RESCUING FLAMING PROJECTS!

Pradeep Anand and Mohit Mahendra

March 2004

The software industry is a glowing world of flaming projects that are in destructive dives. Every aspect of the industry- developing, migrating, maintaining, or implementing software packages- has its unfair share of flaming projects that often can destroy if not severely damage companies' profitability.

Imagine this – you are the pilot of an airplane and instead of arriving safely at a pre-determined location, your engines are sputtering and it won't be long before you go into a destructive dive. Who do you call to rescue you?

It's people who have experienced similar situations and have survived them thanks to their ingenuity and quick thinking, discipline under fire, and passion to live another day. Similarly, turning around flaming software projects requires an organizational presence, mind-set, skill-sets, attitudes and processes that are focused on immediacy and urgency of turning around projects that have gone awry.

SYMPTOMS OF A FLAMING SOFTWARE PROJECT

In physical engineering projects, progress is visible and tangible. Consequently, lack of progress is also visible and discernible. Experts visit plants and structures, and inspect critical junctures in processes.

However, in software projects, "progress inspection" is a difficult task and more often than not, there is little progress inspection - smoldering projects are not always visible. Some broad indicators are used in the industry to anticipate a sputtering project, and to take corrective actions before the project goes into a destructive dive.

1. Budgets exceeded – When Cumulative Actual Cost of the project begins to exceed Cumulative Planned Costs, alarms are set off to stem the gap from growing in magnitude.
2. Missed Milestones – Time is a significant component of the planned Net Present Value and Returns on any project. Even if projects are under budget, the inability to meet milestones and schedules is another indicator of a soon-to-be-flaming project. Close attention should be paid to those milestones on the critical path of a project, and the time and value of the affected milestones.
3. Increasing Change Orders – While time and money are obvious parameters to monitor, a leading indicator to a flaming project is the number of changes and variations that are inserted in a project. Tracking cumulative changes and their incremental change in costs are invaluable in managing projects.
4. Mounting Issue Resolution Efforts - Another predictive tool for flaming projects is the quantitative and qualitative aspects of managing issues. These issues, when resolved lead to change orders and costs – either to the buyer or supplier.
5. Poor Quality – In engineering projects, poor or inadequate quality, and performance can be identified by simply lifting the hood – by inspecting and testing processes, procedures, components, sub-assemblies, and final assemblies. Additionally, the industry has been around for hundreds of years and testing resources and processes abound to accomplish these tasks. In the software business, even if you had the ability to conceptually lift a hood, it

takes special resources with great insight to determine adequacy in quality. The Software Engineering Institute (SEI) of Carnegie Mellon University issues CMM (Capability Maturity Model) certifications to software service organizations. CMM and similar software process methodologies are useful tools to guide process discipline and improvement in software service organizations. However, CIO magazine, in its March 1, 2004 cover story entitled 'Bursting the CMM Hype,' identified numerous shortcomings in this rating. The article cautioned against the risks of over-relying on the CMM score as an indicator of the quality results that could be guaranteed by a software services organization. Initially, a project benefits from a CMM rated methodology. But then, the vendor discovers there is far more to building a successful software solution than the process can possibly deliver. That's when the flames break out.

6. Excessive Personnel Movement – An additional early indicator of a flaming project is movement and disappearance of key personnel in a project. Software projects are difficult to manage, and reassignment of professionals to other or multiple tasks inadvertently and adversely affects quality, time, and costs.

RESCUING FLAMING PROJECTS

Despite best efforts, firms recognize failing projects only when they are in flames. Like many enterprises focused on "emergencies", turning around flaming projects require entirely different organization, processes and technologies - all focused on delivering great results in situations where clients' backs are to the wall.

Silicus has rescued numerous clients from flaming projects. And we have witnessed a pattern of events time and again. It is clear to us that a specific set of attributes is essential to supplement the practice of methodology.

