How to Write a Technology Game Plan for IT Projects

SamsonaSoftware.com

These days, a growing number of companies are adopting techniques from IT (Information Technology) world for solving common problems. For instance, some of these companies are adopting the use of version control for non-technical documents and other objects of collaboration in the workplace. Another technique more companies should adopt is the documenting of solution proposals for non-technical problems. In IT, the use of Solution Architecture Documents (SADs) is common when translating non-technical requirements into a technical specification that can be executed by downstream tech partners. Unfortunately, too many SADs have become very impractical, bloated, and often uncomfortable to work with because the accompanying templates (whenever you can find them online at work) are painful to use and often get in the way of creativity and innovation. They are also kept separate from the user requirements document, creating confusion as to what items belong in which document. In essence, the current and varied formats of SAD documents have become a problem in themselves and usually end up gathering electronic dust in some SharePoint or other collaboration site folder.

That's not very innovative.

With companies like GitHub turning on its head the notion of old-school corporate bureaucracies, it's time to put away old-school approaches to proposing solutions. The idea of simplifying requirements gathering is nothing new, not even in IT, but creating a collaborative solution proposal process with a simplified, comfortable approach is new for Solution Architects – or shall I say, *Solution Coaches*. Perhaps replacing the expression "Solution Architecture Design" with "Technology Game Plan" is more fitting with the role of a Solution Architect, who really is a solution coach.

In every work context, workers encounter problems that need solutions. That's normal and expected. Identifying these problems and proposing solutions to them should be painless and easy to document because both functions, identifying the problem and proposing a solution, lead to innovation. A technology game plan is needed to realize the solution, and every game plan requires a team where each player has a certain role to play. It is up to the Solution Coach to come up with the best game plan for be victorious. Each team member, called generically a Solution Partner, should be specialists in their respective field. There is a Coding Partner, a Data Partner, a Security Partner, Testing Partner and an Infrastructure Partner. A Management Partner covers the usual role of a project manager, but can also be an executive stakeholder or sponsor.

The Technology Game Plan should be formatted to have section geared towards each partner. This is important because current Solution Architecture Design templates have archaic sections broken down by logical, physical, constraints, assertions, etc, that are not always obvious upon first glance. It's better to have a clearer document the is more direct in terms of which sections belong to which partners to execute. Imagine a sports game plan with a logical and physical section. Which partner is logical? Which one is physical. In my experience, on project after project over the years, the same questions always comes up: "Which section belongs to me? What does logical mean? What is physical?" So a simpler, more direct outline should look something like this:

Technology Game Plan

The Problem – Can be a link to an external document or the actual verbiage added here. Should be plain English with a complete thought per sentence and a noun-verb-noun format where the actor, persona or system is obvious. Should be no more than a paragraph or two.

The Vision - Can be a link to an external document or the actual verbiage added here. Ideally, should be articulated in the form of a user story or use case.

The Solution – What's the game plan for each team member (or solution partner) to solve the problem according to the vision outlined by the key stakeholder? More on this section in a bit.

Before there is a Vision, there has to be a Problem to be solved by the Vision. Just a simple description of the business problem is needed, as mentioned above. If your business is being fined by the government for failing to respond to certain customers in a timely manner, then describe how the current system or business process is failing. This is where approaches such as Lean Six Sigma comes into play as they drill down to the root business problem by asking a series of cascading "why" questions.

The question to be answered for the Vision is "How do I want the solution to solve the business problem?" A good way to envision this is to imagine you bought a robot and you wanted to tell the robot what you want it to do to solve the problem without getting into technical detail (the robot can handle the details, it just needs for you to speak the verb command and what outcome you expect when it is done executing your command). So you might tell the robot "As a stock analyst, I want you to gather all the top 100 stocks in the U.S. and display the data in a visually appealing way that supports my decision making."

Every problem does not necessarily require the same set of *Solution Partners* to solve them. Some problems are repeatable while others occur for the first time. Sometimes a worker (aka *Problem Partner*) may not know that a problem they've identified was already solved before. Even these problems should be documented because it may point to a lack of available online training or mentoring. The solution in such cases is very simple. But the fact that the Problem Partner brought up this problem is evidence of that worker's ability and willingness to innovate at any level, and the *Solution Proposal* becomes evidence of their desire to be creative. It's good to encourage that because if makes the worker more comfortable to identify even more problems and, thus, to spark innovation.

