cycle, when they are less costly to repair.

Id. at 113. The study found that maintaining the issue management system "paid for itself" by helping the design team detect eleven problems that would not have discovered otherwise until the system development and testing phases of the project. *See id.* The early discovery of these problems led to a savings of three to six times the actual cost of managing the issues, calculated in man-hours. *See id.; but see id.* at 105-06 (indicating that it is very costly to capture and organize issue resolutions without "very powerful technology"). However, since this observation was made in 1990, technology has advanced and become very powerful. In fact, the study involves a tool emerging in the 1990's that potentially had this very capability. *See id.* at 109-10. Today, capturing and organizing large amounts of data is relatively easy and cheap. Relational databases and GroupWare tools, such as Lotus Notes and Microsoft Exchange, make capturing, organizing, and retrieving issues technologically simple and cost effective.

113. *See* K.C. Burgess Yakemovic & E. Jeffrey Conklin, *Report on a Development Project Use of an Issue-Based Information System, in* PROCEEDINGS OF THE CONFERENCE ON COMPUTER-SUPPORTED COOPERATIVE WORK, 105, 113 (1990) ("[W]e found that the technique [of issue management] helped the team to more quickly understand the problem they were trying to solve.").

114. *See id.*

> *[E]xplicitly* stating the issues to be addressed provided a framework not only for the discussion of the document, but also for the entire development. By stating the requirements in terms of [i]ssues, [alternative resolutions] and [evaluations of alternatives], weak or missing supporting arguments were made apparent, and assumptions made by the document creator which were not common knowledge were frequently exposed. The approach allowed the group to propose solutions which satisfied the rationale more clearly than seemed to have been the case on earlier

- *Improving communication.* The issue management process encourages and improves communication and understanding within the project and with external groups.[115]

Despite the importance of issue management, the industry has not embraced the concept.[116] Issue management requires that the process be taken "very seriously, and [requires] understanding the value of carefully explored problems and rigorous decisions."[117] "So far, there have been very few software organizations ready to embrace the cultural shift that this implies."[118] However, in many circumstances software vendors have a legal duty to manage issues.[119]

7. Relationship of Issue Management to Other Phases and Components of the Project

Issues arise within, have effect on, and are affected by many other project management processes and phases of the development life cycle. As shown in Figure 1 the issue management process underlies all phases in the development life cycle because it is a process that can and should be used throughout the life of the project, and issues are generated and

---

> projects. . . . The information captured in an [issue management process] provides a different view of the software design than is presented by usual design documentation, so reviewing the issue-base allows the design to be reviewed from a 'different angle', exposing different problems than traditional design reviews.

Id.

115. *See id.* at 114 (indicating that the issue management process increased the effectiveness of project meetings and improved inter-organizational communication); Conklin, *supra* note 107, at 7 (indicating that projects that manage issues through a formal system report reduced face-to-face meeting times and increased levels of communication and coordination between and within teams).

116. *See* Glass, *supra* note 87, at 17-18 (indicating that there has been little academic or industry work in the area of issue management and what work as has been done has "faded into the woodwork").

117. Conklin, *supra* note 107, at 8.

118. *Id.*

119. *See* NAPSCO Int'l, Inc. v. Tymshare, Inc., 556 F. Supp. 654, 660 (E.D. La. 1983) (holding that due to vendor's expert knowledge of the system, the vendor owed a duty to the buyer to inform the buyer of problems in the system and to respond to the buyer's requests for changes); NIMMER, *supra* note 41, ¶ 9.20 ("The designer has a duty arising out of the interdependent relationship to report problems it knows of and, if the design process continues, to respond to the change requests and problems described by the customer.").

resolved during each of these phases. Each of the project management processes can generate issues that need to be tracked and resolved. Issues resolved in one life-cycle phase might impact other phases. For example, a defect found during the testing process may generate an issue as to how the defect should be resolved. The issue resolution may result in a change in requirements, a change to the system design, a redevelopment of software objects, a change to acceptance testing criteria, a change to the schedule, and creation of a risk to be monitored.

D. DISPUTE RESOLUTION

In order to resolve conflicts, contracts frequently include provisions for dispute resolution. Effective dispute resolution provisions function as deterrents against nonperformance.[120] Dispute resolution includes less formal methods, such as status meetings, and more formal methods, such as litigation, arbitration, and mediation.[121] Informal dispute procedures should include an escalation process where unresolved disputes get pushed up to higher and higher levels of management until the dispute is resolved.[122] Formal dispute resolution is usually not utilized until the project has already failed. However, formal methods could be employed anytime after informal methods, including escalation, have failed.[123] Figure 4 illustrates the hierarchy of escalation. Disputes should not be escalated to the next level until honest efforts at dispute resolution have failed at lower levels.[124]

120. *See 13 Ways to Avoid Risks of Buying a Large Accounting System*, *supra* note 3, at 6.

