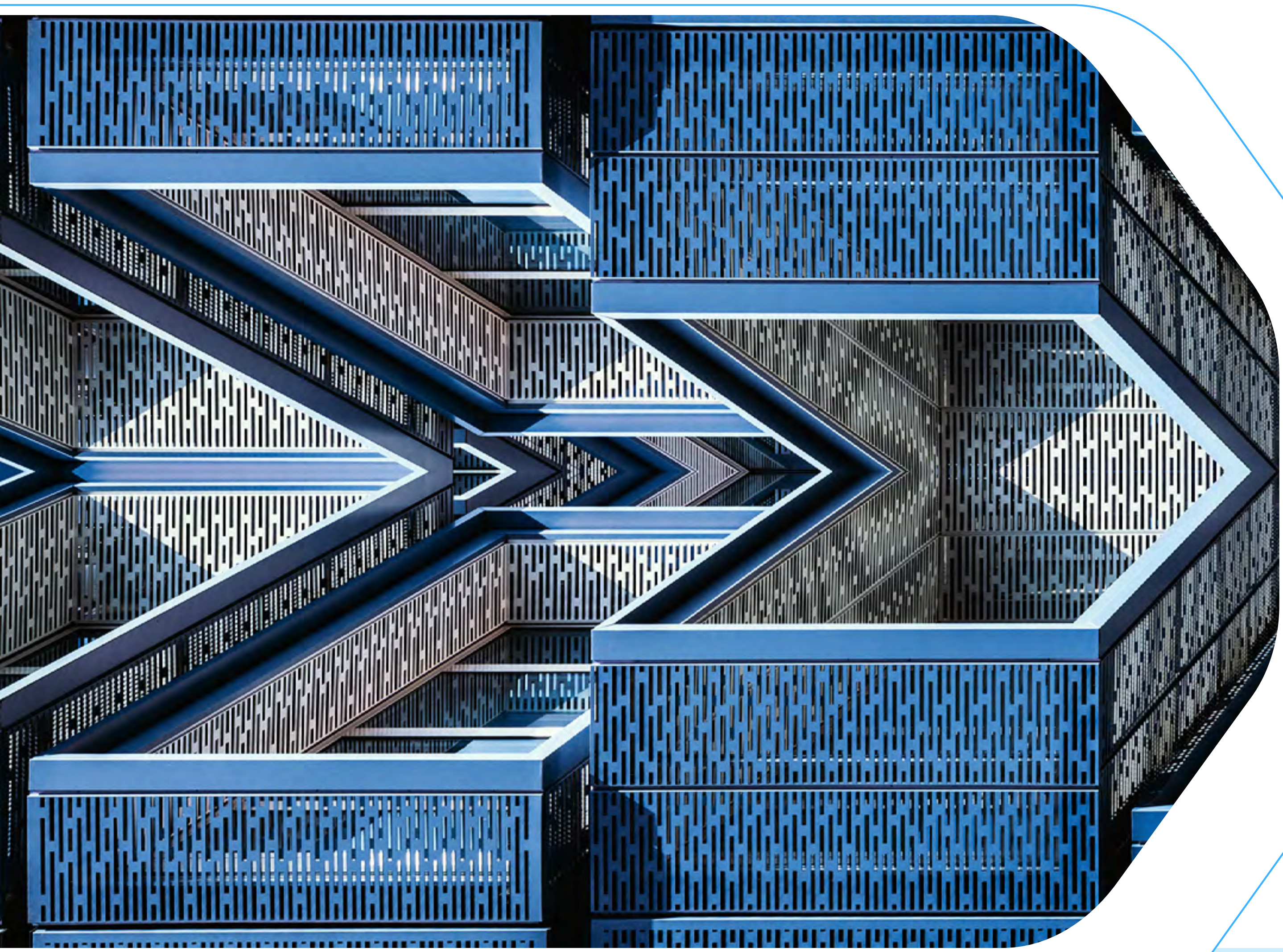
WHITE PAPER

Agiloft



How to Ensure the Success of IT Projects

A collation of recommendations from members of the CIO Forum

Contributors

A. J. Peter Wall, Business Intelligence/Information Management Professional
Alexander Lipanov, Director at Software Outsourcing Company Infostroy Ltd
Anand Subramanian, IT professional in Logistics & Supply Chain
Archie Clark, Project Manager
Bronnie Brooks, Information Technology and Services Professional
Carlos Garriga, Experienced IT Executive
Chet Ung, International IT Auditor
Colin Earl, CEO Agiloft
Col. Shankar Gurkha, CIO at GIPCL
Costas Skouras, Head of Information at The Audit Commission
Craig Elsdon-Dew, Consultant at Ericsson
Craig Humphreys, Senior Commercially Oriented IT Executive
Curtis Christensen, Information Technology Professional
Dave Perfetti, CIO at Royal College of Physicians and Surgeons of Canada
David Elkin, CIO, Queensland Building Services Authority
David LeBlanc, CIO,VP,PMP
David Paschane, Applied Scientist/Human Geographer
Dinesh Chandna, Senior IT Professional (CIO/CTO)
Diptesh SinghLead Consultant
Donny Buckman, Technology Supervisor at Greenfield Union School District
Dr. Paul Coleman, Executive Leadership
Ed Sweeney, Manager of Information Technology at FCI
Gary Breaux, CIO - IT Director - Project Manager
Gary Parkinson, IT Director at Isaac Agnew Holdings Ltd
Gaurav Garg, IT Director
Gene Doody, Director, Network Operations & Quality Assurance at AFBA
George Philip, Project Manager at Global Infonet Inc
Ghazi Qarout, SVP, Head of I.T at Al Hilal Bank
Hari Natarajan, Associate VP of IT at Kotak Mahindra (UK) Ltd
Ian Clayton, Outside-In Thinker
Ian Rogers, International multi-channel retail executive consulting
Jeff Clark, Senior IT Portfolio Manager
J. Kyle Howard, I.T. Manager at Perot Systems Corp
Kent Li, CIO at UniTrust Finance & Leasing Corporation
Kevin J Poulton MBA, Managing Director, 1 B4 All
Lasantha Fonseka, MBA, Director of Information Technology at Tomarco
Linda Williams, Senior IT Manager/Consultant
Mark Erickson, Senior IT Executive
Martin Neville, Head of Service Management
Miguel Rodriguez, Chief Information Officer at PayCargo LLC
Mike McMahon, Director at I M Consulting Pty Ltd
Mik Hartel, CIO/VP of IT; COO/ VP of Operations
Mohammed Abu Hadhoud, Co-Founder/Chief Technology Officer, PMP®, MCPD, MCSD
Mubbisher Ahmed, CIO
Nikhil Datar, BI Business thinker
Nkosinathi Mvelase, CIO at South African Bureau of Standards
Oya Sanli, Information Systems Manager
Paul Irvine, Founder, pCapacity LLC

Contributors Continued

Paul Pal, Project & Program Manager
Paul Pauesick, Director of It at Kansas City Board of Public Utilities
Peter Scheyen, CTO at Richard Ivey School of Business
Philip Wang, Senior IT Manager at IS Department Ltd.
Pradeep Nair, Manager - Systems at Bluedart Express Ltd
