FINANCIAL SERVICES

GREG CROUSE
Managing Director
Financial Services, Banking
214.766.2001
greg.crouse@navigant.com

navigant.com/financialservices

About Navigant

Navigant Consulting, Inc. (NYSE: NCI) is a specialized, global professional services firm that helps clients take control of their future. Navigant's professionals apply deep industry knowledge, substantive technical expertise, and an enterprising approach to help clients build, manage and/or protect their business interests. With a focus on markets and clients facing transformational change and significant regulatory or legal pressures, the Firm primarily serves clients in the healthcare, energy and financial services industries. Across a range of advisory, consulting, outsourcing, and technology/analytics services, Navigant's practitioners bring sharp insight that pinpoints opportunities and delivers powerful results. More information about Navigant can be found at navigant.com

IT PROJECT SUCCESS FACTORS

INTRODUCTION

There are many studies that estimate the percentages of project failures (anywhere from 30%–70% depending on the study), but in our opinion these studies take many projects in the aggregate, and in so doing, fail to mention best practices techniques and do not factor in the root cause for these issues.

After over 20 years in implementing, assessing, and remediating some of the largest IT projects in the world, as well as being an expert witness in several very large systems implementation lawsuits, we have collected many first-hand experiences of what works and what does not.

In this article, we collected these experiences and success factors that can keep your most important projects in green status.

The areas that we have found to be critical for systems implementation success are:

- Scope Control
- Executive Buy-in and Initiative Size
- Requirements Criticality
- Testing Success
- Date & Budget Flexibility
- Vendor Management

As one would guess, the real difficulty is getting all these right. If only 5 out of 6 are successful, you still may have trouble keeping your project out of the ditch.

I. SCOPE CONTROL

You no doubt have heard every cliché related to change: “the only constant is change,” “change is hard,” etc., but when it comes to Scope change and Change Management, almost every implementation seems to trip on this one.

When we refer to scope control, it is typically related to change to an initiative after the blueprint is set, project plans are made, charters signed-off, etc.

It would be unreasonable to expect that scope change should not be done; it is inevitable, especially when dealing with large development projects, the goal is to be cautious and judicious about change.

Success rates in fields such as civil engineering are much higher than IT projects. While it is true that construction projects do at times go over budget and encounter schedule delays, the majority of projects do get completed, and the end product generally fulfills its intended purpose. If the failure rates experienced in the IT sector were replicated in civil engineering, many of our roads and bridges would have gaping holes as well as numerous bridges that go nowhere. The simplistic reason for this is that bridges and road projects don't start going North, and then have the requirements change to alter the direction East.

So the goal is to be: A) judicious about prioritizing your budgeted scope change, B) follow a strict, consistent change management methodology and C) communicate the change broadly

A. **Be judicious about the change. Apply the acid test:**

   1. **Can we live without this change until a later release?**

   2. **What are the impacts of this change at a high level (i.e., schedule, budget, technical)?**

   3. **Why didn't we think of this before (e.g., was there a good reason for not including this in the first place)?**

   4. **Does the financial benefit of the change exceed the associated costs?**

B. **Follow a consistent Change Management methodology. This seems self-explanatory but it's actually more complicated than most think. Utilize a tested Change Management Methodology (there are many available), or adopt your broader, corporate Change methodology for your project, and use it throughout the project. However, the issue seems to be that when you get to "firefighting" mode, the methodology goes out the window. Your adopted methodology needs to include emergency change, but at the same time, true emergencies are relatively rare. Many organizations are beginning to leverage "Agile" frameworks (Scrum, Lean, ASD, AUP, XP, etc.) that move from more traditional or waterfall methods to more iterative approaches. Keep in mind that in our experience, the first experiment of an organization into Agile usually takes longer than planned and does not typically result in the level of documentation that an organization is used to. So our advice is to take it slow, start with smaller programs, and review the documentation created earlier than you have historically.**

C. **Communicate the scope change broadly. There should be no surprises about project changes. This is overly simplistic, of course; those of you who have done this know there are usually several large scale changes every month, so your project personnel get fatigued about your change emails in month two. We've found that targeting these communications to affected individuals helps (albeit more work) as well as broadly publishing the change log. Additionally, part of your overall methodology should include change management as toll gate criteria, and these toll gates should have input by all personnel affected by the change(s).**

II. EXECUTIVE BUY-IN AND INITIATIVE SIZE

Most large organizations are quarterly driven; so when an initiative is begun that has no payback for multiple quarters, this by nature causes stress. Couple this with an average tenure of a CIO of approximately two years, and you are sure to get executive fatigue.

Humans, by their very nature, resist change in a professional setting; their jobs may be fundamentally changed, or they might even disappear. To combat this, every large change must have a strong change agent. This must be a significantly "important" agent role (the larger the change, the more important the change agent). It is also very important for the change agent to be sufficiently involved with the program, to set the tone as far as budget, timing, and scope.

The longer an initiative goes without any return, the more pressure that is put on the team to deliver results and free-up valuable resources. For this reason (along with many others, such as failure risk, complexity, and higher-degree of difficulty to trace the results with the actual design), "big-bang" initiatives are strongly advised against. In nearly every recent instance of a big-bang that we have seen, there was an opportunity to derisk the project and accelerate the benefits, whether through a quick win, a pilot, or an early-phase release. The odds of success are inverse to the size of the initiative, in other words, a $100 million big-bang implementation has lower success odds than a $10 million initiative.

III. REQUIREMENTS CRITICALITY

Despite the numerous tools that exist for requirements management, along with the many rules about what makes up a good requirement (i.e., verifiable/testable, concise, traceable, consistent, complete, necessary, and system agnostic), this seems to be another area where most projects trip up.