ORGANIZATION & PEOPLE

Band of Brothers When the first sign of flames are visible, the critical ingredient for an effective rescue operation is problem-solving leadership and tight, cohesive, and focused problem-solving teams. This Band-of-Brothers is built by recruiting exceptional people who work well in teams, and who have the ability to handle ambiguity and complexity. They are able to bring an unfaltering focus, work-ethic, and attitude towards rescuing a client whose back is to the wall. Their problem-solving skills, understanding of root causes, and stamina to work towards accomplishing the project objectives and business goals are prized and rewarded.

Insight Senior managers define project goals and priorities, and communicate these, without ambiguity, to all stakeholders, project leaders, and team members. Senior managers provide project vision and business insight, while laying out the roadmap to successful project execution. With experience and insight, a methodology can work, and in its absence, it fails.

Software Engineering Culture The upside of a great software solution — extraordinary results — arises from the culture within a software engineering organization. It comes from "involved" software professionals who plan and track their work at a personal level, who find their work truly rewarding, and who are part of motivated engineering teams that use best practice methods in their work.

PROCESS

Figure 1 provides the process outline for rescuing flaming projects. There are four broad phases:

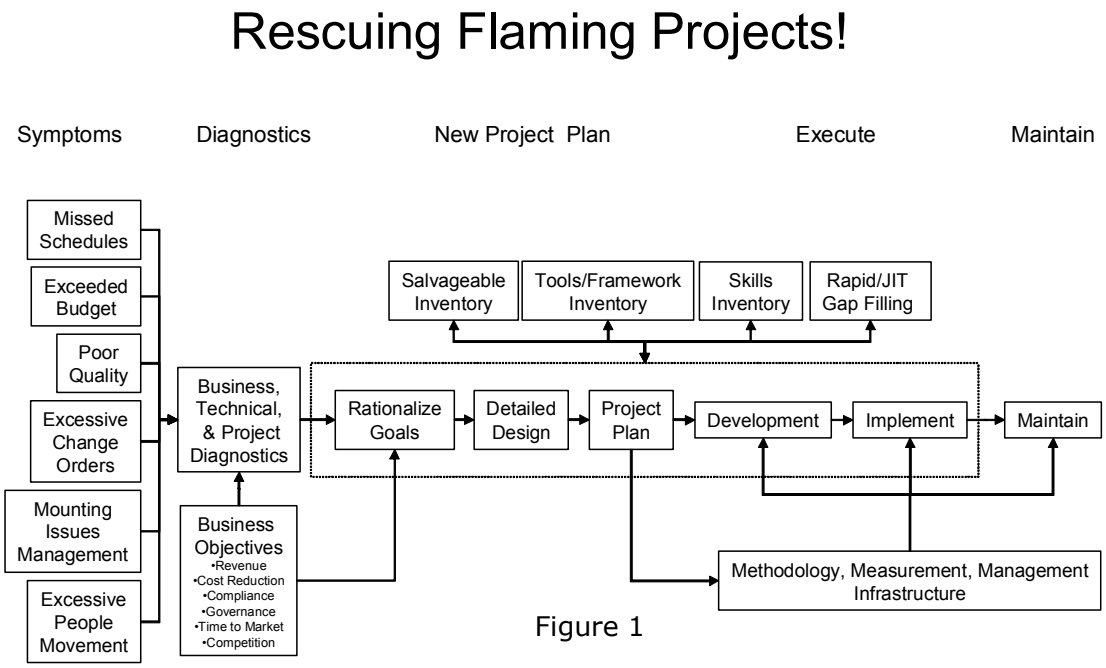

Figure 1

1. **Speedy Diagnostics** – In this phase, the problem project is diagnosed from business, technical and project management perspectives. Business objectives are revisited to ensure their alignment with the software project objectives. Additionally, the technical architecture is ensured to meet business objectives, while conducting trade-offs with constraints such as budgets, schedules, and performance. Stock-taking of the inventory of tools that has already been developed is conducted to identify what can possibly be salvaged in the new project design and plan. Additionally, project and other processes are evaluated for their robustness and ability to deliver the desired results. The deliverable at the end of this phase is rationalized project goals.

