5 Top Myths of Automated Code Modernization

Mobilize.NET

YOUR APP. NEW. AGAIN.

As enterprise IT departments and ISVs look to the problem of moving legacy applications to modern platforms like the Web, cloud, and mobile devices, it's tempting to assume that a complete manual rewrite is the only way forward.

Yet automated code modernization (conversion) has worked for years in migrating legacy code.

This white paper discusses the 5 most prevalent misunderstandings

There are many myths surrounding software modernization.

For starters, there's the perception that automation is just more trouble than it's worth.

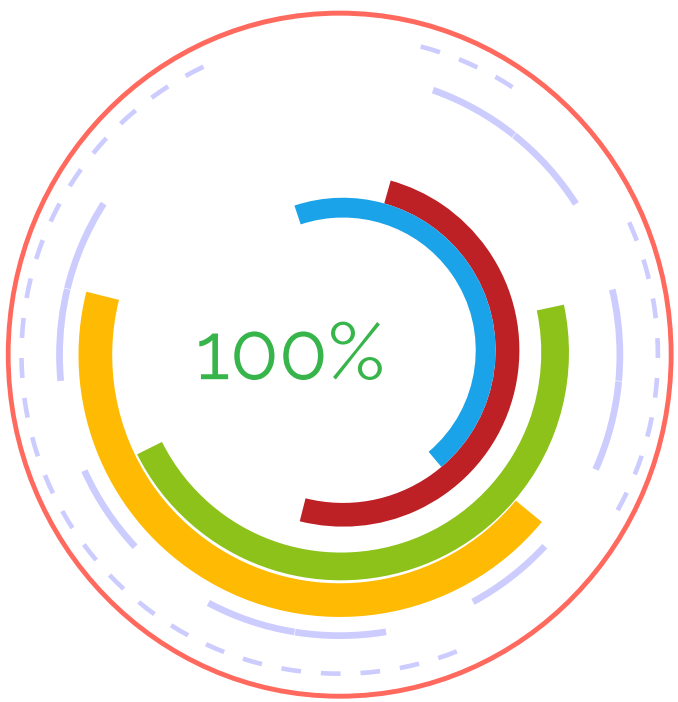
Though this may be true for really small and simple applications, where the setup may not be worth the time, the majority of conversion projects are large enough that there are economies of scale. In these cases, the initial setup might take several days, but then the automation tools can convert thousands of lines of legacy code in just a fraction of the time that it would take to do so manually

Performance expectations lead us to another false impression:

Software modernization can be completely automated.

While it is true that there are amazing technical solutions available that automate most of the work, there are still some aspects that require human intervention.

The most obvious is in the project management and analysis areas, but there are also some code characteristics that cannot be automatically upgraded.



In many cases it's just a matter of the cost of the implementation not making the automation feasible, but in others a native translation may not be possible due to the differences between the source and target platforms.

Finally, it appears that many believe that it's just plain better to start from scratch. It might be the case when you are dealing with a disposable system with useless, outdated functionality and no value at all.

Otherwise, to agree with this idea is to simply devalue all of the effort and thought that was put into developing the application, risking years of business knowledge embedded in these systems.

The truth is, a rewrite from scratch implies a much more difficult task, even though some may claim that it is balanced out by the fact that you can significantly reduce the total amount of code and fix some of the imperfections that were present in the source application. But that's something you can also perform with an automated modernization approach, without incurring all the risk, time and cost.

Due to these misconceptions, valuable resources are wasted in projects that sometimes never resolve to a successful ending. There's no doubt that any software renewal project isn't a simple, overnight task, but a wellplanned automated modernization makes the process comparatively painless. Besides, system upgrades are generally worth the effort.

Of course they involve time and expense, but it provides savings in the long run and a quick return on investment, when your staff stops spending all that time maintaining legacy systems and developing workarounds to compensate for technology shortfalls.

So here are the main 5 myth-debunking reasons why an automatic migration is a far better software modernization approach than a manual rewrite, based not only on Mobilize's own experience in modernization projects but on all the customer and industry analyst feedback and evidence gathered over the years.

1. Code Quality

Automatic modernization technologies today are able to generate code of exceptional quality, where you will not be able to tell if it was produced by a tool or by an experienced developer.

There's real value when you consider that the modernization tools handle the process even when the people involved are not highly skilled in either the source or the target languages.

In the case of advanced VB6 to .NET modernization products, extensive code analysis can be performed to detect patterns that can be upgraded to .NET-like, native structures, making the output more readable and maintainable.

This includes data type enhancements, grammar pattern transformations and detailed code improvements.

On the other hand, an automatic migration does not fundamentally alter the architecture of the original system, even if certain aspects like data access and some pieces of GUI architecture do change.