121. *See* Gordon, *supra* note 24, at 24; *see generally* SCOTT, *supra* note 33, §§ 7.49, 7.55–7.58 (describing litigation, arbitration, mediation, and mini-trials in computer contracts). Mediation is rarely found in vendor drafted software development agreements, despite mandatory nonbinding prelitigation mediation's ability to avoid unnecessary litigation. *See* SIMON, *supra* note 2, § 5.01(b)(22).

122. *See* Gordon, *supra* note 24, at 24.

123. *See id.*

124. *See* SIMON, *supra* note 2, at 180.

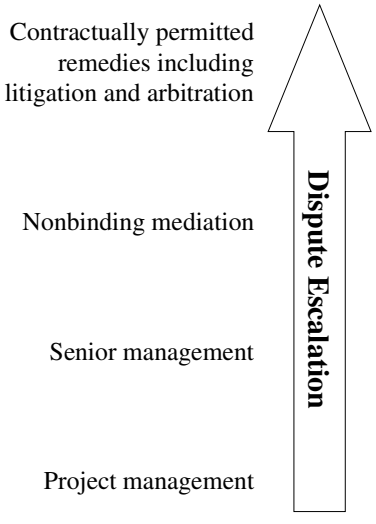


Figure 4: Escalation Hierarchy[125]

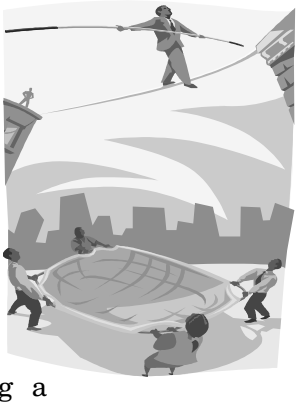
Alternative dispute resolution methods, such as arbitration and mediation, have the advantage of allowing the parties to choose a mediator who has technical expertise.[126] Theoretically, a technical mediator will be able to resolve disputes more quickly and efficiently because the mediator is more familiar with the issues than a non-technical person would be.

Despite the efforts of lawyers to draft contracts that mitigate risks and mandate specificity, software development projects continue to fail at an alarming rate. While much of the responsibility for failure falls on business managers, the legal community can do more to prevent these frequent, costly software development failures.

125. *See id.* at 179-80; *see also* QUITTMEYER, *supra* note 34, at 5-17 (indicating that prior to pursuing legal action the parties shall give senior executives a chance to meet, discuss, and resolve the dispute in an informal, amicable way).

126. *Cf.* Gwyn, *supra* note 7, at 413 (explaining that the Florida bar set up a voluntary mediation process for computer disputes using technically trained mediators).

II. A CONTRACTUAL ISSUE MANAGEMENT PROCESS FORMS A SAFETY NET FOR SOFTWARE DEVELOPMENT PROJECTS



Projects need to change and yet change is one of the common causes of project failure.[127] While all types of changes need to be managed,[128] this Note focuses only on changes that arise as issues and not as change orders to the original contract.[129] This note also focuses on issues that arise in the form of problems and obstacles threatening to derail the project.[130] Using the analogy of a high-wire circus act, the process of issue management is equivalent to walking a dangerous high wire. As evidenced by the high rates of project failure, projects that attempt to walk this high wire frequently fall to failure.[131] The role of the lawyer and of the contract should include the creation of a "safety net"[132] that will allow projects to walk the high wire of issue management with confidence, knowing that a safety-net will protect the client and the vendor and stop the project from falling all of the way to failure. As detailed below, the issue management safety net is woven when obligations, expectations, and capabilities are

127. *See supra* notes 63, 65 and accompanying text.

128. *See supra* notes 83-86 and accompanying text.

129. The change order process has been sufficiently addressed and discussed in the legal literature. *See, e.g.,* sources cited *supra* notes 64-73. Issues can become change orders if the issue resolution results in a material change to the contract. Some contractual changes will originally be identified as change orders and bypass entirely the issue process. *See supra* notes 83-84 and accompanying text for further discussion regarding the differences between change orders and issues.

130. *See supra* notes 87, 98 and accompanying text.

131. *See supra* notes 1, 6-10 and accompanying text.

132. Andy Anderson, Vice-President, American Management Systems, Inc., originally introduced the idea that project management processes should create a "safety net" for the project. Meeting with Andy Anderson, Vice-President, American Management Systems, in Richmond, Va. (Fall 1999). The safety net as introduced by Mr. Anderson was intended to encourage project members to "go out on a limb" and explore new ideas and processes with the security that the management processes would catch the project members if they took too many risks and fell from the limb. *Id.* The "safety-net" idea is used here with permission from Mr. Anderson.

aligned and when change is both facilitated and controlled.

Status reports are one simple example of a management process that forms a safety net. Status reports become a safety net when the information communicated allows managers to correct courses of action that are not aligned with the project objectives and plan.