Prakash Pant, Technology & BPM Consultant
Rachel Sun, PMP Director, US & European Business at Neusoft
Raghu Kastury, Program Management Officer
Ragib Hussain, Vice President Strategy at e.Soft Technologies
Randy James, Managing Director at The Consultants Source
Richmond Mace, Executive Director - Technology Services at Musanada
Robert Herron, Sr. Vice President at VDP LLC
Russell Price, IT Director/Senior Executive
Saji PK, Sr VP at Sify Technologies Ltd
Sandeep Raut, PMP Senior Project Manager - SAS India
Sandeep Singh, Technical Architect at Target
Scott Payton, Director, Professional Services
Shlomo Covrigaru, IT Guru & Business Consultant
Shrinivas Deo, Chief Technology Officer
Spencer Wasley, Momentum Coaching
Sreenivas Chaparala, Technology Leadership at PCHC, MD
Stephen Warner, Senior Director, Global Operations at Reed Elsevier
Steve Hawthorne, IT Executive
Steven van 't Veld, Principal Information Architect, CEO/Owner A/I/M
Steven W. Smith, Senior IT Executive
Suzanne Niedzielska, IT Director at State of Connecticut
Swapna Gupta, Director of Technology at VertMarkets
Teresa Cashen, Project Manager at American Express
Thomas Struan, Principal at Thomas Struan Consulting
TJ Coakley, Senior IT Leader
Toan Do, IT manager
Tristan Chiu, Solution Architect at EnergyAustralia
Utpala Joshi, Internal audit controller at Zensar Technologies
Venkatesh Sanklapur, IT Program Manager
William Goedicke, Senior IT Director

Table of Contents

5 Introduction
- 5 Goal
- 5 The Meaning of Success
- 5 Project Methodology and Specifications Terminology

6 Recommendations
- 6 Requirements Specification
- 6 Engineering Specification
- 7 Software Architecture and Hidden Costs
- 7 Effective Communication
- 7 Effective Teamwork
- 8 Project Management Planning and Time Use
- 8 Early Demos
- 9 Effective Training
- 9 Top Management Commitment
- 9 Use Efficient Tools and Do Not Reinvent the Wheel
- 10 QA Process
- 10 Risk Management
- 10 Integrated Change Control
- 11 Effective Project Management
- 11 Team Resourced with the Right Expertise
- 11 Transparent Workflow
- 11 Infrastructure
- 12 Custom Contracts
- 13 Supplier Management
- 15 Leadership Skills

15 About Agiloft, Inc.

Introduction

Goal

The Healthcare.gov fiasco served as a prime, and very public, example of a failed IT project. The causes are numerous: lack of early end user testing, botched custom code, poor project management, the use of inefficient technology, inadequate capacity assessment – the list goes on and on.

Certainly the development and implementation of Healthcare.gov was no easy task, but it was not doomed to failure, as some would have us believe. With a disciplined approach and adherence to some vital, but often overlooked, tenets of effective IT management, Healthcare.gov could have been a success.

