



The MSIX Report Card

21H2 Edition

Timothy Mangan,
TMurgent Technologies, LLP
January 2022

MSIX Progress Report Card

Student Name: Msix		Grade: 4	
School Year: 2021		Days Attended: 365	
Teacher Name: Timothy Mangan		Absent: 	
Principal Name: TMurgent Technologies, LLP		School Name: Working Tech	
Teacher Signature: 		Principal Signature: 	
School Address: 		School Address: 	


Course	Level	Credits
ISV Partners	RE	1
Packaging Tools	RE	1
Tools for Developers	RE	1
Tools for ITPro Repackaging	RE	1
Tools for Deployment	RE	1
Runtime	RE	1
Total Credits:		5

Course	Level	Credits
ISV Partners	RE	1
Packaging Tools	RE	1
Tools for Developers	RE	1
Tools for ITPro Repackaging	RE	1
Tools for Deployment	RE	1
Runtime	RE	1
Total Credits:		5

1st Semester Grade	2nd Semester Grade	Final Grade (only if using number values for semester grades)
C	B-	B-
B+	B+	B+
C+	B+	B+
B+	A	A
A	B+	B+
B	A	A
	B+	B+

Credit Key	Credits
1 semester course	5 credits
2 semester course	1 credit

Grade	Grading Scale Key	Low %	High %
A		0.9	
B		0.8	
C		0.7	
D		0.6	
F		0.5	



Comments

Starting to show some attention. Small improvements. In attendance. Good results with less work. Participates in lots of activities. Small improvements, will need Win11 for some.

Introduction

Nearly four years ago Microsoft announced MSIX, a three headed effort that intends to be:

- A replacement for MSI based application delivery by software vendors.
- Tooling to help IT Professionals repackage existing software into MSIX. This involves both Microsoft developed tooling and tooling from third party vendors that are already in the repackaging business.
- And a runtime environment that protects applications and the operating system by using an updated container similar to that used previously for UWP programs.

Since that time Microsoft also introduced a fourth head, Project Reunion (now known as the Windows App SDK) for developers as the app development platform for the future. While not explicitly tied to MSIX, this move is a huge part of the carrot to get developers to release in MSIX format.

In the original announcements Microsoft indicated that the road to MSIX would be a journey and it would take several release cycles to complete all the functionality needed. We are still on that road and are likely to be for some time. Nevertheless, there are some great things that we can do today.

This 2022 version of *The Report Card* is an update to our prior views; to review previous versions see:

- <http://m6conf.com/index.php/reportcard/46-report-card-2019>
- <http://m6conf.com/index.php/reportcard/46-report-card-2020> and
- <http://m6conf.com/index.php/reportcard/46-report-card-2021>

We can summarize the changes this year as follows:

- Microsoft is starting to make significant headway with the developer community, both in delivering some of what they need and in interest level from that community.
- The Package Support Framework has stabilized in a new home, and expansions to the types of apps supported has started.
- Long awaited capabilities to the runtime are now available, although Windows 11 might be a requirement.
- Tools for repackaging have focused more on making it easier to package apps, especially when the PSF is needed.
- Overall, application compatibility for repackaging of existing apps makes gains this year even through the focus this year had been on making it easier to repackage the ones that work.
- More vendors have entered the Enterprise IT Pro support arena for MSIX, providing more options to prepare and/or deliver MSIX Apps. MSIX App Attach is also becoming more mature.

My intent is to update this report card in January of the following years so that we can judge the progress. There are undoubtedly many other vendors that are active in this space and yet not included in this list due to my limited resources and/or unfamiliarity with their offerings. If you are one of these companies and are supporting MSIX, I apologize for the slight. Please contact me to ensure that I include you in the future.

And here is this year's report card...

1 Support by Software Vendors to release in MSIX format

C- Microsoft's Independent Software Vendor (ISV) Partners, who build end-user applications for the Enterprise, improved from an "D" (as in "Incomplete") to a "C-" for releasing software products in an MSIX format.

This mark is not a criticism of the vendors. They need tools to succeed; some of those became available late this year, but more maturity in that tooling is needed for anyone other than bleeding edge types. This year, we see evidence that the ISVs are finally interested and many are kicking tires.

Vendors require three things:

- A reason to change what they have been doing for years.
- A clear market.
- And a set of tools that help them succeed.