A. CURRENT CONTRACTS FAIL TO CREATE AN ISSUE MANAGEMENT SAFETY NET

As traditionally developed, most contracts do not create a safety net that would facilitate issue management within the project. In fact, the focus of many traditional contracts is on processes that actually discourage issue management. All of the project management processes shown in Figure 1 and criticized below are essential project management processes that need to be included in the contract. The critique below is not intended to diminish the importance of each of the project management processes, but only to indicate how each of these processes impact issue management and how each is insufficient without a corresponding issue management process.

Schedule Management. Management primarily by schedule discourages effective issue management because issues, by definition, are problems and obstacles that threaten to disrupt the schedule.[133] Management by schedule focuses on decreasing an issue's impact on schedule rather than on facilitating a quality resolution beneficial to the overall project.[134] Further, issue resolutions sometimes result in changes to requirements that increase scope without extending the schedule.[135] Issues that impact schedule without becoming a change order frequently result when the vendor has made a unilateral mistake in estimating the level of effort required to implement a contractual requirement.[136] While schedule is extremely important, a project that ignores issues in an effort to remain on schedule is destined to fail because the resulting

133. *See supra* note 87 and accompanying text.

134. *See supra* note 58 and accompanying text.

135. *See supra* note 85 (discussing changes in scope that are with the performance risks of the vendor).

136. *See id.*

system will not meet business needs.[137] The fact that the unsatisfactory system was completed on schedule will not matter.

Status Reporting. Status reporting does not encourage issue resolution. Status reporting may be an effective way of communicating issues,[138] but the mere fact that issues are reported does not ensure that issues are resolved.[139] Reporting issues without a process to ensure resolution creates a false sense of security in the reporter who believes that if he merely reports the issue, someone reading the report will do something to resolve the issue. Status reporting is not an active issue resolution process nor does status reporting promote ownership of issues. In fact, status reporting may discourage the reporting of issues because project members and managers don't want to look bad by reporting problems in status reports. Status reports have a tendency to over emphasize successes and de-emphasize obstacles and problems.

Change Request Management. The change request management process is a vital component of issue management, but it is incomplete.[140] In fact, a change request management process without a complementing issue resolution process can be counterproductive to change because changes to the contract are generally discouraged. Change orders usually result in schedule extensions and increased costs. Because managers are hesitant to extend the schedule or increase costs, only the most significant and likely-to-be-approved changes are ever introduced into the change order process. Other less significant changes fall through the cracks because they are not tracked as change orders.

The change order process is not an appropriate avenue for evaluating all potential changes.[141] Change orders only address one category of changes, contractual changes.[142] The change request process is overly formal and slow for low-impact changes that are a normal, natural, and anticipated part of

137. *See supra* note 57.

138. *See supra* note 61 and accompanying text.

139. *But see supra* note 62 and accompanying text (inferring that the tracking, discussing, and monitoring of problems leads to the resolution of those problems).

140. *See supra* notes 83-86 and accompanying text.

141. *See supra* note 85.

142. *See supra* notes 83-84 and accompanying text.

producing the design.[143] Changes made during design can frequently be incorporated into the project without an impact on schedule or cost.[144] However, because these same changes alter requirements, they are still contractual.[145] While the change order process is not appropriate for these low-impact design changes, contractual formalities are required and they need to be highlighted separately from the design and approved specially.[146]

Risk Management. The risk management process is essential for avoiding potential adverse effects.[147] However, risks are anticipated events and the risk management process is insufficient for dealing with materialized issues.[148] Materialized issues require resolution, while risks require mitigation.[149]

Issue Management. The issue management process as described above is inadequate from a legal perspective.[150] First, the process is largely a business process that has not found its way into legal contracts.[151] With rare and incomplete exceptions, legal literature has not substantively discussed issue management.[152] One treatise has laid out the review process depicted in Figure 3 above, which recognizes that some changes should not be treated as change orders.[153]

Second, while there is limited recognition in legal literature for the need to distinguish between traditional change order management and other types of changes, there is no suggestion for separate change and issue management processes. However, the change order process is not an

143. *See supra* note 85.

144. *See id.*

145. *See id.*

146. *See supra* notes 93-97 (recognizing that the traditional change order process is insufficient for handling all types of changes)

147. *See supra* note 75 and accompanying text.

148. *See supra* notes 74, 87 and accompanying text.

149. *See supra* notes 81, 98 and accompanying text.

150. *See supra* part 0.

151. *See* sources cited *supra* notes 83-119 (exemplifying that the vast majority of sources describing the issue process are from business sources); *see also supra* note 116 (indicating that while issue management is extremely important, there has been little academic or industry research dedicated to the subject).

152. *See, e.g.,* sources cited *supra* notes 83, 85, 93, 99, 119.

153. *See supra* note 93 and accompanying figure (laying a process for approving various types of changes).

appropriate avenue for evaluating all potential changes.[154] By lumping issues and changes together, there is no recognition of the independent need to manage issues that are problems, questions, and other types of obstacles that are not contractual changes.[155] Because issues threaten to delay or derail the project,[156] issues must be resolved,[157] while many change orders merely need to be approved.[158] These crucial distinctions make inadequate any attempt to treat change orders and issues as the same.