The goal of this white paper is to improve the success rates of IT projects and, hopefully, prevent another disaster on the level of Healthcare.gov. It is based on the collated contributions from over a hundred senior IT executives to the thread "How to ensure the success of IT projects" in the CIO Forum and my own experiences.

To achieve this objective, the paper has to offer precise, *actionable* suggestions with guidance as to how and why they should be applied. But, first let's clarify exactly what we mean by "IT success."

The Meaning of Success

There is a noticeable difference between the 75% to 90+% success rate claimed by IT managers for their own projects, and reports from industry observers that less than half are successful. This does not reflect a difference in the sample data, but rather a difference in the definition of "success," so it is important that we understand this difference.

If success includes all software projects that are actually delivered by IT to the business, the success rate may well be higher than 75%. However, this includes projects that exceeded the projected budget or time-frame, were not accepted by the users, or failed to meet the business objectives that allowed them to be funded in the first place.

If we define successful projects as those that "go live within the projected budget and timeframe, meet the stated business objectives, and continue to function throughout their planned lifespans without major hiccups or incurring significant, unplanned costs," the success ratio is closer to 25%.

CEO's notions of success are based on the second metric, so this is the one that matters.

Project Methodology and Specifications Terminology

There are many different approaches toward IT development: Waterfall, Agile, RAD, Cleanroom, DSDM, Iterative, etc. Each methodology has committed, even fanatical adherents, yet the presence of all these competing approaches illustrates that this is a tough problem for which there is no onesize-fits-all solution. This paper tries to avoid religious wars by providing advice that is applicable to any methodology.

Each approach includes some kind of specification and though the terminology and emphasis may vary, they all include some form of the following:

- A Requirements Specification that describes what the project will do.
- An Engineering Specification that describes how it will do it.

To draw an analogy, the requirements specification describes the need for a 4 door passenger sedan that can cruise at 75 mph. The engineering specification includes details such as whether it will be a red Toyota Camry or blue Ford Taurus, any stereo equipment that will be included and when it will be delivered.

The above example illustrates why it is not possible to develop a project from a requirements specification alone. The team may be perfectly capable of building either a Camry or a Taurus, but it has to make the decisions necessary to actually build one or the demonstrate how they would configure the system to address them in real time. This helps you understand the solution's true capabilities and responsiveness.

other. These decisions are what distinguishes the Requirements Specification from Engineering Specification.

Project milestones, testing requirements, code architecture (if custom coding is involved), maintenance, etc. can all be included in sections of the Engineering specification, though different methodologies might prefer to call it something else.

Recommendations

Requirements Specification

The Requirements specification provides a clear definition of the project scope and the foundation from which the Engineering specification and hence project milestones, dependencies, and other documents are derived.

It also provides the definition of success for the project, expressed in business terms, not IT terms. It must be expressed in terms of the business stakeholders and this illustrates why stakeholder identification is key – a missed stakeholder can kill the project.

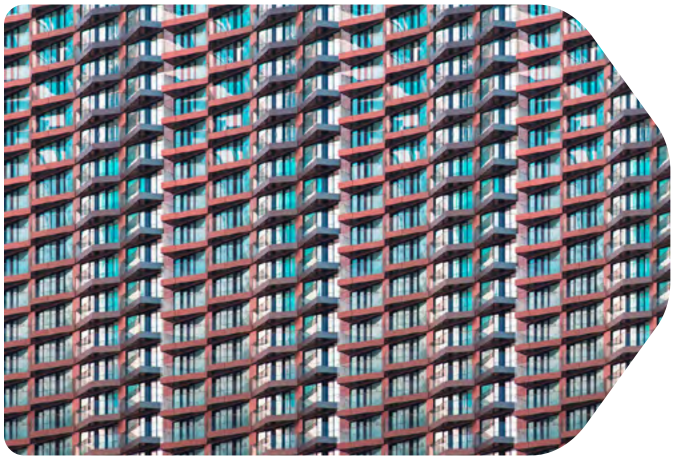
For a project to be successful it must address a particular "pain" for each affected stakeholder in a business process. By understanding stakeholder pains, an experienced project leader should be in a position to present what the project is hoping to achieve from each viewpoint and thus reach a common understanding which will eventually lead to everyone pulling in the same direction.

One critical issue that must be addressed in the Requirements Specification is what the system will be like to use. End users' satisfaction is critical, yet it's common for leadership to be insulated from the concerns of the end users because people are afraid to complain to the big boss.

Identify influential people in the organization and study if the project objectives and the individual power centers are in line. If they are not, develop a strategy to bring them in line.

Engineering Specification

The Engineering specification must be complete, unambiguous and carefully reviewed. One sure



way to get a project into trouble is to give the Requirements specification to Engineering and leave it up to the "common sense" of programmers to decide how it should be interpreted and implemented.

The worst kind of an Engineering specification is one that only exists in the mind of a coder. It often contains truly bizarre interpretations of the Requirements specification and unwarranted assumptions that are only discovered when, after countless man-hours, the project is delivered to the customer, looking like some strange, misshapen sea creature dragged from the depths of the developer's imagination.