The reason

Microsoft must continue to hammer home their answers for the first item to the developers. The message has been fine, it just must be repeated over and over to them.

The market

The market aspect has become much more clear with the passage of time. While MSI versions for those reluctant Windows 7 consumers still need to exist, everyone has moved on to Windows 10 (and now 11).

The market is there if and when software vendors can make it there.

The tools

The release of the first version of the Windows App SDK (formerly known as the delayed Project Reunion) this fall is a major step in the tooling. This, combined with better project support in tools like Visual Studio, make it possible for a developer to create these new modern "native MSIX" apps.

But like many first version products, this first release will likely prove insufficient for the needs of most software vendors. Many consider version 1.1 (expected by the end of 2022) to be necessary, however it might take until the following release cycle to have the capabilities for mainstream application building.

The early versions may prove to be better suited for vendors ready for greenfield projects rather than converting their existing products. It will be interesting to see what comes out this year.

The current state

The chat in developer forums around MSIX is picking up.

There have been a few of well known products come out with MSIX releases in 2021. These appear to have been completed without the new Windows App SDK.

For vendors interested in moving forward while waiting for the SDK to be more viable, a free community book was released in December to help [A Developer's Guide to MSIX \(reverera.com\)](https://www.reverera.com/2021/12/01/a-developer-s-guide-to-msix/). Written by Tim Mangan and developers from Reverera and Flexera, it provides a cookbook for how to move your existing product from MSI to MSIX while keeping your code focused on the SDKs and APIs you use today.

Microsoft, for their part, have been quietly moving more of their own apps into the MSIX world. Notably absent is Office, but we all expect that to take time to do it right.

The bottom line.

This category continues to be the most important part of MSIX success. If the vendors don't go for it, all is lost for MSIX.

It took mainstream vendors about 5 years to fully embrace the .Net Framework when it was released around 20 years ago. The new SDK is now out. This will take time.

2 Support by Tooling Vendors

B+ Tooling Vendors stayed steady at “B+” this year. The established vendors have been very active in the MSIX community from the beginning, but more and more companies are joining in as well.

This category is further broken into three sub-categories, and I will grade each of these independently as well. The category includes:

- Tooling for developers.
- Tooling for IT Pros in repackaging.
- Tooling for distribution.

Many of the players are involved in multiple of these subcategories, and Microsoft themselves are also a first party vendor in this space as well.

But as it is an evolving technology, without a well-defined public roadmap, it is difficult for third parties to judge when and how much effort to put into MSIX. There is a very real possibility that work done to have product today will have to be reworked or discarded as Microsoft makes changes to the runtime in the future.

As usual, Microsoft is tight lipped about their strategy regarding MSIX. My take is that Microsoft is unlikely to do a lot more for IT Pros in the tooling space than they have to date, more likely to rely on the third parties to help instead. While disappointing, if this means that Microsoft spends more development dollars to support the ISV developers, I will agree with such a strategy.

2.1 Tooling Vendor Support for Developers

B Tooling Vendors supporting Developers earned an improved “B” this year. Much more work will be needed this year as supporting the Windows App SDK and improved WAP projects require retooling by these vendors.

Ultimately, the new SDK and new tools will lead to the creation of different “Patterns and Practices” than developers are used to. We really don’t even know what those will be yet, but over the next few years we will figure that out.

2.2 Tooling Vendor Support for Repackaging

B+ Software Vendors supporting IT Pro Repackaging of applications into MSIX earned a “B+” this year. This year we saw a bigger emphasis in making it easier to repackage the apps that can work rather than focusing on more progress in application compatibility.

Microsoft made improvement in the first half of the year with the [Microsoft MSIX Packaging Tool](#) (MMPT), however the releases later in the year created numerous new issues and we generally recommend staying with the older 2021.709 version used in this testing.

Update: *Version 2022.110 of the MMPT was released after completion of the Compatibility testing shown later in this paper. It looks to be the recommended version going forward and may have improved the results of the Compatibility test results documented here.*

Meanwhile, the [Package Support Framework](#) (PSF) was under-maintained and as OS changes and time moved on it became less capable. I believe this has now been addressed around the end of the year in the Tim Mangan fork and this seems to be the version of the PSF with the most interest and promise going forward.