Third, by not including an issue management process in the contract, the parties are not obligated to resolve issues in a timely manner. Issue management is hard, and in the momentum of a project, without a contractual obligation, the parties may choose to ignore issues completely.[159]

B. HOW TO CONTRACT FOR THE ISSUE MANAGEMENT SAFETY-NET

A properly contracted for issue management process is a combination of the existing issue management process (as outlined in the legal and business literature), together with elements from the statement of work and requirements analysis components, change request management process, status reporting process, dispute resolution procedures, and other general project and contract management techniques. Figure 5 below presents an overview of the issue management safety net.

154. *See supra* notes 141-146 and accompanying text.
155. *See supra* note 87 and accompanying text.
156. *See id.*
157. *See supra* note 98 and accompanying text.
158. *See supra* notes 71-73 and accompanying text.
159. *See supra* notes 99-100 and accompanying text.

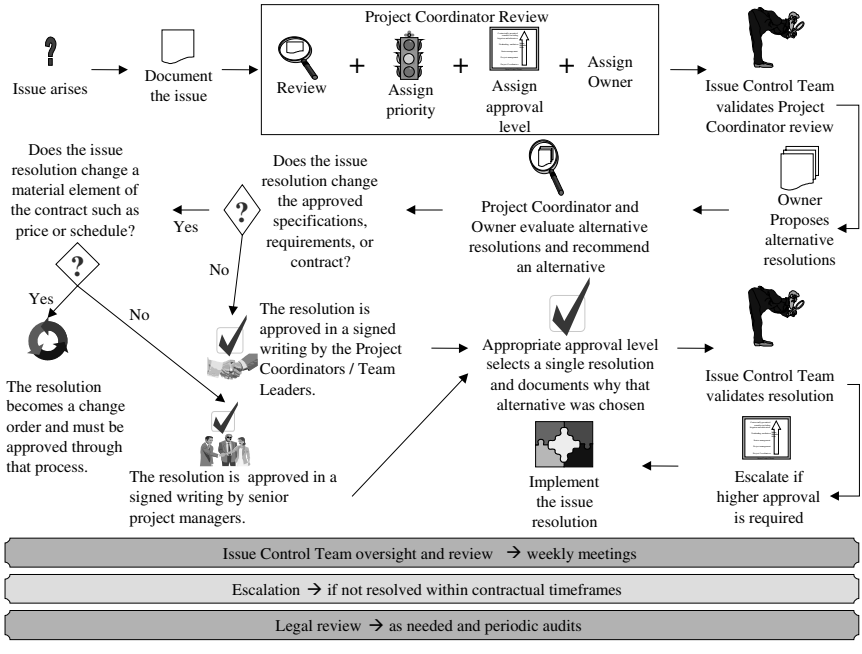


Figure 5: Issue Management Safety Net

1. Raising and Documenting Issues

Issues should be raised freely by any party to the contract and at any level of the project.[160] Frequently, designers and developers in the trenches are able to spot and understand issues that have eluded management. Issues should be raised in written form.[161] Issues of large and small significance can, and usually should, be tracked through the issue management process because all issues have the potential to disrupt the project.[162] Lawyers and project managers should ensure that other management processes, such as metrics,[163] do not discourage employees from raising issues.[164] Both the vendor and the buyer should view issue management as a tool towards ensuring that the buyer's needs are met and that the vendor is justly compensated and dealt with.

Practically, not all questions, problems, and obstacles will be tracked formally through the issue management process. However, issues not tracked formally do not bind the parties formally. The buyer can expect no more from the vendor than what the vendor is contractually obligated to provide, and visa versa.[165] Therefore, every representation upon which the buyer relies should be documented and agreed to in writing by both parties.[166] The issue management process is an ideal place to capture vendor and buyer representations and assurances. For example, a buyer could raise an issue about whether the system will support nine digit zip codes. The buyer might indicate that its current understanding, based on oral discussions, is that the system does support nine digit zip codes. A vendor that thereafter agrees in writing that the system does in fact support nine digit zip codes will be obligated to provide this functionality, even if the vendor was mistaken.[167] Buyers have a right to rely on vendor representations when the vendor has superior knowledge about

160. *See supra* note 104 and accompanying text.

161. *See supra* note 113 and accompanying text.

162. *See supra* note 86.

163. A simplified definition of "metrics" is measurements and data used by management to track and determine the status and progress of the project.

164. For example, metrics could discourage issue management if the number of issues generated was used as an indicator of the system's stability.

165. *See supra* note 50.

166. *Cf. supra* note 52 and accompanying text (indicating that all representations relied upon should be documented in the contract).

167. *See supra* note 54 and accompanying text.

its own product.[168]

Vendors, while more reserved, continue to make broad and lofty statements and promises even after the contract is signed.[169] Broad statements and promises are common if the project falls behind schedule or encounters other problems attributable to the vendor. The issue process forces vendors to either not make overly broad statements or be bound by and liable for those statements.

