What screen-scraping vendors don't want you to know

A whitepaper from mrc

Overview

As time passes and systems age, application modernization becomes a growing need among many companies. Applications created 10, 20, or even 30 years ago cannot handle the technological and business needs of today, and are often more expensive and cumbersome to maintain.

The question faced by many companies is no longer "Should we modernize?" Instead, the question is "Which modernization method do we use?" Companies wishing to modernize will undoubtedly run into multiple modernization methods and even more software tools.

This white-paper focuses on one often misrepresented modernization method: Screen-scraping. The idea behind screen-scraping is simple: A company buys software that creates a new modern-looking interface on top of their old applications.

Vendors sell screen-scraping as a quick and easy way to make your applications modern. They'll tell you it's the cheapest and fastest way to turn legacy applications into modern-looking applications. In fact, they will describe screen-scraping as a foolproof modernization method with no drawbacks. They'll tell you what you want to hear.

But they won't tell you how much it costs over time. They won't tell you what happens if you ever modify your applications. They won't tell you about the extra hardware and software support costs you'll incur. They won't tell you that it doesn't apply to business changes. They definitely won't tell you that screen-scraping isn't even modernization at all.

Most importantly, they won't tell you all the drawbacks that you won't discover until it's too late. This whitepaper dives into the secret side of screen-scraping—the side that vendors don't want you to know. Let's investigate a few of the problems associated with this modernization method.

Problem 1: Is screen-scraping really modernization?

Is screen-scraping really modernization? To answer that question, one must first define both terms.

First, what is application modernization? An unbiased definition is difficult to find, as the term does not exist in the dictionary. A vendor will define modernization in whatever way best suits them. Talk to 10 different modernization vendors, and you'll receive 10 different definitions of modernization. An online search for a modernization definition results in multiple vendor-definitions, all slanted to serve their own personal agenda. Even supposedly unbiased modernization articles are written by vendors and slanted towards their view of modernization.

For an unbiased application modernization definition, we must define each term separately. According to the Encarta dictionary, an "application" is "A computer program or piece of software designed to perform a specific task." Additionally, "modernization" is "a way to change something in order to make it conform to modern tastes, attitudes, or standards."

A combination of both definitions results in this unbiased application modernization definition: **A way to change a computer program or piece of software in order to make it conform to modern standards**.

Second, we must define screen-scraping. Techtarget.com contains the most comprehensive screen-scraping definition, which is quoted below:

"Programming that translates between legacy application programs (written to communicate with now generally obsolete input/output devices and user interfaces) and new user interfaces so that the logic and data associated with the legacy programs can continue to be used. The program must also reformat user input from the newer user interfaces (such as a Windows graphical user interface or a Web browser) so that the request can be handled by the legacy application as if it came from the user of the older device and user interface."

In short, screen-scraping creates a new, modern user interface for old applications. It makes old applications look modern without actually updating the old applications.

To recap: By definition, application modernization changes an application to make it conform to modern standards, while screen-scraping creates a new interface for old applications. Screen-scraping doesn't change the application, it changes the way the application appears. Therefore, screen-scraping cannot be classified as modernization.

To conform an application to modern standards, one must change both the underlying architecture and the face of the application. A truly "modern" application must not only look modern, it must be modern inside and out.

Now, some would argue that the act of creating a new, modern interface allows the underlying application to conform to modern standards, and therefore, fit the modernization definition. But, does a screen-scraped application really conform to modern standards? Perhaps more importantly, why does it even matter? Keep reading…

Problem 2: Screen-scraping omits common features

Applications in need of modernization were created 10, 20, or even 30 years ago, with the best features available at that time. Over time, technology advances and features available to application developers improve. However, since screen-scraped applications are merely modern representations of old applications, they lack many of the basic features found in a typical web application, such as:

- The ability to sort by columns
- The ability to search an application
- The ability to use modern AJAX techniques
- The use of modern javascript features, such as drag-and-drop
- The ability to graph data
- The ability to export application data to Excel or PDF format
- The ability to interact with other modern applications

Depending on who uses the applications, implications of missing features range from annoyance to lost business. For example, if prospects use the screen-scraped applications, what will they think when they discover basic features are missing? It tells prospects that a company is behind the times. Not only does this put the company is a negative light, it drives prospects away. Similarly, if customers use the applications, missing features will either annoy them or drive them away.

Besides a lack of modern features, dated architecture can't take advantage of integration with certain apps and services. The reason is simple: The underlying architecture was designed to work with technology that existed when it was created. It can't take advantage of current applications and features, such as Google maps integration.

Now, a screen-scraping vendor will claim it's possible to add all the current features into a screen-scraped application. Technically, it's true. Those with a good grasp of HTML and Javascript can work many of the current features into a screen-scraped application. However, working modern features into a screen-scraped application requires a considerable amount of time and effort, considering they don't come with the features built-in.

