Ten Tips for Modernizing Your Applications

WWW.REDPILLNOW.COM

While enterprise applications are usually entrusted to Corporate IT to develop and maintain, a range of options exists for developing small to medium-sized applications. These applications should be viewed as the front window of an IT Department. They provide a daily snapshot of what the IT department is capable of delivering for its customers. Any decision made by a business executive to use corporate IT to help solve their next big opportunity may be based upon what they are seeing in that shop-front window each day. If those applications are fresh, functional, innovative, have visual appeal, and deliver an exceptional experience, then prospective customers may be tempted to enter and enquire about purchasing something new. If what they are seeing in their applications is dated, clunky, slow, or just plain ugly, they might look at other options outside of the corporate IT department such as Shadow IT or citizen developers.

The computing power of the typical smartphone is now over 1,000 times greater than that used by NASA to put man on the moon in 1969. Those smartphones will have an average of 30 applications that have been downloaded and installed by the owner of the device. They may drive a car containing 20 separate computers, and yet don't require an IT department to support it. Thanks to the Internet of Things we now have a growing workforce installing computer devices throughout their house and then programming them to interact with each other to meet the specific needs of their family. We are growing accustomed to real-time data, voice input, and artificial intelligence from items as diverse as front-door locks, solar panels, security cameras, speakers, and smartphones. When we come to the office, they are expecting the same capabilities from our applications. If we can't get what we want from corporate IT we are prepared to become citizen developers and install/program our own solutions just as we now do at home. CISCO's Shadow IT Report found between 17 and 20 TIMES more cloud applications running than the IT department estimated.

The bar is constantly moving. Can your aging legacy applications built on platforms such as Access, Filemaker, Lotus Notes, or SharePoint compete against those now being developed using Low-Code platforms such as Appian, Betty Blocks, Mendix, Outsystems, or PowerApps? Which would you rather have on display in your shop window to attract new customers?

1. Act Now

There is a lot of interest at the moment in how to add newer, modern, capabilities to legacy applications that have often remained untouched for many years. The gap between what each of the applications provides and what is now being expected of them grows with each passing day. Many applications managed by Corporate IT were originally written by citizen developers who took it upon themselves to solve their own business needs with a product that allowed them too quickly and easily do just that. If action isn't taken, it is equally likely that a new breed of citizen developers will step in and solve their needs themselves, usually outside the watching eye of the IT department. No-code and low-code platforms are getting a lot of attention.

Many of the benefits derived from the migration of applications to a new platform aren't the result of the new platform itself, but rather the simple fact that somebody actually spent some time looking at the application, the needs of it users, and then matched current capabilities to those needs. Many of those same improvements could have just as easily have been added to the existing application if only somebody had bothered to apply the same level of care and attention.

Doing something now to those applications is a major step forward to not only keeping those applications relevant to their users, but also in keeping corporate IT itself relevant. It's always a good idea for any career tactic to remain relevant.

2. Ready, Aim, Fire!

Having an overall application strategy is perhaps the single most important part of what you may do with your applications. Start working on it now. Become familiar with the options that exist out there and decide on an architecture that works best for your organization. Revise it over time as technology changes so that when you are asked to do something about those legacy applications you will have a clear idea of where you are heading.

The desire to move away from a specific legacy platform is something I often hear. I usually take that as somebody venting their frustrations rather than it being a strategy. Even if that is what you have been told, we would strongly suggest having a clear picture of what you want to achieve in "moving away from X". It is wise to follow the traditional approach of setting a goal, gathering facts, planning a path forward and THEN executing. Many organizations can have hundreds or thousands of small-medium-sized applications still in use. Any decision that affects that many applications should be considered carefully and be based upon fact rather than suppositions, or unsubstantiated claims from people who have vested interests in seeing a particular outcome.

Ready

Modernization should be viewed as an ongoing process rather than a single project that starts when enough people complain about what they have and ends when enough voices fall silent. If that process hasn't yet started, it's always a good idea to have clear objectives that are set well before potential solutions are evaluated. Included in that should be some clear positioning of no-code and low-code platforms fit in with the overall application strategy. For some organizations, all applications are built using low-code. For others there may be a mix of pro-code, low-code, and no-code applications.

Aim
Having defined a high-level strategy for applications it is also a good idea to carefully target technology trends that are likely to have an impact on your business. To what extent does access to applications from mobile devices make a difference? Is there an advantage to be gained from integrating applications into IoT devices? Would voice input or high-resolution monitors make a difference?

Fire
Once a strategy has been developed and a plan put in place, we are no ready to execute. We know where we are and we know where we are going, so now it's time to enjoy the ride.

3. Don't Generalize. Have A Strategy for Each Application.
Most applications vary considerably in functionality and complexity. It therefore makes no sense to assume the same modernization solution will work for them all. Just because you bought a new hammer doesn't mean everything else has turned into nails. Simpler needs can often be solved with simpler (cheaper) solutions. Save the biggest portion of you budget for the more complex applications. While it is something of a simplification, the key choices that can be made when looking at a strategy for individual applications are the four-Rs… Retire, Retain, Renew or Replace.