The Engineering specification is often a Word document, but of course this is not necessary, or even optimal in all cases. Any format that defines exactly what will be delivered can fit the bill. For example, it might be a working prototype, *coupled with a spreadsheet which describes all the changes* that must be made before it can be put into production.

The prototype approach even has certain advantages, such as support for early user testing. However it does carry the danger that naïve staff might confuse it with the actual system. If they like it, this can cause them to be too optimistic by imagining that the project is further along than it really is. If they dislike it, it can cause them to dismiss it prematurely. The key point is to emphasize that the prototype coupled with the set of required changes is the spec, and these changes can include items such as "rewrite from scratch".

Software Architecture and Hidden Costs

When custom coding is involved, the Engineering specification needs an Architecture section which describes exactly how the requirements are going to be implemented. Without such a section, the project may run into severe problems with scalability, maintainability and even security, just as it seemed to be nearing completion.

Even with such a section, custom code introduces a significant level of risk and uncertainty, so it should be avoided where possible. There are several reasons for this. While it is widely understood that code contains bugs that take time and effort to flush out during development, this is just the tip of the iceberg. Over the life span of a large project, the typical ratio of costs for bugfixing/maintenance compared to initial development is 4 to 1.

Custom code also introduces uncertainty. Despite the best efforts of very smart people with various development methodologies and IDE's, the development and QA time is very unpredictable. No one can really say "this code is bug free", all they can say is "we have not found any bugs in recent tests". So when custom programming is required, set expectations that the project may take significantly longer than expected and budget for ongoing maintenance costs.

Effective Communication

Schedule a meeting to discuss the project goals as laid out in the requirements specifications and confirm that the project is headed in the right direction before allocating the resources to get it there.

Have brief (5-15 minute) daily status meetings and longer weekly meetings to ensure that everyone is up to date with progress and to give the business managers the opportunity to suggest course corrections if the business needs have changed.

Keep stakeholders up to date with project management software that automatically sends periodic updates and escalates when a dependency is late. If a team member leaves, the project management software may automatically assign their issues to another manager.

Maintain upward, downward and lateral communication and define preferred communication tools for each stakeholder. Workplace communication is not a case where one size fits all.

Engage people both inside and outside the project and ignite the human spirit. Find people willing to deliver on what they commit to, and allow them the space to contribute and make a difference.

Capture all communications at all stages of the project, including relevant e-mails. These will help you when you create the final project handover documentation for the client.

Till the end of the project delivery, you will face obstacles and in some cases pressure from the senior management. This is where assertive communication skills and clear change management process are needed.

Effective Teamwork

Closely coupled with effective communication is effective teamwork, and by far the most critical aspect of effective teamwork is simply that each member of the team keeps to their commitments.

They do not need to hang out together after work, or even like one another, though it certainly helps. But they do need to act as business professionals who can be relied upon. As a project leader, it is your responsibility to establish such a culture and be willing to hold people's feet to the fire when necessary.

For example, a frequent cause of project slippages is team members who do not complete their assignments, such as making some necessary decisions, on schedule. They then turn up to meetings expecting that everyone else will be happy to wait while they hash them out during the course of the meeting.

It is tempting to accept such a "brief delay", but this will establish a culture where people treat commitments as little more than "good intentions about what I hope to do if everything else in my life goes smoothly". The result will be further missed commitments and slippages, justifiably culminating in your own termination slip. So consider responding by saying "Since XXX has not completed their assignment, I am rescheduling this meeting for tomorrow so that he has more time to work on it. I will notify the CEO that the project has just slipped by a day and explain why". XXX is not going to regard you with great fondness, but everyone will get the message and take their commitments seriously going forward. As Sun Tzu taught, effective leaders are both loved and feared. If you need pure love, get a dog. *Note: Several contributors regarded this recommendation as far too extreme and said that it only be used when lower levels of escalation had failed. The best approach probably depends on company culture and the politics of the situation.*

Conversely, when people do meet their commitments, or go above and beyond the call of duty, give them praise and visibility. A team leader who credits others for the project's success and does not attempt to take the glory for himself, will foster a much stronger team spirit and garner deeper respect than the individual who sings his own praises.

Project Management Planning and Time Use

Problems and unexpected events will always exist in a project, no matter how big or small. It's how the Project Manager deals with these elements that make the difference between success and disaster. You need someone tenacious and fearless.

A good rule of thumb is to plan the resource and time requirements, in as much detail as you can. Then double that estimate —it always takes longer as things always happen that you haven't planned for.

Contingency planning! When you create your project plan you will of course add contingency time for the "unknowns" that you may discover along the way. Build additional time into each task and, most importantly, stress the importance of delivering on time and on-budget in the project definition kick off meeting.

Motivate your team by talking about the positive visibility they will gain from delivering on time.