In our experience, large projects:

A. **Make too few requirements (not testable)**

B. **Do not link requirements to test cases (traceability)**

C. **Dictate a technology direction**

D. **Are not complete**
(specifically in the cases of exceptions)

E. **Realize the benefits of packaged software. In many cases, the packages are so heavily customized, they might as well have been custom solutions, and in these cases, organizations are sometimes left with a support nightmare.**

Luckily, this is the area where small changes can yield big results, and there are many mature offerings in this space, not only for tools, but for methodologies and templates. The best-run projects utilize these tools, while they are a significant investment in both time and money; they usually pay off through the first large project that gets implemented successfully.

IV. TESTING SUCCESS

There are many common testing phases (and many models in the shape of a "V"). The major testing phases are: Unit, String/ Integration, System, and Acceptance. There are subphases that are just as critical (which are often overlooked), specifically in the area of systems testing related to performance, stress, and load tests. Additionally, we have seen a "model office" test, where environmental variables, including the location, target state process, and technologies with true end users are employed to validate usability.

In our experience, Unit, and Integration tests are usually performed uneventfully. It is in the later test phases where problems arise. For instance, we have seen instances where the user acceptance test is truncated (this is usually due to lack of client availability, so all that ends up happening is an additional, short cycle of Integration testing) or where all the performance testing is omitted. Unfortunately, there is no single benchmark for testing effort as a percentage of the overall program, but we have seen successful implementations in the 20%–40% of overall project spend in testing.

V. DATE & BUDGET FLEXIBILITY

Dates are critical for a projects success. Detailed plans that are base-lined and executed according to plan are the bedrock of a successful large project. However, one must always realize that if any significant changes are made (scope changes, leadership changes, requirements changes, or even if mistakes are made), your date will likely need to change. Too many times, we see situations where massive changes in scope are injected into a program, only to see the existing date stay the same (or worse, move up). The reality is, it's a date driven by a compliance entity, or it's a date you've stated to shareholders or other important constituencies. If the date is THAT critical, then don't make the change. It's difficult, and in some cases impossible, to get both. Nearly 100% of the large programs we have been involved with, or conducted a root cause post implementation review, had symptoms of "date compression." One way successful projects deal with this is to meet an original date with the original scope and move change to later releases. Likely, this will result in "throw-away" work, but when you've seen some of the project disasters that we have seen, throw-away work is often preferable in comparison.

Another best practice related to scheduling and budgeting is to have an adequate contingency, both in time and in budget (often 10% or more). This contingency should be monitored closely, as the use of this contingency is a good early warning sign of project issues.

VI. VENDOR MANAGEMENT

There is a LOT at stake in large IT implementations. In some cases, the entire company is at risk (there are numerous case studies with many examples about implementations that incurred so many problems, resulting in the company going out of business). Combine this risk with the fact most IT implementations are not undertaken by technology companies (outside of Google, Facebook, IBM and the like), so these implementations are being conducted by organizations whose core competency is something other than systems delivery.

This is why there are literally thousands of companies in the world who focus exclusively on implementing systems. These "guns for hire" are experts at implementing, managing, and even running IT implementations and organizations. Don't get us wrong: most of these companies do actually "partner" with the client and seek to get the best result possible. It is very rare that a company tries to unduly benefit from a transaction (it is a small world, and word of mouth can damage any company), but sometimes during a difficult project, the relationship breaks down — it becomes "us vs. them" and then finger-pointing starts. This is where lawsuits happen, and when an organization is in this death spiral, it is nearly impossible to get out.

So how do you avoid getting in this trap?

A. **Hire your own army – Obtain the best and brightest implementers to manage the project and your implementation vendor, but these individuals are paid directly by you. Their livelihoods are on the line, and if you get the right ones, they can manage your strategic project effectively and look out for your best interest. The downside here is that, unless you execute many of these initiatives, you have to hire and fire, which reduces the pool of people that want to actually do this (and it increases the cost).**

B. **Engage an entity to manage the implementation – There are entities that don't perform end-to-end implementations. Instead, they advise a client in the oversight of an implementation. In this case, you want an entity that will act in your best interest and should not have an implementation capability (or sell a set of products), as these create bias in their dealings with you (and the other vendor). There are engagements that are so large, that this is the only viable option (roughly $50MM and up), and most mature organizations follow this model for their truly large engagements.**

C. **Trust your vendor management – This is the path of least resistance, and the path most organizations follow for a normal-sized implementation. In this model, vendor management tries to create a contract with penalties and rewards, metrics and service level agreements ("SLAs"), to create a balance between the vendor and the client. Additionally vendor management actually holds the vendor accountable for implementation details (dates, quality, etc.) vs. just set up the contract and walk away. In nearly all of the transactional lawsuits we have seen, contracts are silent on specific roles, responsibilities, and SLAs. It is important to note that if you follow this strategy, the organization must have a very mature vendor management function (which unfortunately is a rarity).**

CONCLUSION

The rewards of a successfully executed IT project can be tremendous — but they are matched by pitfalls and stumbling blocks that spell failure for many, if not most, ambitious attempts. Without a great deal of experience, it's difficult to gain perspective about common hazards or to apply best practices in a meaningful way. Even a well-planned strategy can go wrong when not supported by a team of seasoned experts with practical knowledge. The IT leaders who launch with a sharply focused plan, clear guidelines, and well-chosen support are the ones best equipped to shepherd a major IT project to its successful conclusion.

Navigant has deep expertise and experience helping clients implement large complex IT projects. To discuss IT project management and Navigant's Banking Technology services, please contact Managing Director Greg Crouse at 214.766.2001 or greg.crouse@navigant.com.

Navigant Consulting is not a certified public accounting firm and does not provide audit, attest, or public accounting services.
See navigant.com/licensing for a complete listing of private investigator licenses.