Retire
If your assessment is that an application no longer has any business value, or its value is less than the cost of following one of the other strategies, the best thing to do is simply to stop cluttering your application attic with junk and just throw it out. Note: There can be a danger if you rely on usage statistics alone to determine if an application should be retired. It is not unusual for a useful application to stop being used simply because the original owner moved on and nobody explained to the new people where the application was and how to use it. It might also be that a worthwhile application has stopped being used because a key bug (which could be easily fixed) was never addressed. If you do decide to retire an application, you may decide to archive the data is some way such as a PDF first. But again, only do this if you really think there is a chance the data would be needed. Most companies will survive quite OK without a record of who registered to attend your 2003 IT Christmas Party or the discussion threads for that Y2K project!

Retain
In some cases, you might decide that the application is doing what is needed and no further action is required. This might include applications whose end of life is nearing that doesn't warrant any further investment. It can also include scenarios in which you are still

quite happy with the legacy platform and have no need to make any changes. Regardless of the reason, it's usually a good idea to set a review date for every application based upon its strategic importance. An application's business value and technology alternatives — can — and often do — change over time.

Renew

There are a lot of situations in which a review identifies that an application is not meeting its full potential for the organization and/or its users. Because so many applications just keep working it can often be hard to justify investing in regular enhancements to keep them in sync with modern best practices such as a mobile interface, touch screen support, or even an Alexa interface. Some platforms like SharePoint tend to deprecate older features making it necessary to renew applications on a regular basis. Platforms such as Lotus Notes strive to maintain backwards compatibility. This could mean messaging is sent via carrier pigeon and output directed to papyrus scrolls!

Renewal can come in many forms. It could simply be to modernize the look and feel of the legacy application to apply a UI that is consistent with the corporate brand, replace or upgrade to newer controls and plugins that support responsive layouts or touch interfaces, or perhaps simply replacing BMP images with PNG files. Renew can also involve improving the UX with additional business functionality that has long been needed but never addressed. All have the ability to improve the user experience without replacing the platform – especially the data store.

Replace

The final option is to replace the application. This could include replacing the application with 3rd party solutions. Discussion databases can be easily replaced by persistent chat clients such as Slack or Watson Workspace. Document libraries can be replaced with file storage systems such as One Drive, Box, or Connections Files. Specialized solutions such as Help Desk, Fleet Management, Expense Tracking, CRM, ECM, and ERP all have a wide range of third-party solutions that should be considered if these processes are not proprietary.

For proprietary applications in which it makes sense to continue having a custom application, it might make financial sense to move the solution away from the legacy completely (i.e. platform change). This is especially true of larger Enterprise applications where it has been decided a better choice would be to move from a low-code to pro-code platform and replace important business logic written in an outdated programming language. The other reason might simply be a decision to move to a new strategic platform such as SalesForce, Office 365 (Powerapps), or Mendix. Maintaining a mixed application portfolio can create difficulties ensuring the skills are available to maintain and support each. Whatever the reason, Replace is often the most expensive and includes the additional risks associated with having to cutover the data to the new platform.

4. One Size Doesn't Fit All.
When it comes to modernization, almost every vendor has a belief their solution is the best one for your modernization needs. We are usually conditioned to believe the choice is to use my product or somebody else's. That for every problem there is one and only one single best solution. When it comes to modernization it is hard to imagine there is ever going to be one solution that is better than all others in all cases. By definition modernization is trying to hit a moving target and there is always going to be something new and different that somebody else may develop a better solution for. When it comes to finding a match for your modernization needs, don't just consider that each application may have its own solution, also consider the possibility that a single application might benefit from a number of solutions being used to solve different aspects of the functionality. One solution might provide the best outcome for the many people who view the content while a second may best meet the need of the few people who edit the content and a third could be better approach to handling just the workflow aspects.

5. Seek Help.
As a consulting company, we advise our clients to invest between 1% and 5% of estimated project cost in initial strategy development advice and high-level project plans. It's also a good idea to seek independent advice if a project involves product selection, keeping in mind that a vendor tied to one solution is almost always going to recommend its own. When it comes to strategy execution, a number of options are worth considering. Because a project like this is likely to supplement an existing workload, you will likely need to expand your team in the short term to cope with the extra demand, either hiring somebody or engaging one or more experienced contractors familiar with the type of migration and/or technologies that you are going to start using. Alternatively, find experts in the new technology and engage them to work on the first application in order to create a model for your own team to work from, shifting those experts to mentor roles for the next few projects, ultimately transitioning them to on-call status for assistance when problems arise with later applications.

6. Find the Low Hanging Fruit.
Strategies developed for individual applications should take into account the role the application plays within your organization. Personal applications are those that are used by one or two people whereas departmental applications are typically used by a larger group of people located within the one department (which might span multiple geographies).

For Personal and Departmental applications, we should consider carefully the extent to which there is a need for a low-code environment to sustain the application moving forward. If those applications are being (and/or will be) maintained by citizen developers rather than trained professional developers, the skillset of those citizen developers needs to be considered carefully. Some IT departments often cringe at the idea of citizen developers without understanding the economics of application development. Adding a self-service ability for simpler applications not only drives down development costs but can also improve the perception and relationships within corporate IT departments.