As mentioned in another article, software design principles borrow heavily from a non IT source: classic architecture. When an architect designs a building, he or she is solving a problem and take into account what we now call "user experience" (aka UX). In other words, when a person enters a building, what is it that the owners of the building want them to accomplish? What problem is being solved? With a home, there are many problems being solved, such as the need to cook food, sleep, use the bathroom, be protected from the elements, be cool when it's hot and be warm when it's cold, etc. Each of these problems has a solution that does not require the same Solution Partner. For instance, to install plumbing, you don't need a Solution Partner who installs windows. Thus a window installer does not need to provide input into the Solution Proposal. This is natural, and the Solution Proposal should reflect that natural simplicity.

In its simplest form, a Technology Game Plan can just contain a Problem Description and a Vision Description if both the problem and solution are simple. In the Agile world, this is the best approach as it lends itself to the developer working directly with the stakeholder to solve the problem quickly. More complex ones might require a bit more info, but certainly not too much that the partners lose sight of creativity. A new age Solution Proposal for more complex solutions might have the following minimal elements as a starting point:

1. What is the Problem and how do you think the Solution should address the Problem?
    a. Problem Partner 1
        i. Problem: [Describe in plain English when possible, including jargon and acronym definitions]

ii. Solution idea: [Try not to get bogged down in technical details, just describe in plain English as if, for example, you were telling a robot what you want it to do for you]

b. Repeat for each Problem Partner

2. Given your input, here's how the Solution can best address the Problem:
   a. Solution Coach:
      i. The Solution Proposal
         1. Context Diagram with explanations for each component or role
         2. User-friendly description of how the Solution solves the Problem
         3. Repeat if there is more than one proposed Solution to the Problem. Be sure to include a *Solution Comparison* (discussed later)
      ii. Any general risks that might hinder the solution from being delivered on time and within budget?
      iii. What Solution Partners will be needed?
      iv. Is a Problem Coach needed?
   b. Solution Partner 1
      i. Diagram if any
      ii. User-friendly description of how your part of the Solution solves the Problem
      iii. List any security or other risks specific to your contribution and how they are to be mitigated
      iv. Any legal, regulatory or corporate policies apply to your contribution? If so, list them briefly and add any online links to details
      v. Attach any necessary documents to explain the process you will use to solve the Problem
   c. Repeat for each Solution Partner as needed
3. Any other legal, regulatory or corporate policies apply that have not been identified? If so, list them briefly and add any online links to details
4. Approval Signatures if needed

Notice that in the Partner sections, there is a dynamic section that can scale to accommodate any number of Problem and Solution Partners. Also, as with any sports teams, the head coach (Solution Coach) may need assistant coaches (Problem Coaches) to ensure that all work (done by Problem Partners under them) needed to articulate the Problem is done effectively. Usually, only a Problem Coach is needed to document most business problems, but some complex problems may require Problem Partners to do some leg work in a particular area of the problem.

Implementing some solutions may not require sections 3 and 4, depending on the nature and scope of the problem. Be sure to ask key decision makers at your company if the proposed Technology Game Plan might incur any level of compliance issues or risks.

Some Technology Game Plans will require signatures from decision makers in the organization, depending on the level of complexity in delivering the solution. The Solution Coach should work with either the Lead Problem Partner or an Executive Sponsor to determine if signatures are necessary. It is good not to burden the Problem Partner with seeking signature info so as not to stifle innovation and creativity.

Let's go over the roles.

Solution Proposal Roles

Problem Partner

This worker is the one who noticed a problem and was inspired to find a solution to it. This is how innovation starts. Some problems may have more than one worker who collaborates to identify the problem and have separate but dependent ideas for solving it. This is typical in workflows. In such cases, it is good to have a *Problem Coach* who champions the cause on behalf of other problem partners. This keeps from having more than one worker bottlenecked in the day-to-day details of working with the Problem Partners to come up with a solution. The Problem Coach can keep the other problem partners updated on the solution's progress and also gather updates from these same partners when they come up with other ideas or provide clarity on existing problem descriptions. The Problem Coach can be a project manager and probably should have some experience managing projects.