[PsfTooling](#), a free app in the Microsoft Store that I created to be used with the MMPT to inject and configure the PSF components. The tool has improved over the years and can help someone detect and add the PSF with much less work than before.

Other packaging vendors also include the PSF in their own repackaging products, combining the equivalent of PsfTooling and the MMPT into a single experience. Some appear to be following my fork of the PSF to improve their products.

2.3 Tooling for Distribution

A Software Vendors with support for MSIX Distribution at a solid “A”. I’d give it an A+, but until we have production ready packages to distribute, most customers are not trying these out.

Microsoft themselves have support for MSIX in the Microsoft Store (the “Consumer Store”), the Microsoft Store for Business, Intune/MEM, and good old Configuration Manager. Additionally, we can use the same PowerShell commands used to install and uninstall AppX (UWP based) and Windows Bridges (Centennial based) programs.

On top of that we now have some standard support to help vendors distribute MSIX packages from their own websites instead of relying on the Microsoft store via AppInstaller files. This new support leverages the embedded update processes inside of MSIX, allowing the vendor update from their own websites using this built-in capability and to then drop the auto-update code that they have been building into their software in the past. These vendors will need to also support download of the MSIX without the update from their sites as that remains a requirement of many large enterprises. Unfortunately, Microsoft temporarily turned off a key component for AppInstaller file based deployments in January of 2022. We are awaiting a resolution (and explanation).

A long-awaited *MSIX App Attach* finally achieved GA status this year as a feature for Azure Virtual Desktops. Several vendors are using the App Attach techniques to provide fast delivery for non-persistent and semi-persistent scenarios outside of Azure Virtual Desktops.

We have also seen some other players enter the MSIX space for supporting Enterprise IT, both in the app preparation app distribution arenas. It is great to see their energies and ideas added to this space.

3 MSIX Runtime Support

B+

The MSIX Runtime moved up to a “B+” grade this year, but it was a tough call. Microsoft continued to make improvements, although some only appear in Windows 11. But customers have been moving on from now unsupported OS versions to newer versions, even if they aren’t ready to roll out Windows 11 to everyone yet. And each time they do they are getting a better runtime that gives them the ability to deploy more apps as MSIX.

The significant improvements to the MSIX runtime that I noted this year include:

- *Shared Package Containers.* Again, currently requires Windows 11. It seems like we have been waiting for this for 2 years now. I am glad it is here, but I don’t think I’ll replace my use of Modification Packages with it. I also don’t need it yet, as I need those big apps that are the hardest to get to work under MSIX to work properly by themselves first. Eventually this will be huge, for example should Office come out.
- *Support for Certain Context Menus.* This currently requires Windows 11, but some of the apps with styles of context menus not previously supported now work.
- *Other Shell Extension support.* Some of these appeared in 21H1 and some in 21H2. This adds things like Property Data and Preview handler support.

To date, Microsoft continues to be pretty quiet about roadmaps for MSIX support and features, and often even when they add support for a new feature we must “discover” the changes ourselves. This is unfortunate, but I don’t expect that to change.

4 Repackaging Testing

At the end of the day, we need to be able to deploy the apps. Without a lot of vendor supplied apps available, we can work the path of repackaging existing applications into MSIX and testing those. In this section, as in years past, I'll document the result of the testing. The experience with this testing was crucial to, and had significant impact up, the grading earlier in this report.

This year, as part of the *Report Card*, I expanded the set of applications to be tested in a repackaging scenario to 74 to improve coverage. Most of these applications were applications that we commonly see Enterprises distributing to employee workstations today. A few "special purpose" applications that I commonly use to demonstrate application integration capabilities were also included to ensure that we are covering the needs of most apps. Not included in the application mix were applications that are known to not work under any virtualization or container, such as App-V and MSIX – for example those with device drivers or plug-ins for Internet Explorer. Also excluded were problematic "heavy" applications like AutoCAD and ArcGIS that are always difficult to deploy. The applications that were added into the test suite this year were all apps that would probably have failed in past years.

These applications were repackaged and tested three times:

- Using Microsoft App-V.
- Using the Microsoft MSIX Packaging Tool (2021.709 release) without the PSF.
- Using the Microsoft MSIX Packaging Tool (2021.709 release) with TMEditX 2.0.0.0.