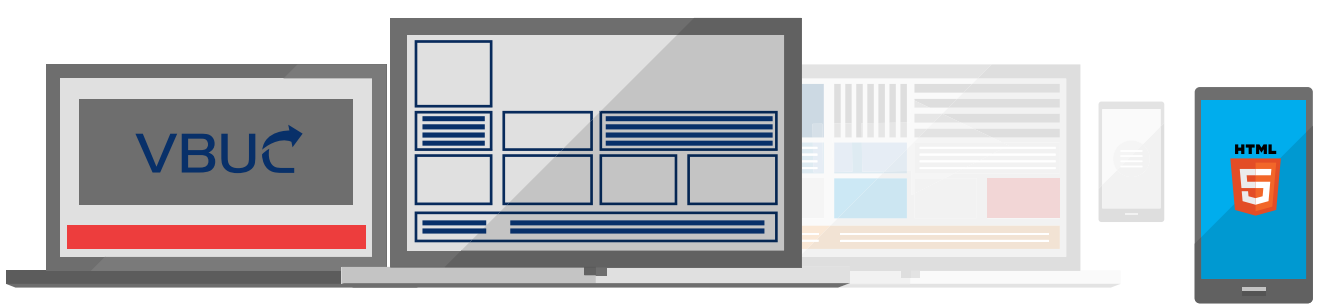
But the output of a good modernization tool, which needs to strike the right balance between automation and quality of the generated code, is ready for application enhancement and evolution.

Mobilize. Net's Visual Basic Upgrade Companion (VBUC), produces native VB.NET or C# code, with no dependencies on third-party runtimes, and preserves the business logic, including all comments, of the original application.

This allows you to take advantage of the technologies available in the .NET Framework, and eases the subsequent development and maintenance efforts.

The VBUC employs best programming practices, and even has the ability to be extended and customized. This allows generated code that to adapt to specific requirements, be it corporate policies or personal styles and preferences, increasing even further the percentage of code automation.

Even in the worst case scenario, where you may need to rewrite a specific piece of the application after the automated modernization phase, the end result will always be a fraction of the total cost and time.

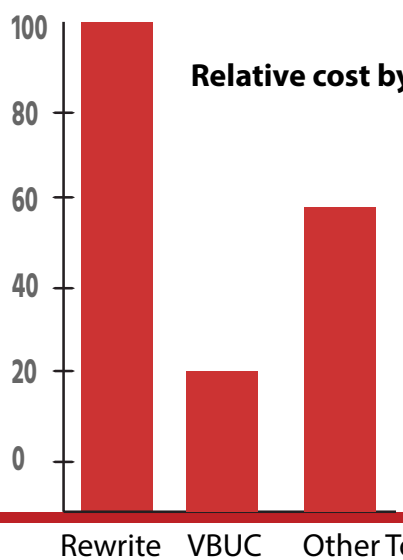


2. Overall Cost

When dealing with projects related to business applications, money is a critical factor. One cannot justify automatic modernization based solely on technology itself, but on cost savings, and when we look at this variable there are several dimensions.

The most obviousis the cost of the actual upgrade process, where it can be said based on experience that an automatic modernization can be done at 20% of the cost of a manual rewrite. And most of that expenditure is due to testing and fine tuning of the application on the new platform.

The graph above reflects the feedback from one of Mobilize.Net's customers, who concluded that an automatic modernization using the Visual Basic Upgrade Companion yielded 80% savings when compared to a rewrite (and 60% savings when compared to a modernization using another tool).



The graph reflects the feedback from one of Mobilize.Net's customers, who concluded that an automatic modernization using the Visual Basic Upgrade Companion yielded 80% savings when compared to a rewrite (and 60% savings when compared to a modernization using another tool).

There's also other savings with this approach that must be considered. Take for example the training of end users. Since the application that results from an automatic modernization is 100% functionally equivalent to the original one, it is not necessary to retrain them.

With a rewrite, chances are that the output is not going to be familiar, unless you manage to follow an algorithmic approach like the one employed by sophisticated automatic modernization technologies.

In some cases, the retraining costs can be enormous. For example, one of our customers estimated that its system would have required more than 6 weeks training time for 3000 users.

All in all, using technology is viable as long as it helps reduce the overall effort and cost. Even more in times of economic uncertainty, where it is crucial to save on scarce, valuable resources.

Minimizing the investment when evolving the infrastructure is definitely a must, and automation is a good way of doing that.

3. Project Time

Normally, a legacy renewal project is a daunting task, but a modernization tool can perform a significant portion of the work automatically, in a fraction of the time and using much less resources.

We have many realworld examples:
Using the same instance of the customer above who compared application renewal solution costs, the following numbers speak for themselves; a manual rewrite for a highly complex application was estimated to take 18 months, involving key personnel from various departments, while a modernization with the VBUC required the allocation of much less resources for only 6 months.