For enterprise applications, those used across the organization or across multiple departments of strategic importance, it is important to have a clear strategy of both the architecture and platforms that should be supported for all these applications. "Moving away from platform X" is neither a strategy or an architecture. An application development strategy requires a clearly defined picture of what you are moving to. These will usually be the responsibility of corporate IT to maintain and should be developed in a way that matches governance standards such as documentation, testing, audibility, security, change control, DRP etc. They should also allow the development of applications that meet the expectations of your users today and tomorrow.

Sometimes the timetable for moving away from a legacy platform doesn't always coincide with the effort required to move everything to the target platform, even for just Enterprise applications. If that is the case don't lose sight of the end goal. Plan an interim step that gets you to where you need to be in the short term and then plan to make the final jump as part of a second stage. Don't just look at the paths available to migrate to that interim platform, but what are the options for moving on from there.

7. Fail Fast.

Having established a strategy for all your applications, the order of execution can have a significant impact on the success of the project. The order of execution can determine how quickly you learn if there are possible issues as well as providing a more data from which to revise estimates of the total project cost. There are several options worth considering.

Simplest First

Perhaps the worst strategy to follow is the one in which the easiest applications are selected first followed by applications of growing complexity. It might sound like a good idea to start out easy and work your way up. The problem is that you are assuming the strategy, budget, and timeline are all correct. The easy applications are rarely the ones that are going to cause problems. If you are going to have a failure it is often better to fail fast than it is to delay the failure until a significant proportion of the budget and time has been invested in that approach.

Most Complex First

The opposite approach is to start with your most complex application work. If for any reason the strategy doesn't work you will have invested a modest amount of time and money to find out. If the strategy works but ran over budget or time it may be possible to refine the strategy or revise the process to be applied to those that follow.
Simple First, Most Complex Second

In this approach, we chose one of the simpler applications to use as a learning exercise. This allows us to complete the process once and refine it before applying it to a much larger project.

Most Value First
Across an entire portfolio of applications, we usually find a level of diminishing returns. A few applications can provide a high return on the investment while others provide much less or even negative returns on the investment made. By starting with the applications generating the highest returns, we make the biggest business impact early on in the project. If for any reasons budgets become an issue or resources get reassigned to other projects we will have gotten the highest return from the amount of time and money spent.

Most Users
If one of the goals is to eliminate the legacy platform, the fastest way to start achieving this is to focus on the applications with the highest numbers of users. This process can be further refined by looking at the people who only make use of one (or two) legacy applications and focusing on the applications that are most frequent amongst those. If 10% of your users only make use of one legacy application and that gets renewed or modernized in the first month, then 10% of your users can uninstall the legacy platform very quickly.

8. Deliver More.
It is rare that a major enhancement does not have some operational impact on its users. There is the time taken up by meetings and executing the project. There are potential changes to the behavior of the application that cause confusion and mistakes to be made. And then there are the bugs that find their way out of development into production. If all your modernization delivers is infrastructure change such as moving from the Notes client to a browser client, the users of the application see little reward from their participation in the project and the inconveniences created. It is a good idea to always find a way of adding value to each and every application that is modernized that is at least commensurate with the inconvenience caused. Make it available from a mobile device, add voice input or click to call for phone numbers. Something meaningful that delivers business value to the application and lets your end users know their interests were being looked after.

9. It's a Journey, Not A Destination.
Once a modernization process has started its always a good idea to also start a process to keep your applications modern. If your applications have been left dormant for many years it may take 2-3 years to complete the first round of modernization. This means the first few applications may already become outdated before you have reached the last application.

Plan accordingly.

Technology changes at a sufficient rate that it warrants an annual review to update the application strategy. Mission critical applications should be reviewed on an annual basis. Medium-impact applications once every three years, and low-impact applications at least once every five years. The solutions for modernization are constantly changing, new third-party solutions are constantly entering the market, and the requirements of the applications themselves regularly change.

It is suggested that an annual review process be established and implemented 12 months after the first wave of application modernizations have started.

10. Act Now!
Having read and considered the above tips it bears repeating the importance of taking action now. There are solutions out there to modernize your legacy applications. There are no magic buttons, and no magic timelines that will get you to where you want without a lot of time and effort. Failing to take action leaves your applications further behind tomorrow than they were today.

If for some reason you aren't ready to start your modernization project just yet, there are still a number of proactive things that can be done to prepare:-

- Bring out ya dead! – Remove as many databases from your production servers as you can. Find all the backup databases, development copies, test databases etc that are no longer needed and remove them. Identify databases that are no longer being used and either archive or discard them.
- Play the Dating Game – Match as many databases to templates as possible. It is going to be important to find all the code that is common across multiple applications and template inheritance is a good start.
- Return to Sender – Build/extend a catalog for your applications that identifies the owners, developers, and Subject Matter Experts (SMEs) for as many as possible. This will be useful at a later stage when we want to learn more about each database.
- Just in Case – Where possible, establish a source code management system for each application (or template). Later on, when changes are being made, it just might be necessary to recover from an oops moment.