CloudROI’s approach to optimization

If you have looked at our first two articles in this series – *Functional Debt* and *Technical Debt* – then you know it is near-inevitable to completely avoid the two types of debt. You also know that incurring a bit of debt is sometimes necessary and good for your software development projects.

What is crucial is to minimize functional and technical debt. The problem is that debt optimization is easier said than done. No single business can claim to have full knowledge of this kind of optimization for their own debt. To minimize functional and technical debt a holistic, industry wide view is necessary – which is where CloudROI comes in.

By utilizing experts from both technical and business fields, CloudROI can architect a comprehensive method that considers all system users and administrators.

This involves taking an aggressive approach to mitigating debt, aimed at finding that optimized point between technical and functional debt as detailed in this paper.

Brief overview of functional debt

As we indicated in our first paper, functional debt occurs when it costs users more clicks, screens, revisions, time, and more, to use or navigate a system than it would have in a completely optimized system. This phenomenon is often the result of forcing processes into molds and templates that result in a clunky user interface and increased time to execute daily tasks, which do not optimize the end user’s experience.

Oftentimes, the developer has chosen a heavy framework to support these molds and templates, when in actual sense the project should have been small and simple. They end up with a product offering more functionality than needed. The development and maintenance of these extras in terms of functionality is functional debt.

Technical debt

Technical debt, on the other hand, refers to “immature, incomplete, or inadequate artifacts in the software development lifecycle that cause higher costs and lower quality in the long run. These artifacts remaining in a system affect subsequent development and maintenance activities, and so can be seen as a type of debt that the system developers owe the system.” – Carolyn Seaman and Yuepu Guo, 2011.

You incur technical debt when you customize something with software code that could have been solved with point and click configuration. This customization takes more resources to maintain and troubleshoot later, making it future "debt."

This debt quantifies the implicit cost of additional software maintenance that crops up in future due to the technical decisions chosen now to ship things faster.

Some of the activities and decisions that result in functional debt are also responsible for technical debt. This means one debt is never that far from the other; they coexist in your business.

How do you minimize debt?

In our previous articles we suggested a few ways to deal with technical and functional debt in your business.

Here, we explain the approach CloudROI uses to minimize technical debt and functional debt, to optimize your business.

Business first mentality

Most of the value in IT systems comes from what they can do for the frontline business personnel. To minimize both technical and functional debt, you must fully understand the business processes that are aligned to the system.

1. **Functional minimization**

To achieve debt optimization from a functional perspective, we start by hosting discovery sessions with key stakeholders from the business. In these sessions, we review why the current processes are the way they are, to get to the bottom of the debt situation.

Often, environmental and political issues within a company can stand in the way to innovation. Such deterrence frequently comes in the form of people wanting to protect jobs by not innovating or changing their daily process. This sort of thing can be a cancer within the business.

Therefore, to attain functional optimization, we begin by creating an environment that allows these issues to show themselves for what they are. This takes the form of sessions where we discuss business process requirements, regulations, legal and business environment factors, customer requirements, and partner uses. A lot of valuable information usually comes out of these discovery sessions. As it is, much is revealed when you are trying to teach someone your processes.

Third parties such as external consultants are more immune to being influenced environmental or political issues within a company, so they can often see things that company employees or stakeholders cannot. In essence, external consultants bring a much-needed fresh set of eyes into the equation.

Determining the best possible state of business processes

Based on the insights from the discovery sessions, we help the business dare to dream a bit and determine the best possible state of business processes.

In the process, a couple of questions come in handy: could these processes be automated? Who determined that this process should be the way it is? What exactly do you need to make this process report?

While the responses to these questions often vary from business to business, they serve the same purpose across the board: establishing the actual state of the business process, and providing a roadmap of everything needed to get the job done in the most efficient manner possible.

Painting out the business process as a picture

A picture is worth a thousand words. So we map out business processes to the current-state technical documentation, or create a new set of documentation that can be utilized by the business going forward.

Presenting these process maps in a visual form often helps bring out a more vivid picture that stakeholders can easily understand. They are way more effective than text documentation that does not parade by a concrete enough definition.

While at it, we also show some examples of such documents that have a lot of detail in them.

2. **Technical minimization**

Optimization from a technical perspective lies heavily in the hands of the technical staff. So the discovery sessions in this case often primarily involve the company's technical leads, system administrators, and developers.

These sessions cover the IT environment, applications, and their respective uses. Here is also the point to gather any technical documentation available, review source tracking systems such as JIRA, determine development and release cadences, and review policy for application updates.

Who owns what? We help determine which IT staff owns which system as they will be responsible for any technical changes implemented in these respective systems.

Some of the vital questions geared towards technical optimization include: what systems are absolutely critical to getting everything working at optimum? Who originally implemented the systems? And was it in-house staff or were external resources involved?

All these are questions that help bring out the underlying technical problem in the company.

Then there is the question of time. How long has it been since the applications code has been reviewed? Remember, technical debt accumulates interest with time. If the code has been in use for more than two years without review, then it is overdue for a code review.

Central source of truth

The other key thing is to determine who the central source of information for the IT department is, and what systems are used to back them up.This will vary depending on the organization size.

For a large organization, it is commonplace to work with different software houses in evolving and maintaining information systems. As a result, the missing conceptual integrity may lead to architectural stratification that opens room for technical debt.