Ideally, the Problem Partner should not describe the Problem or suggest a Solution Idea in a manner that is too heavy with work jargon or acronyms. Instead, a good technique for suggesting a Solution Idea is to imagine your manager or team leader purchased a *robot* to be your assistant. What would you tell the robot to do in order to solve the problem? The only rule in communicating with this Imaginary Robot is that it does not understand workplace jargon, slang, vernacular or acronyms. It only understands plain English commands. Finally, keep in mind that you are not necessarily the expert in solving and delivering every aspect of the problem (unless you're truly gifted in multiple areas and the organization hired you as such). By fine tuning the way you describe the Problem and Solution Idea this way, you minimize that chance that a Solution Partner misunderstands your innovative ideas.

Some readers may say, "I don't want problem partners explaining the solution because they might get bogged down in the delivery details." Although this has been a long-held rule of thumb in IT, the fact of the matter is that problem partners always imagine the solution, whether you forbid them or not. In practice, requirements have always suggested solution ideas, and problem partners feel the nee d to convey their innovations and desires as they come to mind. Sometimes it's difficult for workers to explain a problem without also explaining how they might solve them. The trick then becomes to allow them to express them using the Imaginary Robot technique described in the previous paragraph.

Problem Coach

A Problem Coach coordinates all work, including research and business process analysis, to help finalize the Problem Statement, and may even help with the Vision Statement if the Executive Sponsor is busy.

Such a worker must work with a Solution Coach and Executive Sponsor to ensure that the organization can deploy enough resources to realize the vision. This aspect of enabling solutions must be kept separate from the process of identifying the problem in order to not burden workers with implementation details.

This worker can, in certain complex contexts, fulfill much of the same responsibilities of a traditional project manager, minus the bureaucratic overhead of some heave project management disciplines. The idea of a Problem Coach is not to manage a project, but to enable innovation. Unfortunately, there is a big difference between the two. With classic project management approaches, the idea is to tell team members what they can and cannot do, and if someone gets out of line, the PM escalates the problem to higher ups, who in turn disciplines the perpetrator. This paradigm is not a way to enable innovation .

Not all problems require a Problem Coach. Simple or trivial problems sometimes only require only a few minutes to solve (such as granting permissions to a resource , replacing an allergy-causing ingredient in a food dish, or adjusting an accounting entry). Each organization should come up with a set of criteria for determining when the complexity of delivery a solution requires a Problem Coach and even a Solution Coach. A discussion on “Determining the Complexity of a Solution Proposal” is later.

Solution Coach

This worker is the brains behind the Technology Game Plan. He or she can be a permanent employee or temporary consultant and is usually an expert in solving and designing problems. A Solution Coach need not necessarily be an expert in a particular business process, technology or industry as this role is responsible for understanding the Problem and lining up Solution Partners to deliver the best Solution. The Solution is composed of business process components or system components, with workers either triggering the process or waiting for the outcome of a scheduled process. The Solution Coach should have an Executive Sponsor (or similar role) to work with to ensure the solution is in line with corporate design guidelines and expectations if necessary. It is important that the process for providing a solution is not unduly burdened with bureaucracy, only enough to meet regulatory, legal or other compliance if the solution rises to that level. If not, then don’t introduce such overhead.

Solution Partner

An expert in delivering an aspect of the Solution. For instance, in IT, a software developer creates or updates a software application, or an enterprise architect might design a component that can be used by dozens of IT projects. Outside of IT, a marketing professional may design an advertising creative. An HR professional might streamline the hiring process. A housing construction supervisor might replace a problem contractor with a better one. The Solution Partner can be a permanent employee or temporary consultant. This last fact is important because too many companies are not strong in IT operations, so this approach to Technology Game Plans facilitates allocating workers when needed in order to solve a problem.

Executive Sponsor

This worker has decision making authority to enable the solution to be designed and delivered if the problem was determined to be of sufficient complexity to have the need of an Executive Sponsor.