   Most important is developing a unique business needs-driven technical architecture that has undergone trade-offs in Budgets, Time-to-Market, Functional Performance and Total Cost of Ownership. This is an iterative process that yields the optimum architecture to meet business goals, project schedules and budgets, and long term quality and cost expectations. Priorities are established upfront and are tied to the client's specific goals and objectives. Invariably there are trade-offs necessary to accomplish project goals. When rescuing flaming projects, time-to-market is typically the main driving factor.

2. **Develop New Project Plan** – During this phase, the rationalized goals and the technical architecture are used to create a detailed design and a project plan, using the project's salvageable inventory, and the consulting firm's Tools & Framework Inventory. The firm's skills inventory also undergoes a thorough check. Gaps in skills and tools are identified and Rapid/Just-in-Time filling efforts are also included in the project plan. This project plan is then absorbed by a software process management infrastructure that becomes the operating substrate throughout the rest of the project.

Engineering wisdom recognizes that design errors are the most expensive to rectify. Design flaws often don't get discovered until late in the project, and can sabotage the success of a project. Impatient managers on tight schedules tend to gloss over early milestones, and want to dive immediately into coding and testing stages. Poorly guided development teams also tend to plunge quickly into development.

Smart software consulting firms recognize the value of a well thought out design strategy and place emphasis on every technical aspect that precedes coding and testing. From the outset, functional groups are brought together to define all aspects of the design—usability, functionality, testability, migration, production, scalability, and maintainability. Then, each design concept is tested as early in the schedule as possible. These steps may take up more time in a tightly scheduled project, but they dramatically reduce the risk of project failure.

If a project is about to erupt into flames, it's never too late to stop, identify the design flaws, and invest in redesign. Patches may seem like a quicker fix, but clients often end up unhappy and frustrated, limping along with a poorly designed system. The best way to get back on track is through smart reworking of the design, even if it means a fair amount of recoding.

3. **Swift Execution** – When a motivated team works on a prudent design combined with the right process methodologies, measurements and management, they deliver extraordinary results. These processes enable transparency of client's objectives to the rank and file of the project team- an imperative for avoiding slips between the cup and the lip – so that the deliverables are achieved on already constricted schedules and budgets.

4. **Method Improvisation** There are numerous methodologies today from which to pick and choose. Some tools and techniques, right for some situations, are totally wrong for others. Consulting practices that rigidly adhere to one methodology can cause clients real problems.

   The best practitioners are aware of the limitations of each methodology and know when to "break" with one to get the job done. They also know how to customize methodologies, blending a client organization's practices with their own. In jointly executed projects, parts of the project lifecycle may be executed within the client organization, and parts may be outsourced to a service provider. Such situations require a rational assessment of the best way to go forward to accomplish the business goals. Smart consulting practices embrace openness and flexibility, and have the expertise to lay out a clear joint project roadmap.

TECHNOLOGY

Like any other emergency situation, it helps to have a deep inventory of tools and frameworks that can be quickly brought to a software project rescue operation. Table 1 gives an example list of Tools and Frameworks for different applications.