It is also common for applications to become rapidly outdated, and it can be difficult to evolve due to high maintenance costs that only accrue over time. Taking such a setting as an example, the challenge in maintaining the quality of an information system is apparent.

How do you solve this kind of problem? How do you onboard new users in this kind of setup? What we do is look at all of these holistically, whether the IT's central source of information is a ticketing system like Jira, HPs Agile Manager; or a comprehensive management information system.

Also, we consider the primary systems the company utilizes, where are the most users actively working in the system? Along the same line, we work to determine the amount of support requests and whether or not they are aligned to the volume of use of each system.

Automation in play

The next area of focus is whether automation is used anywhere in the company. By automation, we are referring to those systems in which there is significant substitution of human effort with processes that can operate without human intervention. To get to the bottom of a company's state of automation, we look at the history of automation in that company. Have they tried to automate before? What were the results? Were they successful?

Today, it is hard to talk automation without robotics, or the use of robots, coming to mind. Robots are equipped with abilities equivalent to the human senses of touch, vision, and hearing – so they can perform tasks traditionally done by humans. Ideally, automation does not have to be in the context of robotics, or the realm of manufacturing. You probably know how critical it is for businesses to manage their customer relationships to succeed. Customer Relationship Management (CRM) platforms like Salesforce today enable companies of all sizes to organize, synchronize, and automate aspects of their customer interactions and the resulting back office activities.

One of the key identifiers of a potential automation is whether or not the process is stable, repeatable, and can occur on smaller set of variables.

Another factor of determining how ready a process is for automation is how complex or rigid the process and its resulting output is. If you have got a very complex process requiring tons of variables from different systems, it may not be a good candidate for automation.

To put it simply, you have to have very well-defined processes in order to successfully automate.

Calculating the return on automation

When you implement a business process automation (BPA), you will want to determine the amount of change it brings.

Integrating this automation means changing the way your business operates, training staff to follow new processes, and learning how to use new software. All of this is in addition to the capital investment you put in.

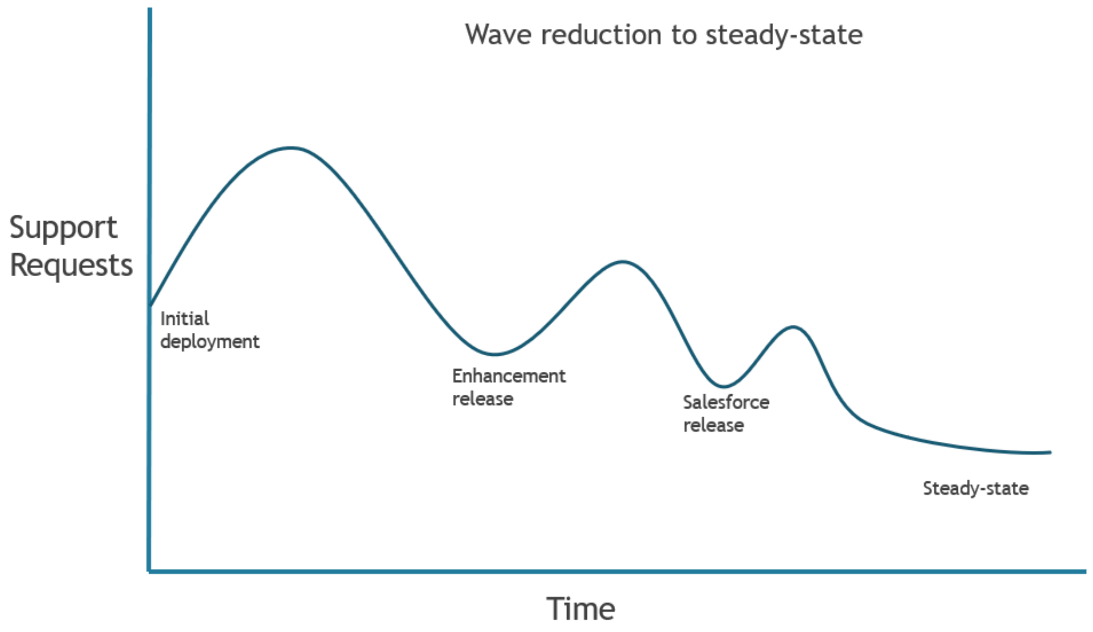
You will want to measure the value of this change; the return on your investment.

To calculate the ROI, divide your net savings by the cost of software and its implementation (your investment). To get the percentage ROI, multiply the resulting figure by 100.

What this means is, you are going to have to perform an estimate of the number of hours saved through automation. This is necessary in order to get your net savings from which you will calculate the ROI.

Wave reduction as the path to getting into an optimized state

The objective of technical and functional optimization is to reduce the overall number of support requests to the IT team. This is a good benchmark for how well your optimization efforts are working. In the graph below we show an additional deployment of the software system, particularly Salesforce.



Once an enhancement is released, the number of support requests go up. However, as you see in the graph, the time to reducing the amount of support requests goes down too as the volume of support requests is reduced.

Parting shot

Clearly, it is not possible to completely avoid technical and functional debt in a business. However, it is possible to minimize both types of debt and optimize your business operation. It all starts with understanding functional and technical debts, and how they impact your business.

We have provided this information in the other two of our papers, namely *Technical Debt* and *Functional Debt*, so you should have everything you need to keep debt in check and increase your ROI.