However, this creates another problem. Screen-scraping is sold as a way to save time and effort, yet working all the missing features into a screen-scraped application negates the time-saving benefits. Those who screen-scrape their applications have a choice: Spend time adding missing features to screen-scraped applications, or use applications that lack common features.

Additionally, what happens to a screen-scraped "application" if the underlying application needs changing? Keep reading…

Problem 3: Difficult and costly to maintain#

Screen-scraping adds an extra level of complexity and cost to application maintenance. Since the underlying applications still exist, they still require normal maintenance apart from the screen-scraped applications. This leads to a few problems:

First, every alteration to the underlying applications requires re-scraped applications. In other words, if a company has spent time manually adding missing features as mentioned above, they must redo everything if the underlying application ever changes. While a vendor sells screen-scraping as a time-saving method, any time-savings initially gained from screen-scraping is lost every time the applications are altered.

Second, maintaining outdated applications is a costly proposition, in more ways than one. An article on SearchDataCenter.com entitled "Weighing the costs and risks of mainframe application modernization," breaks up the costs into three parts:

- **Maintenance costs.** Outdated applications are more expensive to maintain as software and hardware is no longer adequately supported.
- **Opportunity costs.** Maintenance spending crowds out new application development and other projects.

- **Inefficiency costs.** Costs incurred as the failure to proactively upgrade causes crisis mode, costly application fixes, periodic directives to move to a new platform, and wasted time and effort on flawed or failed major software improvement projects.

The article goes on to explain how these costs turn into a never-ending cycle that leads to even more problems:

"These costs often feed on themselves in a vicious circle. Choosing to maintain instead of improve an application means that the application continues to age, thus not only increasing maintenance, opportunity and inefficiency costs, but also increasing the gap between the legacy application and current technologies. This age gap in turn makes improvement more costly, which makes the organization more likely to choose the as-is strategy."

Essentially, the article highlights a common problem among companies that choose screen-scraping over modernization. They invest significant time and money into creating screen-scraped applications and maintaining the underlying applications. Meanwhile, they become further and further behind technologically. In a few years, they face this problem: Although their applications look modern, their outdated architecture can no longer support their business. The company can't afford to fix the problem because they've wasted so many resources maintaining old technology and applications. This results in what the article describes as "crisis mode."

Third, maintaining outdated applications requires a very specific, yet dwindling, skill-set. Considering that outdated applications may have been written up to 30 years ago, whoever wrote them initially has probably moved on to another company or retired. Finding someone with the appropriate skills to maintain these applications becomes increasingly difficult with time.

Finally, maintenance costs don't end with software. Supporting outdated applications requires the support of outdated hardware as well. Old applications were designed for use on hardware available at the time they were written. As long as the underlying applications are still outdated, they are tied to old hardware. Companies who want to move to newer, more efficient hardware may be handcuffed by their old applications. Instead, they are forced to support more hardware than necessary, and miss out on any advantages that come with new hardware.

Problem 4: Screen-scraping can't address business changes#

To this point, we've covered how screen-scraping addresses technology changes, namely, screen-scraping makes old applications look modern. But, modernization is more than updating technology. Modernization is about changing applications to conform to modern standards, both from a technology and a business perspective.

Over time, all businesses change in one way or another. For example, government regulations and taxes might change, affecting the way a business operates. Additionally, with the rise of the internet, many businesses changed how they sell (online), when they operate (24x7), and who they sell too (Anybody).

Application modernization must leave a business with applications that apply to their current business needs. Since screen-scraping only makes old applications look modern, it cannot address the current needs of a business. Instead, screen-scraped applications will address the business needs of whenever the underlying applications were built. Can a company afford to run on 20 year old technology built for 20 year old business needs?

Additionally, screen-scraping does not allow for business changes in the future, as they do not grow and change with a business. Rather, as previously mentioned, the underlying application would need to be updated, which would require re-scraping and re-modifying of applications.

Conclusion

Vendors sell screen-scraping as a cheap and easy modernization method, but unwitting companies won't realize the drawbacks until it's too late. Yes, screen-scraping is cheap and provides modern-looking applications quickly. However, screen-scraped applications are not actually modern. They lack common features found in modern applications which can only be added with considerable effort.

Additionally, screen-scraping raises maintenance complexity and cost. Because the underlying applications still exist, they must be maintained. Because the screen-scraped applications are merely modern representations of the outdated applications, they must be re-scraped every time the underlying application changes.

Furthermore, since the underlying applications were designed so long ago, they are often tied to old hardware. Companies that choose screen-scraping must maintain and support old hardware and old software designed to address outdated company needs.

While screen-scraping is advertised as fast and cheap, the end product lacks common features, ties companies to old hardware, won't address the changing business needs, and is difficult to maintain. In the end, screen-scraping requires a considerable investment of resources but leaves companies worse off than when they started.