Although there were newer versions of the MSIX Packaging tool available, the 709 release was used as it produced the best results. PsfTooling was not used to add the PSF as an updated version was not available in time for the testing. When version 5 of that tool is released, I would expect that testing would produce similar results with some apps having minor features that it is not designed to fix.

All testing was performed on Windows 10 21H2. Although Windows 11 21H2 was also available for testing, it was felt that such testing would only produce slightly better results. In addition, due to the hardware incompatibilities, it will be some time before enterprises are able to flush out all of their Windows 10 devices.

For each application, I categorized the results of testing into one of five buckets:

- **No Packaging Workflow.** The tooling does not yet support a way to package the application. Currently, we no longer have packages in this category.
- **Does Not Package.** Using all the tricks that I am aware of, I could not get the app to produce a MSIX package file. This category includes situation where an MSIX file would get created but it could not be signed by signtool.
- **Failed Installation.** A package was created and signed, but AppInstaller refuses to install it.
- **Failed Smoke Test.** A smoke test is nothing more than installing the package and trying to see if it installs and the primary application shortcut can be launched. For apps that are simple, the primary use of the app might also be tested. Passing the smoke test does not mean that the application is production ready, only that it is good enough for full acceptance testing.
- **Failed Feature Test.** The acceptance test showed a failure that would clearly prevent an enterprise from putting this package into production if they required that feature.

- **Partial Feature.** The acceptance test showed that the most important features of the application work, but that the full fidelity of the MSI app was not available. A subjective decision was made that the lack of the feature(s) would keep most enterprises from releasing this package into production, but that some might. In addition to features not, and app can fall into this bucket due to the inability to disable updaters or application licensing challenges.
- **Full Feature.** While not every feature was necessarily tested, the Acceptance test indicated that it is highly likely the package could be put into production.

While every attempt is made to be objective in testing, the interpretation of test outcomes nevertheless is subjective. Someone else testing the same packages might categorize the results into different buckets than me (especially between the last two buckets). But we must start somewhere!

In the charts that follow, color coding is used to signify the categorization of the testing result. The following table should be used to interpret those colors:

Legend

Category	Description
Untested	Incomplete. Most likely this indicates incomplete testing. There are cases where something might be blocking the test, or perhaps you are viewing the results at a time while testing is in progress.
No Workflow	Tooling vendor has no existing reasonable workflow for producing this package. For example, Add-on packages and Modification packages might not be possible with a vendor at this time.
Failed Packaging	The tooling failed to generate a signed package MSIX file at all. This usually indicates an issue with the capture process or in package formatting.
Failed Installing	A package was generated, but it fails to install/deploy. This likely indicates a problem in the package format and contents.
Failed Smoke Test	A package file was generated, but the package failed the primary smoke test, indicating a complete or major functional loss.
Failed Feature Test	The package works to some extent, but failed a major feature and would not be acceptable.
Partial Feature Issue(s)	The package is lacking in one or more minor features that might prevent production deployment at some or most customers.
Success	The package passed all tests. The package would be suitable to send to final acceptance testing and/or piloting.

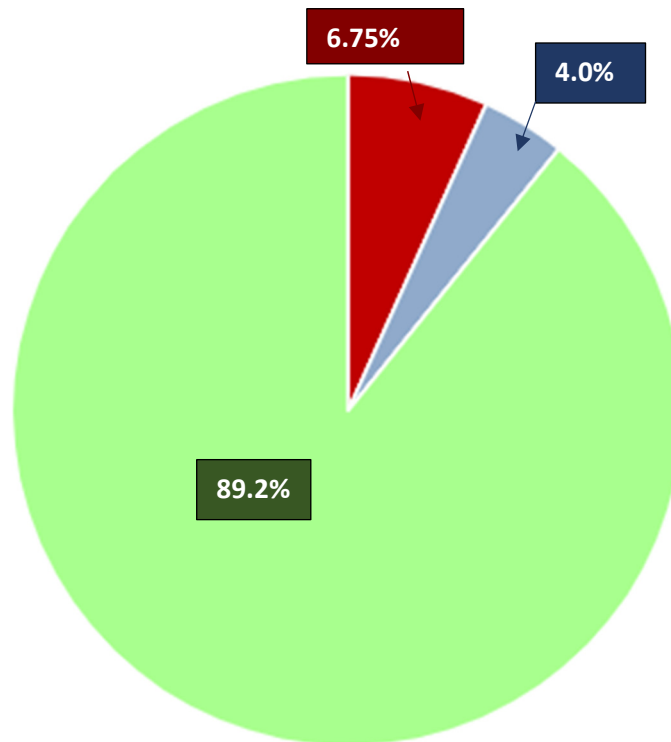
Finally, it is worth noting that the testing performed was using recapturing techniques. There may be ways to build a MSIX package using traditional install builder tools from these same vendors that have been extended to support MSIX, however that was outside the scope of this testing.