SAMPLE LIST OF TOOLS/FRAMEWORKS FOR A FEW APPLICATIONS

	Business Intelligence & Data Warehousing	Collaboration and Portals	Content Management	Document Management	Capital Asset Projects	Mobility	Project Management	Regulatory Compliance	Software Application Components	Sourcing and Procurement
SOLUTION COMPONENTS	Data Analysis	Activity Management	Author Templates	Automatic Document Generation from Templates	Asset Management	Pocket PC Applications	User Calendaring	Artifact Management (Sarbanes-Oxley)	**User Interface**	Transaction Document Management
	Data Consolidation	Announcements	MS Office Connector	Automatic Naming and Versioning	Asset search	.NET Compact Framework	Change Management	Artifact Review Process Management (Sarbanes-Oxley)	Calendar Control	Document Translation Management
	Data Correction	Articles	Revision Management	Check in - Check out Feature	Asset Servicing	Sycnhronization Component	Client Management	Defect Tracking System (Sarbanes-Oxley)	Data Grid Control	Document Workflow Rules Engine
	Data Enhancement	User Calendaring	Publishing Activity Scheduling	Document Control	Asset Tickets and Followups		Cost Management and Estimation	Encryption Algorithms (GLBA, FFIEC)	DHTML Menu Builder	Document Transport and Routing Services
	Data Matching	Contact Management	Task Management	Template Management	Asset Tracking and Monitoring		Discussion Forum	OSHA Compliance Management	DHTML Pop Up Control	Application Integration Adapter Development
	Data Measurement	Customizable Multilevel Menu	Template Management	Document Repository	Change Management		Email Notification	HAZMAT Inventory Management System	Graph Generator	Transaction Analysis
	Data Parsing	Discussion Forum	User Management	Document Transportation	Client Management		File attachments and Uploading	Workflow Engine	Tab Control	User-Role Management
	Data Standardization	FAQ Module	Web Authoring and Publishing	Inline Document Generation	Cost Management and Estimation		Issue Management	Process and Workflow Configuration		
	Reporting Services	Google Search Plugin	Work Flow Management	Work Flow Management	Discussion Forum		Project Statistics and Reporting	User-Role Management	**Application and Data Services**	
	Data Extraction, Transformation and Loading services	HTML Document Management	User-Role Management	User-Role Management	Email Notification		Project User Management		Application Adapter Frameworks	
	Data Explosion	Organizational Hierarchy			File attachments and Uploading		Risk Management		Authentication Services	
	Caching	Product Management			Issue Management		Scheduling		Crystal Engine Component	
	Metadata Management	Surveys			Risk Management		Task Assignment		Data Import Control	
		Task Management			Scheduling		Timesheet and Time Tracking		Engine-Collection Pattern	
		User and Role Management			Task Assignment		User Hierarchy and Role assignment		Entity Meta-Data Management Framework	
		WYSIWYG Editor			Timesheet and Time Tracking		User Preferences		Exception Handling Framework	
					User Hierarchy and Role assignment				Logging Framework	
					User Preferences				Notification Engine	
									Synchronization Engine	
									Parallel Query Pattern	
									Security Framework	
									Single Sign-On	
									Smart Object Generator	
									Synchronization Engine	

Table 1

CREATING FLAMEPROOF SOFTWARE PROJECTS

To create flameproof software projects a hierarchy of conditions has to be met. The initial conditions are hygiene factors, without which projects are doomed. And their presence only assures, at best, meeting average expectations.

1. **Focus, Focus, Focus:** Vendor's focus on a client's industry domain is a basic hygiene factor. However, to add that extra flameproof coat the vendor should have the ability to astutely apply technologies that are aligned with the client's business goals.
2. **Knowledge:** Surprising as it may sound, knowledge of technology can be a hygiene factor but a vendor's ability to bring the advantages of the economies of knowledge may provide another flameproof coat. The depth of this coat can be seen in the inventory of tools & frameworks that a vendor possesses and can bring to the party. Knowledge in the client's organization should also be reviewed. Sometimes, a lack of knowledge in a client's organization makes a project inflammable, right at the outset.
3. **Project and other processes:** Many firms profess to possess this hygiene factor. However, flameproof projects have malleable project processes that fit industry and client processes.
4. **Excellent Communications**: Now, we are beginning to get into an area that truly assures flameproofing. With excellent communications, there is transparency of client objectives with the rank and file of the vendor firm, wherever they may be. There is little or no slip between the cup and the lip. Clients and vendors work in a spirit of open cooperation, on the same side of the table, focused on achieving project objectives - their mutual goals.
5. **Passion/Attitude of People:** The ultimate flameproof coat that can be applied, making all the difference, is the passion and attitude of the people involved in the project. A "get-it-right-the-first-time" Project "A team", bound together by common goals, malleable processes and clear communications will consistently deliver flameproof projects.