The formalities of the issue management process raise issue resolutions to the level of contractual obligations. These same formalities could discourage some project members from raising issues that have potentially adverse resolutions or political consequences. As an incentive to raise issues, the contract could include a clause holding a party that fails to raise an issue liable for the consequences of nondisclosure when the other party had no reason to know of the issue.[170]

2. Reviewing Issues

In order to minimize the unnecessary and inefficient commitment of buyer and vendor resources to the issue resolution process, a project coordinator should review all issues that are submitted before they are approved for evaluation.[171] A project coordinator is typically a low- to mid-level manager, such as a team or group leader. The project coordinator should review the issue for completeness and accuracy, and should ensure that the issue is not a duplicate.

The project coordinator assigns an initial priority to the issue.[172] As discussed below, the priority determines the timeframe within which a resolution must be agreed to. Priorities will generally be determined by the impact that the issue has on the project schedule. The priority can be changed at anytime to accurately reflect current realities.

The project coordinator assigns an initial approval level to the issue. As discussed below, the approval level is the management level that must approve the issue resolution. The approval level will generally be determined by the impact that

168. *See supra* note 119 and accompanying text.
169. *Cf. supra* notes 48, 51.
170. *See supra* notes 44-45, 119 and accompanying text.
171. *See supra* notes 100-103 and accompanying text.
172. *See supra* note 106 and accompanying text.

the issue has on the project and the contract. The approval level can be changed as more information is known about the actual impact of the issue resolution.

The project coordinator assigns an owner to the issue. An owner is a person who will take responsibility for the issue throughout the issue process, propose and evaluate issue resolution alternatives, and recommend a resolution. An owner essentially works the issue through to a resolution.

3. The Issue Control Team

The issue control team oversees the issue resolution process and ensures that issues progress to timely resolutions. The issue control team is made up of at least one vendor and one buyer employee.[173] If more than one person from either the vendor or buyer is on the issue control team, then the buyer and vendor shall each have one unified decision-making vote.[174] In the event that the issue control team cannot resolve a tie vote, then the issue is automatically escalated to senior project management for resolution as described below in the escalation process.

The issue control team shall meet on a regular basis, e.g. weekly, to discuss issues.[175] The team shall specifically review new issues and ensure that they have been categorized with the appropriate priority and approval level. The team shall discuss issues that need to be escalated to a higher level of management for either approval or for resolution, as discussed below. The team shall ensure that all issues are progressing towards timely, high-quality resolutions. In order to promote efficiency, the issue control team could consist of senior vendor and buyer project managers with authority to approve issues that alter the contract.[176]

---

173. *See supra* note 95 and accompanying text.

174. *See id.*

175. *Cf. supra* note 62 and accompanying text (indicating that problems should be discussed in regular status report meetings).

176. *See supra* notes 95-96 and accompanying text (indicating that the modification control team has authority to approve issues that do not result in a material change to the contract).

4. Evaluating Alternatives and Recommending a Resolution

The owner is responsible for drafting written alternatives and, along with the project coordinator, deciding on an alternative resolution to recommend for approval. The evaluation of each alternative and the reasoning for the recommendation should be documented in writing.[177] The issue process needs to capture the essence of the thought process behind a decision so that reviewers, approvers, and future readers of the issue statement and resolution can understand the *why* of the decision.[178]

In many aspects, issue resolutions are extensions and refinements of contractual requirements and of the statement of work. As with requirements, issue resolutions must be sufficiently detailed to allow systems design, development, and testing to proceed.[179] One of the main advantages of the issue resolution process is that alternatives are well thought through and resolutions are high quality.[180]

The process of evaluating and approving issues is a time-consuming process that requires the dedication of resources.[181] The contract must anticipate this level of commitment and the schedule and price should include such efforts.[182] While the process is time-consuming, the process itself adds value to the project by improving communication and understanding, rather than merely serving as a means to a resolution.[183] Because issues are difficult to resolve and potentially contentious, managers need a contractual impetus to ensure that the issue process is not avoided.[184]

5. Approving the Issue Resolution

The contract must precisely define the issue resolution approval process. Issues must always be approved in writing by both the vendor and the buyer.[185] Written approval can also

177. *See supra* notes 107-109, 114 and accompanying text.
178. *See id.*
179. *See supra* note 38 and accompanying text.
180. *See supra* note 90.
181. *See supra* notes 100-102 and accompanying text.
182. *See supra* note 89.
183. *See supra* notes 113-115 and accompanying text.
184. *See supra* note 99 and accompanying text.
185. *See supra* notes 72, 94-97 and accompanying text.

include electronic signatures. Approvers are responsible for ensuring that the issue resolution is of high quality, thorough, and that the impacts have all been considered.[186] Approvers should document any rationale for their approval so that the issue captures a complete understanding of why that decision was made.[187]

The contract must state which managers or management roles have contractual authority for approving issue resolutions of which types.[188] Because issues impact the project differently, and some are less significant and more at the *working-level* than others, the approval levels should vary to reflect these realities and to avoid overloading upper management.[189]

Table 1 below provides an example of different approval levels that might be included in the contract.