Ideally, innovations are best realized in organizations that are not rigidly hierarchical. Obviously, for a lot of organizations, bureaucratic hierarchies cannot be avoided, so the Executive Sponsor should create an *Innovation Incubator* (aka Solution Bubble) that shields the Problem and Solution Partners from as much of the outer bureaucracy as possible. The Executive Sponsor, then, becomes a protective shield for the team, much the way the Façade design pattern works in IT (for those in the IT industry who know about design patterns).

Such an approach is also a good transition strategy for organizations wishing to shed older hierarchical patterns with newer "open collaboration" approaches that encourage innovation. In legacy hierarchies, regular workers feel inherently inferior to managers and executives, some of whom often come across as unapproachable and distant. Sometimes the higher up leaders even have offices that look intimidating. Workers feel they can get bad performance reviews or get fired for any number of reasons, some of which may or may not be valid. But the overall psychological effect is that workers see these as negatives, and as a result, they tend to avoid innovation just to avoid being fired.

This negative vibe is not good for identifying problems and delivering solutions to those problems.

An article in Fast Company about Open Collaboration quoted GitHub CEO Tom Preston-Werner as comparing GitHub's org structure to a "neural network, where you have nodes in a graph and then connections between them, and they go everywhere. Everyone talks to anyone." This is a perfect way of viewing how an organization that encourages innovation should potentially be viewed. (Scientists who work with the brain should also enjoy the analogy). When we think thoughts with our brains, neurons and synapses come alive to carry out whatever action is necessary for whatever triggered that thought.

SamsonaSoftware.com

Determining the Complexity of a Technology Game Plan

In everyday life, some problems are more complex than others. For instance, when a light bulb goes dim at home, you just go buy another one and screw/plug it in. When your car runs out of gas, you go to the gas station and refill it. When you feel hungry, you go and eat (unless you're worried about your beach body – just kidding). These Single Step Problems would get a Complexity Score of 1 on a scale of 1 to 10, where 1 is simple and 10 is extremely complex. Some problems, however, require multiple steps. For instance, a problem that many Chili's restaurants have solved is the problem of paying your tab at the end of your meal or happy hour.

Instead of waving for a server or bartender, you can now just swipe your card at the table to pay for the bill. It may sound simple to the Problem Partner (the Chili's customer in this case), but for the Solution Partner, there are multiple steps involved in making sure the tab is paid properly: the user interface of the device has to present the receipt with the accurate line items, the card swiping feature has to capture the payment info from the card, the device has to submit the payment to the bank card network for authorization, the card network then has to debit and credit the proper accounts (Chili's and the Customer's) in a secure way, etc. So the complexity of this Solution would subjectively be 8 to 10. A Solution Coach and Problem Coach would be necessary.

Every organization needs to come up with a weight/ranking system to determine complexity. For some, a scale of 1 to 10 may not be enough. Maybe you want to instead add 1 for each step it takes to implement the solution, giving you a bit more concrete metrics. Some organizations may not be so objective, satisfied with just a subjective 1-10 ranking system. Whatever ranking system is chosen, it is good to distinguish at what level a Problem Coach and a Solution Coach is needed. As a starting point, on a scale of 1 to 10, you might want to only bring in the two Coaches if it's 3 or higher.

What to Do If There's More Than One Proposed Solution to a Problem?

There are times when problems of a certain complexity will lead to more than one possible solution. For instance, borrowing from an earlier example, if a light bulb goes out at home, you can either get another light bulb, or get a candle or flashlight or lamp. With the Chili's restaurant example, Chili's might have chosen between more than one device vendor for their table side kiosk. In such cases, a *Solution Comparison* might be necessary if the solution's complexity is sufficient enough to warrant it. Earlier, there is a section under Solution Coach which allows for more than one proposed solution. If the solution is considered trivial (a score of 1 or 2, for example), then the choice of solution can be informal. It is best to just list the choices and identify which one was chosen and why. But for more complex solutions, your organization may desire a more formal analysis of the decision.