4.1 Comparative results using Microsoft App-V

Many of the applications in the list are older applications that are challenging to deploy in today's environments. Indeed, although I did not categorize the list using Native installation techniques, the results would be far less than for repackaging in App-V. This is after all why we have App-V!

The 74 packages were packaged on Windows 10 21H2 and tested on the same OS. The 2004 ADK Sequencer was used and packages were not further fixed up by TMEdit which would have improved the results.

Result summary for 74 packages using Microsoft App-V from 2004 ADK Sequencer and Win10 21H2



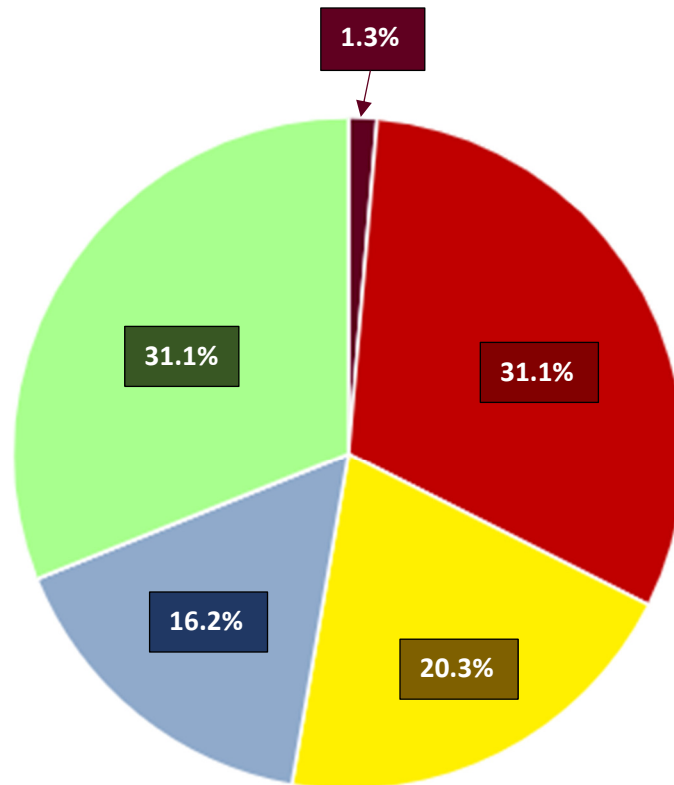
Not surprisingly, these results were very good using seasoned tooling and collective wisdom on techniques to package with App-V.

The testing value for high-fidelity dropped from 95% last year to 89.2% due to the extra applications added to the mix this year.

4.2 Results from Packaging with Microsoft MSIX Packaging Tool

The same 74 applications were packaged using the 2021.709 release of the MMPT. These tests use current best practices for packaging, without the use of additional tooling or extraordinary measures such as manual manifest editing.

Result summary for 74 packages using MMPT 2022.709 on Win10 21H2



The results are about the same as those from last year, due to a combination of improved runtime support and the increased difficulty of the application mix. Customers are cautioned that the results will be lower on back-rev Operating System versions due to the runtime improvements of the tested OS.