A CASE IN POINT: A SOFTWARE DEVELOPMENT PROJECT CATCHES FIRE

In early 2003 a software product vendor—let's call it ABC Company—marketed a product that had a Windows-based client-server architecture with a rich set of features and functionality.

The company's competitors—Microsoft among them—sold a similar product innovated with a new generation of web architectures. The web-based software caught the market's fancy and sales took off. With its older client-server technology, ABC Company was under pressure to upgrade its product.

Its management was eager to re-launch its product with .NET standards based web architecture. However, in trying to match innovations in the marketplace, web architecture alone was not sufficient to differentiate the new product.

ABC Company noted a weakness in other web-based products—they scored low on usability and user experience. Customers complained of an excessive number of screen views and clicks required to get most tasks done. ABC Company sought to exploit the problem and differentiate its product by providing a higher quality user interface—one that combined the user efficiency of a desktop with the technical advantages of web architecture.

ABC Company selected a CMM Level 5 rated software service provider to help develop the new product. The service provider, which claimed to have 1,000 projects under its belt, thought it could easily handle the project with its cut-and-dry, low risk "Application Migration and Reengineering" methodology. The firm used a framework to reverse engineer the existing database. Migration of product functionality was scoped on a "one-to-one basis" to replicate client server functionality.

When the development reached testing stage, the project was in deep trouble. Application performance appeared extremely poor, with each screen taking several seconds to pull up. The design did not factor in the large amount of information that was being pulled out of the database and presented on the screens—the functional improvisation required while transitioning from client-server to the web.

ABC Company was fast approaching their market launch date and a crucial sales opportunity milestone for the new product. Efforts to rectify the situation were frustrated by the bureaucracy in the service provider's organization and the communication overheads imposed by process rigidities. A major share of the project budget was already exhausted. As for the product, it functioned, but very poorly and certainly nowhere near expectations. ABC Company had a classic "flaming project" on its hands.

THE SILICUS APPROACH: RESCUING ABC COMPANY FROM A FLAMING PROJECT

ABC Company turned to Silicus for help. We recognized immediately the issues facing the client's product management team and the challenges of operating in a highly competitive market segment. The top priorities were a) product performance and b) fast time to market. We took a number of steps to turn a desperate situation around:

- Senior management at Silicus communicated the client's business priorities and project issues to team members across the US and India offices and prepared them for a "fast-track" project.
- High level requirements were gathered and documented, and the existing design of the problem application was reviewed to understand its problems.
- Major design changes were recommended, with formal design documentation presented to the client for review and approval.
- To rescue the project, immediate focus was placed on the core product, with peripheral product features grouped into a later phase of development.
- Detailed requirements were captured through consulting sessions and application analysis, and documented and approved by the client.
- Engineering decisions were made regarding database redesign, reengineering of the code that accessed the database, and reengineering of the code that determined the efficiency of information transported to and presented on-screen in the browser. Effectively, beneath the UI skin, this meant redesign and development of the entire application.
- Rapid and iterative development, build, and testing cycles were adopted to verify that design decisions had a positive impact on product performance. Analysis of the performance was documented and presented to the client for confirmation.
- As product delivery approached, the development team followed 24x7 development and testing schedules, distributed across the Houston and India development centers. The client's personnel were jointly involved in the process to provide immediate review and feedback.
- During testing, all team members accessed a common issue-tracking system to log issues and track issue status.
- The entire effort was scheduled across six weeks with a cohesive team of 10 people. Each team member put in significantly more time than a standard eight hour workday.

Silicus, in partnership with its client, put out the project fire. The final product was delivered on time, within the client's original budget, and with market leading performance.

About the Authors:

Pradeep Anand (pradeep@seeta.com) is the President of Seeta Resources, LLC. Mohit Mahendra (mohit.mahendra@silicus.com) is a Principal of Silicus Technologies, Inc. Both firms are based in Houston, Texas, USA.