Enter the Solution Comparison table:

Common Tasks	**Necessity Score**	**Solution 1**		**Solution 2**	
		Quality Score	**Notes**	**Quality Score**	**Notes**
Task 1					
Task 2					
Total					

When preparing a Solution Comparison, remember that every Solution has to fulfill one or more tasks for a given problem. In the workplace, a solution might span multiple tasks in a workflow process. At home, a solution might include multiple tasks, such as painting a wall using faux painting techniques. In IT, a solution might be a web application that has multiple features (similar to tasks), each of which may require a component that multiple software vendors offer. All Partners should work together to come up with a list of tasks to compare the Solution Choices by. For the purpose of the Solution Comparison, try not to make the comparison task list too long to avoid paralyzing innovation. Although a complex solution may encompass hundreds of tasks, you can generally get a good feel for a Solution Choice by comparing no more than 20 tasks or features; if a choice has the highest quality scores for 20 tasks, then it stands to reason that it will probably be the highest for the next 80 tasks. With this in mind, when choosing the list of tasks to compare, be sure to pick the Top 20 with the highest Necessity Scores if you have more than 20 tasks or features.

To arrive at a score for each task item, it is good to meet with all Partners (Problem Partner, Solution Partner, Coaches, and Executive Sponsor if available) to provide the most quality to the scoring effort. One partner, ideally the Solution Coach, can do the scoring if no one else if available, but particularly the Problem Partners will likely have the best understanding of the tasks being compared. If the solution is technical, then it would be good to have the necessary technical Solution Partners evaluate the solutions along with the Problem Partners.

A *Quality Score* is a subjective number from 1 to 10 that indicates how confident you are the proposed Solution Choice can fulfill the requirements of the given task. 1 is low confidence in the choice, and 10 is absolute confidence. A *Necessity Score* is a subjective number from 1 to 10 that indicates how important you think the task is compared to other tasks. For instance, borrowing from our light bulb replacement example, you might give a candle a Quality Score of 3 since lighting is generally limited, and a Necessity Score of 1 since you value the other choices more. (But then again, if your problem revolves around having a romantic dinner at home, both scores may be higher because you'd value candlelight over other forms of light when it comes to dating).

To borrow from our Chili's example, the table kiosk device might have multiple vendors competing to provide the solution to the restaurant chain. The following table provides some example values for illustration using two make-believe vendors (DiningKiosks.com and Table Devices, Inc):

Common Tasks	**Necessity Score 1-10**	**DiningKiosks.com**		**Table Devices, Inc.**	
		Quality Score 1-10	**Notes**	**Quality Score 1-10**	**Notes**
Display the menu	10	5	Wasn't impressed	8	
Allow customers to easily choose menu items	10	8		9	
Quickly route menu selection to cook	8	9		10	
Allow customers to easily request the bill	10	6		9	
Allow customers to easily review the tab	8	8		10	
Allow customers to easily swipe a card	8	9		10	

Securely send the Customer's card info to bank card network	10	7	Needs better encryption	10	
Securely receive authorization and confirmation details from the bank network	10	8		10	
Display successful or unsuccessful payment confirmation to the customer	10	10	Good display of line items	10	Great UX
Total	***84***	***70***		***86***	*Clear winner in our opinion*

As you can see from the table, after assessing each vendor per task item in the leftmost column, Table Devices, Inc. beats DiningKiosks.com easily. With this number in hand, the Executive Sponsor can feel quite confident that a reasonable, good faith effort was made by the team to arrive at a consensus about the best solution choice for the problem.

Next Steps

The goal of this paper is to make sure you have enough tools on hand to solve the world's problems…well, maybe just the problems in your world. Whatever problem you encounter in the work place, this paper hopefully provided you a minimalist tool to propose a solution to it without being held back with heave methodologies or stifling overhead. Although my background is in IT, I am also a business process thinker and have been solving problems abstractly for many years, especially in the Financial Services, Telecommunications, Hospitality and Healthcare industries (among others). Those organizations that are dedicated to coming up with improved hierarchies that encourage innovation should use the Technology Game Plan (TGP) approach describe in this document as a starting point. The first problem that organizations with legacy hierarchies can solve is how to make the organization encourage instead of stifle innovation. The TGP approach should be easy enough to execute immediately without an outside consultant, but if your team or organization needs some outside prodding (and it's needed in political environments), let me know today by using our Contact form.

John Conley III

www.SamsonaSoftware.com