Engage the team by asking for their commitment to communicate any delays they discover to the project manager right away. Go around the table and ask each person, "Do you agree with the project plan and how we need to manage it?" and "Do you have ideas/ suggestions for a better way to manage the project?" Write these two questions on the white-board and as you ask each team member, add their initials if they agree. This visual aid such as the white board or the poster paper will be remembered most. Then follow up with the meeting minutes. Make those brief but get the two questions in there and the fact that the entire team agreed to this approach. Hold people accountable. Make sure you fully understand any delays they mention to you and document those by adding a note to the task in the project plan.

Early Demos

A system cannot be fully demonstrated until it has been developed, but this is no excuse for not providing some kind of prototype. A software mockup without a working back-end can provide a reasonable facsimile to the planned system at a cost that is trivial compared to the actual system, and should be included as an early milestone. Even a stack of papers with hand-drawn images may suffice, and is infinitely better than nothing.

An example illustrates what can happen if the end user is not considered well in advance. A hospital in the Los Angeles area implemented an Electronic Medical Record system, spending well over $10M and countless man hours. A large, well-reputed vendor delivered everything on time, on budget, and in full compliance with the spec. However, the clinicians did not like the system and refused to use it. So the system was forced to scrap it and write off the entire investment. From the perspective of the hospital administration, this was an IT failure.

Effective Training

Train the users in a test environment before the system goes live. Gather feedback during this training and plan on modifying the system based on their input.

Top Management Commitment

All the contributors to this paper agreed that top management commitment is important. However, gaining this commitment is not under the control of the IT department, so this recommendation is not actionable. Further, the contributors did not detail why such commitment is important.

For example, is top management commitment important because:

1. They will throw additional resources at the project even if it is behind schedule and costing more than expected? *This sounds like top management using their muscle to claim that a project is a success, even when it should be judged a failure.*

2. Otherwise, they may cancel the project. Even if it is on time, on budget and has a clear ROI? *It would be interesting to hear of such examples.*

3. They will force people to use the result, even if the staff hates it? *This also sounds like top management forcing a project to be considered a success.*

There are other questions: If top management is not fully committed to a project, how can IT obtain such commitment? If IT goes to top management and asks for their total commitment on every project, it may eventually be perceived as "crying wolf." How should we even measure management commitment?

In brief, obtaining top management commitment is very desirable, but not necessarily actionable and cannot be a substitute for disciplined project management.

Use Efficient Tools and Do Not Reinvent the Wheel

It is a lot easier to finish a race in 5 minutes if you start 10 yards from the finish line than if you start 10 miles back. Enterprise IT projects generally require a web interface, auditability, dashboards, automated backups, security, web services/REST APIs, reporting, searching, synchronization with other systems, data integrity constraints, export/import capabilities and more. The more core facilities are provided by the chosen development platform, the less needs to be developed.

Everyone is competing for the IT dollar and business managers will go outside the company if they cannot get the level of responsiveness that they need internally.

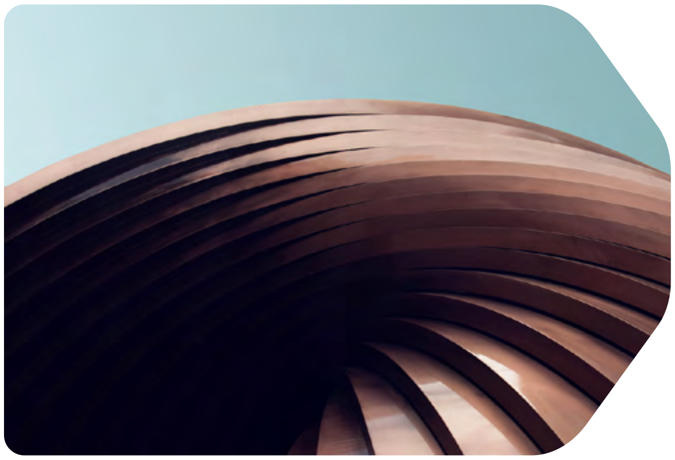
Example: When a business manager asked his IT department for a quote on developing a tool for managing COCOM compliance, he was told that it would take about six months effort using Lotus Notes. An outside vendor (Full disclosure: The vendor in this example was Agiloft) quoted him two weeks as a fixed price contract, got the job and delivered on that promise. From the perspective of the business manager, IT failed before they even started.

Yet the IT department could have installed the vendor's software on their own server and configured it in exactly the same way. Apart from needing a week of training to come up to speed on the technology, they would have been in exactly the same starting position and when IT delivers a finished product in less than a month, they look like heroes. This example also illustrates why an effective project manager must be willing to accept some level of risk. It may seem that choosing a new tool is riskier than a well-known solution with in-house expertise, but a refusal to move with the times can actually guarantee failure.

Another example: At one company a development manager was planning, and had actually begun writing, a comprehensive document on the coding standards to use for a major J2EE development project. 20 minutes research on Amazon brought an entire book on coding standards and avoided a lot of argument. Incidentally, the critical tool in this case was not so much the availability of the book, but Amazon's rating system that made it easy to find the best book on the topic and so avoid the religious arguments that usually accompany the choice of coding standards.