Using Visual Basic Upgrade Companion (VBUC) Banamex was able to convert more than 5 million lines of code in 12 months. Rewriting from scratch would have taken an estimated 175 people and 6 years

One of our published case studies, featuring Banamex, one of the largest Mexican financial institutions and part of Citigroup, shows how they opted for an automatic modernization with the VBUC because of the shorter migration lifecycle it offered, which in turn helped protect their market position. After an extensive analysis an automatic modernization emerged as the most costeffective solution.

Banamex considered rewriting 124 of their business critical VB6 applications, comprised of a total of more than 5,000,000 lines of code, and estimated that at least 175 and up to 185 developers would need to be involved in the project for a period of 6 years, in addition to the resources required to manage and coordinate the normal maintenance of these systems in the meantime.

In contrast, using the VBUC allowed less than half that amount of resources to complete the job in just 12 months .

Finally, there's another success story involving Vertex Financial Services in the UK, who converted a 616,000 lines-of-code application from VB6 to C#. Being in such a competitive industry, they needed to accelerate the time to market of their renewed application, in order to meet the next product release. They accomplished it by using the Visual Basic Upgrade Companion instead of rewriting the application from scratch.

Additionally, the above results were achieved using previous versions of the tool.
The latest release is much more powerful, allowing greater time-savings..

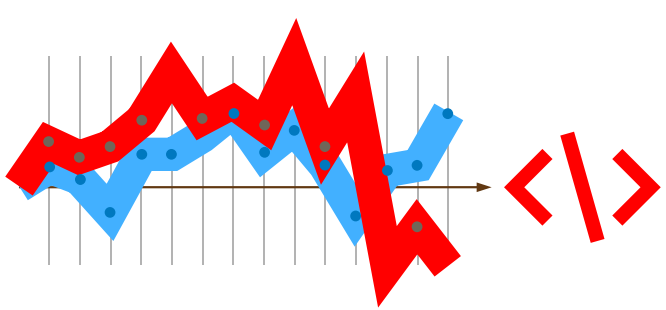
4. Risk

It is a well-known fact that, by nature, humans are prone to mistakes. Add the fact that legacy applications and all the changes made to them during the years are not always well documented, and that the original developers sometimes are no longer available, and you have high risk of business knowledge loss when performing a manual rewrite.

Plus there is always some level of uncertainty involved, which makes costs and project timelines difficult to keep under control.

An automatic modernization process leverages the capital investment made in business applications.

That is, all customizations and business rules will be preserved in the migrated system. Plus, as explained before, this approach provides a much faster renewal lifecycle, and with minimal organizational disruption, which ensures speedy time to market that provides organizations with a valuable competitive advantage.

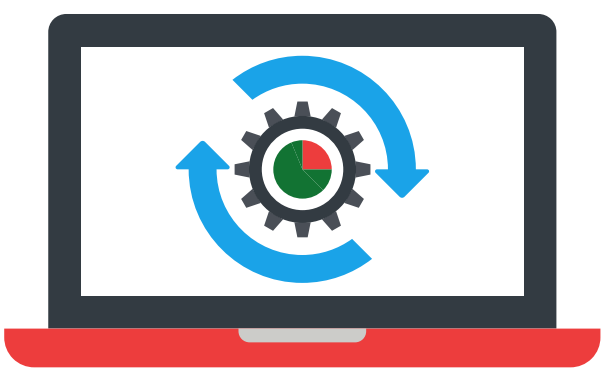


5. Technology Availability

It's simple: many people just aren't aware of the existence of automated modernization solutions.

Case in point, in a survey sponsored by Microsoft in the UK, almost 40% of the respondents answered "False" on the statement "There are tools to automatically convert Visual Basic 6.0 applications to Visual Basic .NET". So it was no surprise when more than 85% answered equally when asked about Visual Basic 6.0 to C# conversion tools.

But these tools DO exist, and have been helping companies and developers for years to leverage the investment embedded in legacy applications.



There are manual adjustments required to achieve compiling running code in the target language, and even sometimes a realworld migration process is an iterative task where tweaking the source codebase beforehand is also anticipated and recommended.

But technology has evolved to the point that it is hard to find a good reason to carry a software transformation effort based entirely on manual labor.

Using automated modernization tools as part of an overall software renewal initiative is the most viable way to leverage the current investment in legacy applications and move them to the latest platforms.

The technology available today produces outstanding results in terms of cost and time savings, risk avoidance and generated code quality.

LET US DO THE MIGRATION FOR YOU.

YOUR APP. NEW. AGAIN.

With thousands of projects and billions of lines of code converted.
Mobilize.Net is the global leader in application modernization.

Need more help?

Let a Mobilize.Net migration engineer help you figure out how to convert your legacy application:

http://www.mobilize.net/talk-to-an-engineer

www.mobilize.net **+1-425-609-8458** **info@mobilize.net**