Issue Resolution Type	Contractually Permitted Approvers
Material change to the contract (e.g. change to schedule or cost) → Change Order	Senior Managers with authority to amend the original contract
Material clarification of the contract due to a unilateral mistake or misunderstanding	Senior Project Managers
Immaterial change to the contract (e.g. change to or addition of a requirement specification that impacts only functionality and not price or schedule)	Senior Project Managers
Clarification of an approved requirement, contractual specification, or design	Project Coordinators (e.g. group leaders and team leaders)

Table 1: Issue Approval Levels

186. *Cf. supra* note 73 and accompanying text (requiring approvers to evaluate the impacts of change orders).

187. *See supra* notes 107-109, 115 and accompanying text.

188. *Cf. supra* note 71 and accompanying text (requiring the contract to specify who has authority to approve change orders).

189. *See supra* notes 91-97 and accompanying text.

Typically a material change to the contract will be a change in project schedule or cost. However, some issues may alter the project schedule and/or cost and not be considered a material alteration of the contract.[190] Such issues are labeled material clarifications in Table 1. Significantly, issues may impact and even change the scope of work and still be within the limits of the contract.[191]

6. Escalating the Issue

Escalation involves the shifting of responsibility from one level to a higher level. Issues escalate under two circumstances.

a. Escalation for Failure to Resolve within Contractual Timeframes.

First, issues are escalated if they are not being resolved within the contractually required time frames.[192] One of the keys to making the issue resolution process function as a tool is to resolve issues in a timely manner. The priority assigned to an issue determines the timeframe within which the issue must be resolved.[193] The priority may be changed as the importance and impact of the issue changes upon further analysis. The contract should define priority levels and the timeframes within which issues of that priority must be resolved. Table 2 provides an example of the timeframes that could be assigned to priorities.

190. *See supra* notes 83, 85 and accompanying text.

191. *See id.*

192. *Cf. supra* note 70 and accompanying text (including in the contract a requirement that change orders be promptly addressed).

193. *See supra* note 106 and accompanying text.

Priority	Required Resolution Timeframe
Level 1 – critical work is stopped and cannot progress until the issue is resolved.	2 days
Level 2 – important work is stopped and cannot progress until the issue is resolved, or progress on critical work is hindered until the issue is resolved.	5 days
Level 3 – substantive work is stopped and cannot progress until the issue is resolved, or progress on important work is hindered until the issue is resolved.	10 Days
Level 4 – cosmetic work is stopped and cannot progress until the issue is resolved, or progress on substantive work is hindered until the issue is resolved.	14 days
Level 5 – suggestions, low-impact questions and concerns.	20 days

Table 2: Issue Priority Levels and Required Resolution Timeframes

If the issue is not resolved within the contractually required timeframe, then the issue automatically escalates to the next level. However, buyer and vendor senior managers with authority to amend the contract may agree to grant up to two reasonable extensions before an issue will automatically escalate. Responsibility for resolution shifts to the escalated level, as described in the next section, and someone from that level becomes the new issue owner. The issue requires higher approval because it is either more difficult to resolve than originally anticipated or because the current approvers, for whatever reason, are not resolving the issue. Upon escalation, the priority timeframe for resolution escalates one level. For example, a Level 3 issue to be approved by project coordinators that is not resolved within 10 days automatically becomes a Level 2 issue assigned to project managers and must be resolved within 5 days. This process repeats itself until the issue is ultimately resolved.

Some projects may want to contain issues within the phase in which they were raised. In this situation, the project should not progress to the next systems development life-cycle phase

until all open issues are resolved – unless exceptions have been granted in writing by all parties.

b. Escalation for Higher Approval

Second, issues are escalated to a higher approval level, if, after analysis, they are determined to have a greater impact than originally anticipated.[194] Figure 6 below provides an example of the various levels of approval that an issue can escalate through. Final approval of an issue is granted at the level agreed upon by the issue control team. The contract must empower employees at each level to approve issues assigned to them. For example, an issue appropriately approved by project coordinators has the same legal effect as issues approved by senior management.

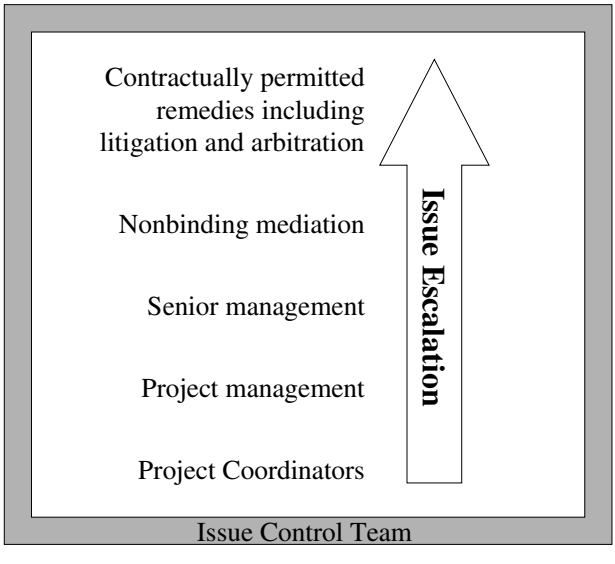


Figure 6: Issue Escalation[195]

The issue control team oversees the entire escalation process and ensures that issues escalate smoothly from one

194. *See supra* note 122 and accompanying text.