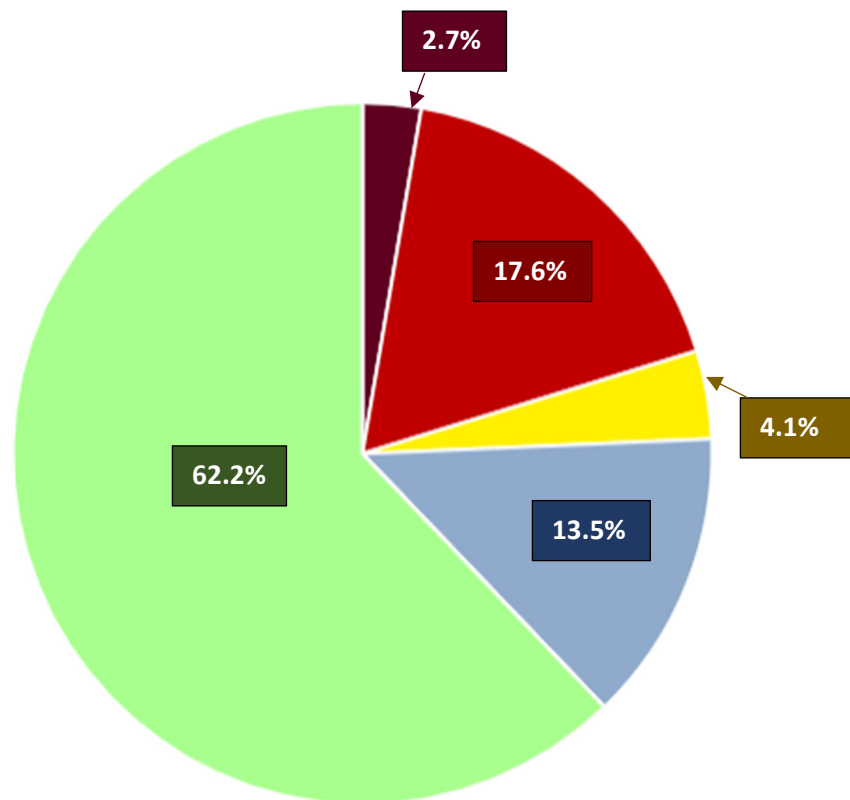
Except for a few cases, these packages were fully automated using remote package capture techniques. The reason that a few of the apps were manually packaged was that the disabled Windows Updates were re-enabled by the OS during packaging causing bad packages.

4.3 Results from Packaging with Microsoft MSIX Packaging Tool and with PSF

For these tests, the MMPT version 2021.709 was also used, but the packaging process was enhanced by using TMEditX 2.0.0.0 to inject and configure the Package Support Framework (PSF) into the package as well as make additional improvements available in MSIX but not through the MMPT. (Information on TMEditX is available <https://www.tmurgent.com/AppV/en/buy/tmeditx/tmeditx-about>) .

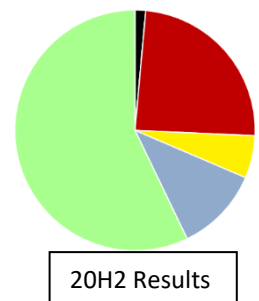
For the most part, TMEditX was used via automation as well. In about 20% of the packages it was necessary to manually run TMEditX due to conflicts that can occur with some of the shell extensions. This version of the PSF used contains a build of the latest PSF source code from GitHub available as of the end of January 2022. The same set of 74 Apps were tested.

Result summary for 74 packages using MMPT 2021.709 and TMEditX 2.0 on Win10 21H2



Even with the more difficult application mix, the high-fidelity result managed to pop above the 60% mark for the first time! But more important is how many of those applications were very close to getting that mark, with only a minor nit keeping them out of the green. Meanwhile the categories of unusable apps dropped from 25% to under 20%.