QA Process

A good QA process is critical. Ensuring a level of quality is in place reduces negative perceptions from the end users of failed projects. The QA team must ensure that requirements and functionality are solid before end users get a hold of the product. It only takes one negative view to spread the perception that the project is a failure.



Risk Management

Prepare for the worst. Hardware will fail, users will demand changes, and staff may be unable to meet their commitments.

Be willing to take calculated risks; change almost always requires some risk but there is no progress without it and stasis absolutely guarantees long-term failure.

Be transparent and open regarding all project setbacks and escalate in good time. Bad news can be dealt with if it is early. Bad news late is a disaster!

Gather sufficient information about past projects in the organization, their success stories and disaster stories. Analyze reasons for success and failure and make use of your assessment in the current project.

Obtain estimates for remaining work on a task in workdays rather than % complete.

Have a Plan B in place before you need it, not when disaster strikes and people start slipping into panic mode.

Integrated Change Control

True success includes adoption of change in the trenches. Make sure that your requirements translate into changes that are actually achieved and welcomed.

Have a well-documented change management process and get it signed off from the client; otherwise clients will believe that it is their right to change the requirements whenever they want without understanding the impact. In particular, business users need to appreciate the financial impact of changes.

A change management system should capture the impact of each change on the project schedule and cost, who requested it and who approved. But it does not need to be complex. The ideal might be a web based system with automated workflows and LDAP integration, but even a simple Word document is infinitely better than nothing.

The key point is that if the business manager complains that the project is $80K over budget and 3 months late, you must be able to respond "Actually, it is $15K under budget and 2 weeks early. The post-spec changes that you requested and whose cost you approved, added $95K and 16 weeks to the delivery time. Here is a report detailing each of them". Of course, one of the reasons for doubling the initial estimate of the project cost was to avoid such unpleasant conversations and if you failed to do so, that is your fault. Surely you know by now that users almost always change their minds!

Making prudent use of document and code repositories for collaboration and unification across development teams is a great help for successful delivery of projects. By using a knowledge base and search tools, the iterative process being done by collaborative teams can be managed throughout the project life-cycle.

Some application development environments come with these tools built in. Training and full use of such programs will certainly keep the scope, design, and development in check as everything is run through the disparate teams on the way to implementation and delivery. Many more eyes are able to quickly examine the routes developers are taking at any time, which will give the opportunity to head off potential scope creep and re-coding of available classes and modules.

Effective Project Management

Everyone agrees that an effective project manager is critical and this paper aims to provide guidance for the project manager. The following suggestions are intended to help you hire an effective project manager or indeed other IT employees, but first lets clear up a couple of myths:

Myth 1: Interviews are Effective

The exact number depends on the interviewer, but researchers have found that typical interview accuracy rates are not much better random selection. Further, there is almost no correlation between how good someone thinks they are at interviewing and how effective they are. Well, actually there is a correlation and it is negative. The people who are most confident of their ability to hire based on an interview are precisely those whose judgment was found to be the worst when based on objective criteria. If you have never hired a dud employee, congratulations! If you have, read on.

Myth 2: Checking references improves outcomes

Not only are many employers too afraid of lawsuits to give bad references, some will even give excellent references to help rid themselves of problem employees. In general the only thing harder than finding a good employee is finding a bad reference.

Prior experience is an excellent indicator of how much money employees will want, but a poor indicator of how well they will actually perform.

A More Efficient Process

Of course, interviews and reference and background checks are an essential part of the hiring process, but they should not be the sole criteria. Here is a suggested process:

Create custom web forms for each job opening and set up business rules to prioritize and/or autoreject candidates based upon information provided in the form. For example, ask them to take an online test and include the result in their application.

Appropriate rules can eliminate 80% of the candidates without any effort on your part.

Call the remaining candidates and ask them to take an online aptitude test for that specific position Depending on the position, only progress the top 10-20% to the time-consuming phase of an inperson interview and another test under controlled conditions.

Only hire candidates who achieve unanimous buy-in from team members. Immediately make a job offer, contingent upon reference and background checks. Do not keep them waiting or another employer may step in.

A disciplined, automated hiring process can not only free up your time and ensure you find the best possible candidate, it is objective and can be audited to demonstrate an absence of bias.

Team Resourced with the Right Expertise

This might seem obvious, but we have all seen projects flounder simply because the participants did not have the necessary expertise. The Milestones section of the Engineering Specification should include a description of the required expertise to meet each milestone and which team members will provide it.

Transparent Workflow

The workflow tool should make it clear what the workflow is for any given process and how this process has been followed in any particular instance. Unless the users are also programmers, this implies that the workflow representation and its application must be accessible to non-coders.

Infrastructure

Many IT professionals who are software focused don’t include hardware in their planning; so many projects that do a great job at meeting the functional requirements fail during production use.

A very high majority of small and mid-size clients’ infrastructures are in bad shape even before new loads are introduced, so the results are predictable. Larger clients also suffer this same fate, but they sometimes are better prepared, or at least they say they are.

Plan for at least four times the "expected peak" load and assume that any given piece of hardware, including motherboards and RAID cards, will fail.