195. *Cf. supra* note 125 and accompanying figure (illustrating dispute escalation levels).

level to the next. The issue control team is also responsible for recommending that an issue approved at the project coordinator or project management level be approved by a higher level of management.

Mediation[196] and litigation[197] (or arbitration) should not be viewed negatively, but rather as tools for facilitating the progress of the project. Some issues may not be resolvable within the project due, for example, to contractual misunderstandings or personality conflicts. However, issues are obstacles that threaten to stop the progress of the project or even to derail the project, and, therefore, issues must be resolved, even if litigation is required. Mediation and litigation over a single issue makes sense if such steps allow the project to continue progressing and to avoid failure. Frequently, the knowledge that mediation and litigation will be contractually required will inspire managers to resolve issues on their own.[198] Viewed in this way, mediation and litigation become tools for success rather than failure.

An issue that escalates from one level to the next may escalate merely for approval or for further evaluation. Anytime an issue escalates, the higher-level approvers always have the option of reevaluating the proposed alternatives, if any, and proposing new alternatives.

7. Legal Review

Lawyers should be regularly involved in reviewing the issue process and specific issues.[199] Lawyers need to be involved because vendors have a legal duty to address issues.[200] Additionally, lawyers need to be involved because all issues have the potential of ending in mediation or litigation and the lawyer should not wait until this happens before becoming involved.[201] In fact, the lawyer should play an active role in helping the parties resolve issues before mediation or litigation is required.[202] Lawyers must do more than prepare for

196. *See supra* notes 121, 126-120 and accompanying text.

197. *See supra* notes 121, 123 and accompanying text.

198. *Cf. supra* note 120 and accompanying text (explaining that dispute resolution provisions function as deterrents to nonperformance).

199. *See supra* note 22 and accompanying text.

200. *See supra* notes 45, 119 and accompanying text.

201. *See supra* note 24.

202. *See supra* note 30 and accompanying text.

litigation, and must facilitate the resolution of issues.[203]

Lawyers also need to be regularly involved with issues because issues have contractual implications. The lawyer should ensure that issues are not being approved that really need to be designated as change requests to the contract. Further, issues individually may not materially impact the contract, but in the aggregate, issues have a significant impact on the contractual statement of work and project scope. The lawyer must ensure that if issues are changing the statement of work, that they are being appropriately approved by managers with authority to alter the contract, and that the resolutions are clear and sufficiently detailed.[204]

Lawyer involvement combined with the issue escalation process ensures that failing projects are either brought back on track or cancelled.[205] Projects that should be cancelled are frequently continued.[206] This is known as project escalation.[207] If a project must fail, the earlier the failure occurs, the better for all parties involved. By forcing resolutions to issues, problems that will cause failure are likely to surface earlier in the project. The lawyer should aid managers in identifying when a project should be cancelled.

8. Other Contractual Considerations

The contract must be drafted so that it encompasses the future resolution of issues without having to renegotiate the contract every time an issue impacts the contract.[208] Issue resolutions are express warranties, and warranty provisions in the contract must include future issue resolutions.[209] Any integration clause must include future issue resolutions identified through the issue process.[210] Issue resolutions must be contractually binding on the parties. The contract schedule and price should allow for time and resources to be dedicated to

203. *See supra* notes 27-29 and accompanying text.

204. *See supra* note 24.

205. *See supra* note 25.

206. *See id.*

207. *See id.*

208. *Cf. supra* note 68 and accompanying text (indicating that change orders should be anticipated in the contract so that each change doesn't require a renegotiation of the entire contract).

209. *See supra* note 54.

210. *See supra* notes 47-46 and accompanying text.

resolving issues.[211] Additionally, the schedule should be lengthened and the price should be increased in anticipation that some issue resolutions will be within the performance risks undertaken by the parties, even though not detailed in the contract.[212] For example, if one party negligently misunderstood a contractual provision, a subsequent issue clarification may affect the schedule and that party's costs, but that party will have no right to additional compensation or a schedule extension.

C. HOW THE ISSUE MANAGEMENT PROCESS CREATES A SAFETY NET

A properly constructed issue management process can become a safety net for a software development project. The issue management process combined with a change management process creates a safety net that gives a project the confidence and ability to change, address problems, and overcome obstacles in a controlled way. A safety net gives the project the confidence that it needs to explore changes and problems without having to worry about losing control and falling to failure. Thus, the issue management process enables change and decreases the risk of change at the same time.