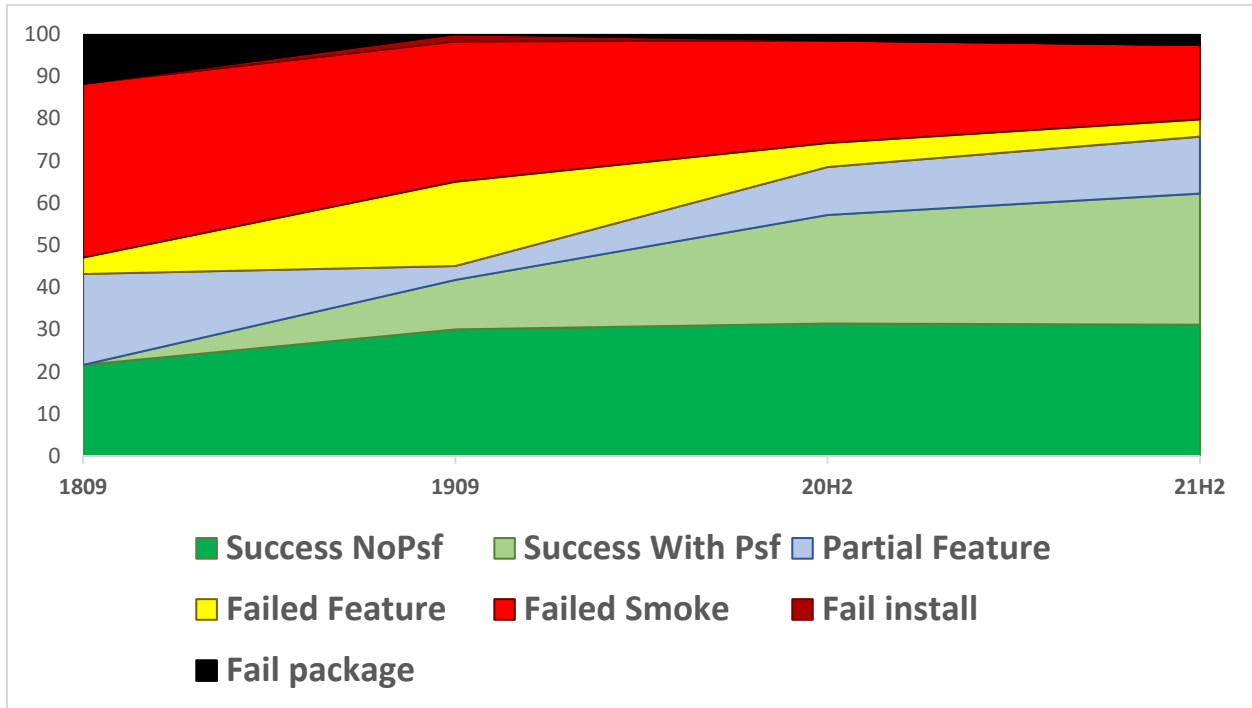
But more striking was the amount of effort that it took to produce those packages. Even including the time spent on setting up the automation, the packaging effort in this year's test was less than half what it was last year.



4.4 The trend

Now that we are in the fourth year of this testing, we can look at the numbers to see how the trend has been going.

Trend of MSIX Compatibility



Overall, the trend has been improving. Because of the focus last year was on making it easier to get applications packaged and through UAT, it was not expected that the compatibility rate would improve. But changes made late in the year, both the PSF and the TMEditX tooling, allowed for the improvements seen in these results.

My focus in 2022 will be in efforts to increase the application compatibility further. While we hope to see Microsoft continue the trend to improve the runtime capabilities, there is still many apps for which the PSF needs to be extended and that will have the larger impact in results next year.

About the Author

Tim Mangan is an independent consultant and the owner of TMurgent Technologies, LLP. Recognized as an industry leader by Microsoft as an MVP for more 15 years, and by Citrix for a dozen as a CTP/Fellow, Tim is generally known as “the Godfather of App-V”, having led the effort to build the original version of App-V at Softricity.

TMurgent provides consulting and training around application and desktop deployments. Our “Packaging for App-V and MSIX” training classes are sought after by IT Professional Desktop Engineers around the world.

TMurgent Technologies, LLP is an independent company engaged in the packaging space. TMurgent primarily provides training to IT Professionals that are involved in Desktop Engineering and Application Packaging, but we also provide some free and licensed software products in this space.

Those interested in MSIX are encouraged to download one of our two community eBooks on MSIX. Each is over 200 pages and co-authored with leading community technologists for a thorough coverage:

- **For the IT Pro:** [MSIX Packaging Fundamentals \(tmurgent.com\)](https://tmurgent.com)
- **For the Developer:** [A Developer's Guide To MSIX \(tmurgent.com\)](https://tmurgent.com)

As an independent contractor, TMurgent may relationships with several of the vendors, some of which provide support. Despite this, we believe that the information in this report does provide a fair and independent view that represents the state of the industry currently.

This report contains information gained from personal experiences and may not represent the best that can be said about the vendors and products mentioned. Omissions and mistakes are my own, but they are “honest” mistakes and not intended to malign. The Vendors have not been given an opportunity to review or correct potential factual errors in this report prior to publishing, however they may contact me if they feel there are errors in the report that should be addressed.