An assessment of the state of the starting (i.e. current) infrastructure should occur at the start of a major software project. As many software professionals do not have strong infrastructure experience, it is wise to hire a consulting firm or use an IT Infrastructure Assessment tool to conduct an objective survey of the reliability, utilization, performance, capacity, and design of the major components.

A key outcome is to identify the risk of the current infrastructure and its current weaknesses, and provide data to forecast the loading of the components when the additional application(s) are fully deployed.

Experience demonstrates that lack of capacity or poor infrastructure design leads to high latency, application timeouts, and random errors due to high loads that are nearly impossible to troubleshoot, yet consume a large amount of software developer resources.

Capacity planning is a key process that many small and medium size organizations do not perform on a regular basis. A capacity plan should include: Servers, Web Servers, Storage Devices, Networks, Desktops, Printing, Database, Telephony and Data Center Resources.

Custom Contracts

Like the section on "Top Management Commitment", it is not clear whether this section should be included. We decided to include it on the grounds that it contains some valid considerations and can always be ignored. However, it is worth noting the objections:

> *Some contributors said that it fails the "actionable" standard because the project manager often lacks the authority to decide what legal contracts can be accepted. Others argued that it is outside the scope of the paper because the project "does not begin until the paperwork is signed".*
>
> *Contributors also disagreed over the term "neutral", with some arguing that a "neutral contract" implied that the vendor should share the risk of cost overruns or project failure, while others arguing that this was unrealistic since almost all contracts were T&M based.*
>
> *They also disagreed about the budget at which custom contracts were justified. We tried to present the overall consensus, but you may well disagree.*

For engagements with budgets less than $75K, it is seldom worth the overhead of negotiating a custom contract if the vendor's standard contract is reasonably compliant with industry norms. In daily life, people make purchases such as a Tesla car for daily life, people make purchases such as a Tesla car for similar amounts of money, without expecting to negotiate custom terms. Consider what Tesla would have to charge if every purchase involved legal negotiations.

For budgets in the $75K – $250K range, the overhead becomes increasingly justified.

For budgets greater than $250K, it is usually worth negotiating a custom contract, but be aware of the drawbacks:

- Large vendors and those whose goods or services are in high demand, may simply refuse to engage, leaving you to deal with struggling and uncompetitive suppliers.
- With vendors who do agree to negotiate, the effort can easily add 3+ months to the project, and you may fail to reach agreement.

- You will be starting the engagement with an adversarial relationship established by the bickering between your respective lawyers.

Compare these downsides with the percentage of engagements that were actually helped by custom terms in the contract language and the resulting impact. Do not expect your lawyers to perform this kind of cost-benefit analysis, or even consider business implications. They are lawyers, not business people.

The advantage of a custom contract is the inclusion of terms that are favorable to you as a customer, multiplied by the probability that those terms will ever become active and their impact if they do. If the vendor's default contract is heavily weighted in their favor, this upside can be very significant. If the default contract is neutral, the upside of changing it is correspondingly limited.

The first step should therefore be to review the vendor's contract and determine if it is reasonably balanced and compliant with industry standards.

- If the standard contract is reasonably balanced, then be willing to accept it. But try to negotiate even better terms, especially if the engagement is large.
- If the standard contract is imbalanced towards the vendor, then negotiate and be willing to walk away.
- If the standard contract is extremely imbalanced, then negotiate or simply walk away immediately - you need to ask yourself whether you really want to do business with a company whose default terms are so unreasonable.

Naturally, you should make sure that any standard contracts that you are want vendors to accept, are not extremely imbalanced. Your lawyers may think that an imbalanced contract is in your favor, but all except the most overpriced or desperate vendors will simply refuse to do business with you.

Supplier Management

Be tough with suppliers. For sizeable engagements, build performance clauses into the contract, including the ability to walk away without paying if they are significantly late, and do not count on delivery promises until they are actually met.

In order to avoid re-inventing the wheel, most IT projects will be built upon third party software. Here is a checklist of items that may be relevant when evaluating which third party framework to use. Naturally, not all items are relevant to all projects.

Auditability

The system must be auditable in multiple senses to ensure compliance with Sarbanes-Oxley and other regulations. It must make it easy to show an auditor what a defined business process is, how the system enforces the process, and how the process has been followed in any particular instance. Further, the solution should make it possible to capture and collate data, such as who logged in, what IP address they came from, and what records they viewed, edited, etc.

Integration

The solution should include prebuilt integration with standard technologies, such as LDAP/Active Directory, Google Docs, MS Office and Single Sign-On. It should also provide a robust set of APIs and scripting options, including Web Services. Ideally, even the source code should be accessible – not that you'd want to change it any more than you'd want to use an emergency parachute, but it is nice to have the option.

Adaptability

Once the system has proven itself in the initial deployment, it should be easily extensible to other business areas. So the data models, business rules, workflows, access permissions, and data input forms must be fully and rapidly customizable.

Scalability

The solution must scale to support thousands of current users, the update of hundreds of thousands of records per hour, and databases containing tens of millions of records, without requiring noncommodity hardware.