When project members know that there is a management review process in place to check their actions, project members—particularly designers, developers, and testers—feel free to do their jobs and address actual client needs, rather than incomplete and outdated contractual requirements.

The issue management process is a safety net to a poorly defined statement of work.[213] The statement of work is almost always insufficiently defined.[214] The issue management process provides the means for creating legally binding obligations that should have been included in the statement of work. A majority of resolved issues are essentially the detail that was missing from the statement of work. Accurate and detailed issue resolutions ground the expectations and obligations of the parties in a common understanding.[215] Vendors who choose to

211. *See supra* notes 100-101 and accompanying text.
212. *See supra* notes 59, 83, 85 and accompanying text.
213. *See supra* note 40 and accompanying text.
214. *See supra* note 43 and accompanying text.
215. *See supra* note 42 and accompanying text.

ignore issues and develop the system without sufficiently detailed specifications, including issue resolutions, assume the risks of their inaccurate predictions.[216]

The issue management process is a safety net to the project schedule. Because issues are inevitable and because they, by definition, disrupt the project schedule,[217] the most effective way to minimize the disruption is through an issues process that identifies and resolves issues quickly and efficiently.[218]

The issue management process also creates a safety net for the vendor. Buyers frequently have expectations that do not match their contractual requirements.[219] The issue management process creates a safety net for the vendor because it provides an avenue for the buyer to clarify its expectations.[220] A buyer who does not clarify ambiguous requirements through the issue process cannot complain when the vendor provides a system that meets its contractual obligations.[221] The contractual obligations of the vendor are made clear by the contract, issue management process, and change management process. The vendor's safety net is a set of clearly defined obligations.

D. ISSUE MANAGEMENT SAFETY NET WILL INCREASE THE LIKELIHOOD OF PROJECT SUCCESS

Resolving issues is "the essence of system management"[222] and critical to the success of any project.[223] By including the issue management process in the contract, the parties are obligated to resolve problems from the outset of the project.[224] In order to demonstrate how issue management increases the likelihood of project success, reconsider the failure factors presented at the beginning of this Note, and consider how the issue management process counters many of these factors.[225]

---

216. *See supra* notes 44-45 and accompanying text.
217. *See supra* note 87 and accompanying text.
218. *See supra* notes 111-112 and accompanying text.
219. *See supra* note 43 and accompanying text.
220. *See supra* note 50 and accompanying text.
221. *See supra* note 50.
222. *See supra* note 98.
223. *See supra* note 87 and accompanying text.
224. *See supra* notes 99, 105 and accompanying text.
225. *See supra* Part 0.

- *Incomplete and changing requirements and specifications.* The main purpose of the issue process is to provide a means to clarify incomplete requirements and specifications. The issue process further provides a means for controlling changing requirements. Requirements cannot change without review and approval by management. The scope of the project is kept under control by regular monitoring of the issue process by management, the issue control team, and lawyers.
- *Poor communication.* The issue management process is a communications process.[226] The issue process facilitates communication not only about obstacles and problems, but also, more importantly, about what is being done to resolve those obstacles and problems and why it is being done.[227]
- *Inconsistent decision-making.* The issues management process is a process for capturing information about decisions so that there is a record of what decisions have been made and why.[228] By capturing decisions, inconsistencies become apparent and are reduced.[229] The issue management process improves the timing and quality of decisions.[230]
- *Poor project planning – including inadequate risk management, budget overruns, and schedule overruns.* The issue management process finds problems earlier in the project life cycle and thereby significantly decreases the effects that problems have on the schedule and the budget.[231] The issue process is essentially a planning process for problems, changes, and obstacles that were not properly included in prior project plans.
- *Lack of top management involvement and support.* Through the escalation and approval process, top management is actively involved with the project and, more importantly, with the struggles the project is facing. Issues are a key way for management to keep a finger on the pulse of the project.

---

226. *See supra* note 115 and accompanying text.
227. *See supra* notes 107-108 and accompanying text.
228. *See id.*
229. *See supra* notes 109-110 and accompanying text.
230. *See supra* notes 113-114 and accompanying text.
231. *See supra* notes 111-112 and accompanying text.

- *Lack of end-user involvement and support.* The issue process is open to all users and users of all levels are encouraged to raise concerns through this process.[232] End-users have an opportunity to review and evaluate issues. Additionally, by documenting issue resolutions, the issue management process is a key way to communicate decisions and to obtain support for what is being done on the project.[233]

CONCLUSION

Traditional contracts have ignored the essence of project management by failing to sufficiently provide for the resolution of issues. This Note proposes that issue management be contracted for in detail in software development contracts. By contracting for the issue management process, contracts will require the parties to address problems head on, in a timely and efficient manner, and in a contractually binding way. The issue management process addresses the dynamic and changing needs of software development projects not captured in more traditional, static, development contracts. This Note concludes that the inclusion of an effective issue management process in software development contracts will create a safety net for development projects and will increase the likelihood of project success.

232. *See supra* note 104 and accompanying text.
233. *See supra* notes 107-108 and accompanying text.