Security

The system must support a fine-grained security model for precise access control. The software platform and, if SaaS-based, the hosting infrastructure, should be subject to regular security audits from an independent firm and the vendor should make the results available.

Uptime

For SaaS-based products, vendors provide up-time guarantees that reflect their confidence in the availability of the service. Some vendors just offer a pro-rata refund, while others return the entire cost of that month's service, or more, if the target up-time is not met. If the product is installed inhouse, it should support high availability options so that service can continue even in the event of a motherboard failure.

Reporting

The system must support dashboards, charts, and reports that provide quick insight into business processes. But passive access to information is not always enough, so it should also support the creation of business rules that provide active notification of any problems.

Standards Compliance

The system should support standards such as HIPAA, ADA, ITIL, and CFR 21 Part 11.

Platform Choice

The vendor should offer a SaaS option so that customers don't need to provision a server to get going. Once the solution has proven itself, it should be movable to their choice of in-house Linux or Windows server to allow full integration with sensitive back-end systems without impacting the firewall.

Web-based to Reduce Maintenance

The product should be 100% web-based so that no installation or upgrading of client software is required. It must support the customer's choice of browser.

Backups

System backups should be fully automated and include everything necessary to move the entire deployment to another server or to restore in case of disaster.

Upgrades

Upgrades should require little effort and must allow migration from any revision to any later revision without affecting customizations.

Cost

The cost to get started must be reasonable and the product should provide a rapid ROI, ideally within the first few months of use. The time required to get a reasonably complex production system up and running should therefore be no more than a few months. The cost of extending the system to cover new processes should be modest and the cost structure should be simple and inclusive, ideally without incurring per-function charges or hidden extras when you extend the system.

Vendor Independence

IT staff should be able to extend and maintain the system themselves after training, rather than tying the company to long-term dependence on $300-per-hour consultants. Ideally, the training time should be short. Systems designed to be maintained by the users may require a week of training to reach proficiency, whereas those designed without this criterion in mind may require over a month of training and carry increased effort/risks when making changes.

Company Stability

The vendor should have a ten-year or more history of providing enterprise solutions. For CIOs of large companies, the vendor's track record with other Fortune 500 companies is most relevant. For start-ups, experience with small companies is of greater interest. The vendor should be financially sound and profitable.

Risk Mitigation

The vendor should be able to describe exactly how the software addresses current business needs and demonstrate it running at least one specific process prior to purchase.

The vendor should be willing to commit to a fixed-price implementation for the entire project based on a mutually agreed specification.

Different vendors may offer different forms of refund if a project fails, ranging from a credit towards additional software purchases, to a full cash refund of all software costs and consulting services. The strength of the warranty indicates the vendor's confidence in their software and implementation services.

A sample response to the above items is provided here.

Leadership Skills

"A CIO has to demonstrate the right level of sophistication for the job. A lack of leadership will be the first thing to expose a CIO's incompetence," says John Stevenson, a former president of the Society for Information Management, a professional organization for IT leaders.

As Gartner points out, "Leadership and management are different, but complementary. Management is about execution. Leadership is about change." CIOs must "influence and lead their business colleagues by influencing their view of IT."

This task is a lot easier if the changes being proposed are clearly desirable from a business perspective. For example, if the CIO can propose changes that shorten application deployment times, reduce complexity, and improve IT responsiveness, buy-in will be immediate and enthusiastic. Demonstrable success in achieving such goals will give the CIO the credibility and political capital necessary to institute other changes whose benefits may not be so immediately apparent.

Conclusion

The true success rate of IT projects is currently an abysmal 25%, yet it is possible to virtually ensure success. This does not take superhuman effort or resources, just proactive, disciplined management and adherence to principles learned from hard-won experience and encapsulated in this report. To summarize:

- Begin with clear and complete requirements and engineering specifications.
- The needs and requirements of both the business stakeholders and the end users should be carefully considered early in the process, and revisited throughout development through consistent communication.
- Avoid gold-plating wherever possible and clearly document the cost and impact of any change requests. Make sure that the people requesting them take responsibility for the impact.
- Assembling the right team with the necessary expertise – especially a tough, experienced Project Manager – is vital, and a disciplined hiring process can help make it possible.
- Leverage existing technology to avoid reinventing the wheel, avoid custom coding whenever possible, and don't forget to consider infrastructure and capacity issues.
- Establish a good QA process before turning the product over to the users, and provide an early demo environment in order to capture feedback.
- When sourcing third-party software, vendor solutions should be thoroughly vetted for scalability, integration, security, auditability, and any other factors which are critical to the project. By requiring vendors to share project risks, success rates can be greatly improved and costs contained.

About Agiloft, Inc.

As the global leader in agile contract lifecycle management (CLM) software, Agiloft is trusted to provide significant savings in purchasing, enable more efficient legal operations, and accelerate sales cycles, all while drastically lowering compliance risk. Agiloft's adaptable no-code platform ensures rapid deployment and a fully extensible system. Using contracts as the core system of commercial record, Agiloft's CLM software leverages AI to improve contract management for legal departments, procurement, and sales operations. Visit www.agiloft.com for more.