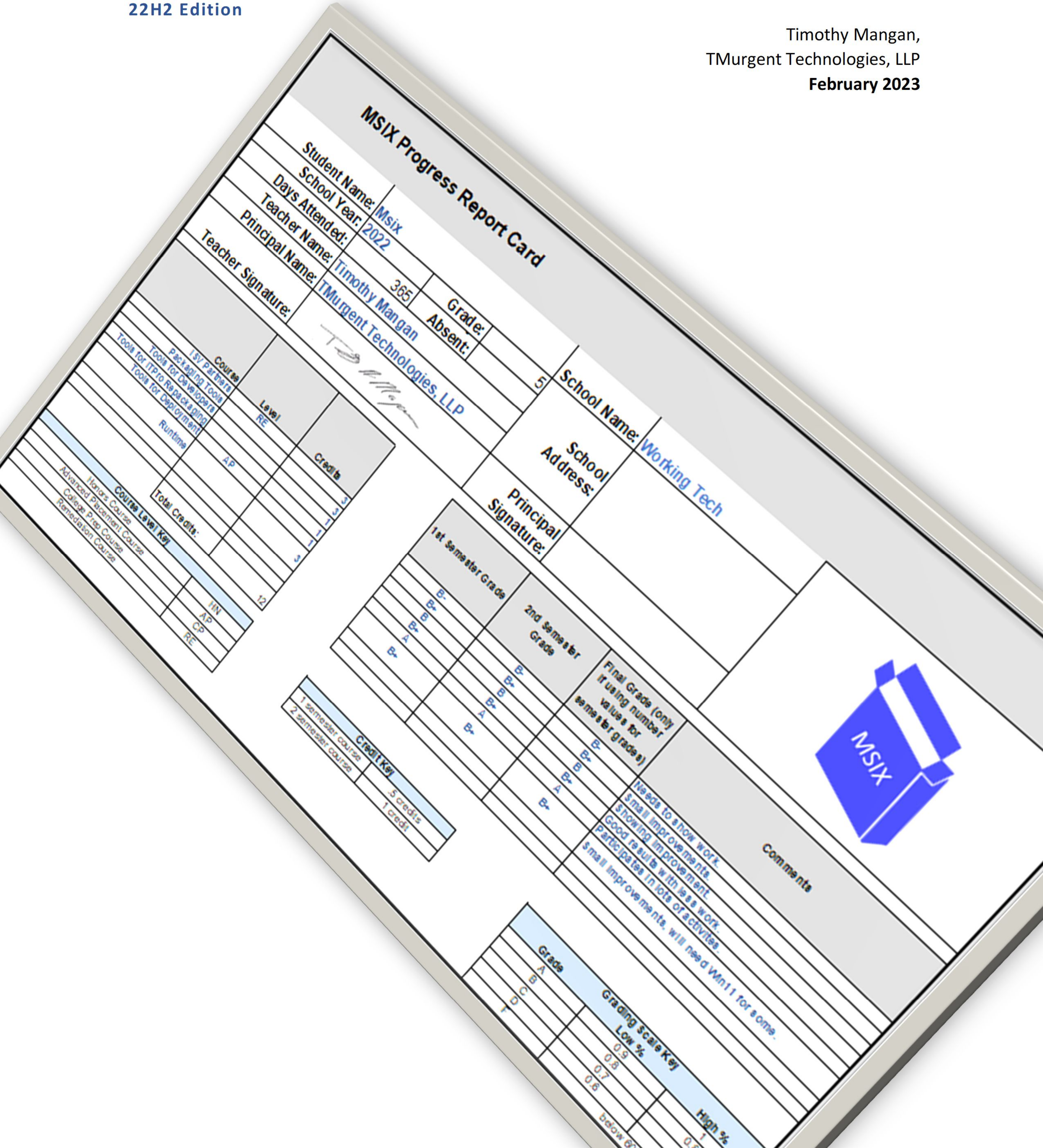
The MSIX Report Card

22H2 Edition

Timothy Mangan,
TMurgent Technologies, LLP
February 2023



MSIX Progress Report Card

Student Name: Msix
School Year: 2022
Days Attended: 365
Teacher Name: Timothy Mangan
Principal Name: TMurgent Technologies, LLP
Teacher Signature:
Grade:
Absent: 5

School Name: Working Tech
School Address:
Principal Signature:

Course	Level	Credits	1st Semester Grade	2nd Semester Grade	Final Grade (only if using number values for semester grades)	Comments
ISV Partners	RE	3	B-	B	B-	Needs to show work.
Packaging Tools		3	B+	B+	B+	small improvements.
Tools for Developers		1	B	B	B	Showing improvement.
Tools for ITPro Repackaging		1	B+	B+	B+	Good results with less work.
Tools for Deployment		1	A	A	A	Participates in lots of activites.
Runtime	AP	3	B+	B+	B+	small improvements, will need Win11 for some.
Total Credits:		12				

Course Level Key	
Honors Course	HN
Advanced Placement Course	AP
College Prep Course	CP
Remediation Course	RE

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

Grading Scale Key		
Grade	Low %	High %
A	0.9	1
B	0.8	0.8
C	0.7	
D	0.6	
F	below 60	

Introduction

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

- A replacement for MSI based application delivery by software vendors.
- A new SDK for developers to use. The Windows App SDK (formerly known as Project Reunion) helps the developer to write natural code that works inside the MSIX container, or now also possibly outside of the container as desired.
- 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.

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. In recent years customers have become outspoken about the issues they encounter and question just how committed Microsoft is to this path.

This 2023 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
- http://m6conf.com/index.php/reportcard/46-report-card-2021 and
- http://m6conf.com/index.php/reportcard/46-report-card-2022

We can summarize the changes this year as follows:

- We have made some headway addressing issues affecting the compatibility of traditional applications repackaged into MSIX. Much of that is due to out own efforts in the PSF, but Microsoft has also made some platform changes that we are just learning how to leverage.
- Microsoft has made significant headway with the developer community, both in delivering some of what they need and in interest level from that community. However, at the same time they confused the developer community with mixed messages as they added non-packaged capabilities to the SDK and tooling.
- The Package Support Framework (PSF) has thrived in it's new home at TMurgent, with a newly written fixup that leverages past lessons and new platform capabilities, expanding the types of apps supported.
- After ignoring their own repository for the PSF for over 3 years, Microsoft finally integrated prior accepted contributions into the main branch of the repository, and is offering it as part of their tooling. This, along with other MMPT tooling changes appear to be in response to the questions customers are asking about commitment.
- Tools for repackaging from numerous vendors have focused more on making it easier to package apps, especially when the PSF is needed.
- MSIX App Attach is continuing to be a delivery vehicle for certain scenarios, with Vendors like VMWare, Citrix, and others adding support. Limitations to the underlying technology (more

limited than MSIX in general) continue to limit the appetite for App Attach, with customers in those scenarios continuing to leverage Microsoft App-V, or other “App Layering” solutions.

My intent is to update this report card in (roughly) 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

B-

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

A small number of vendors have released versions of their software in MSIX format. Nobody seems to really track how many, but products like VLC Player, Blender, Firefox, and others have solid releases available. Additionally, looking at the developer forums there is a significant uptick in questions about how to get their product to work in the container.

In the past, I said vendors require three things:

- *A reason to change what they have been doing for years*. The Windows SDK is the reason.
- *A clear market*. MSIX is available on all currently supported platforms. Consumers (the only ones still using Windows 7) that might be still running Windows 7 aren't likely to buy much new software anyway.
- *And a set of tools that help them succeed*. There is an SDK, and there are tools. But Microsoft has more work to do here. I see the vendors trying and being frustrated.

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.

Mixed messages by Microsoft on their commitment to getting apps to run inside containers must be cleaned up, and additional help for software vendors provided.

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 we hear concerns from those involved from in supporting MSIX from the beginning regarding Microsoft’s level of commitment.

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.

2.1 Tooling Vendor Support for Developers

B+ Microsoft has been doing a lot for developers, not only in improving the SDK, but advanced in Visual Studio itself.

The SDK is maturing and supporting more APIs that developers require and were initially missing. Much more is to be done, but Microsoft has been delivering on their promises.

Visual Studio improvements have been outstanding. The IntelleCode feature has become more AI-like, not just offering corrections afterwards, but anticipating what the developer is about to type based on existing code patterns and content. It even can now suggest better ways for the developer to write their code, utilizing new programming language extensions that the developer may be unaware of. Increasing developer productivity will drive developers into these new technologies and Microsoft seems to be on the right path here.

2.2 Tooling Vendor Support for Repackaging

B+ Software Vendors supporting IT Pro Repackaging of applications into MSIX earned a “B+” again this year.

Microsoft continues to invest in their MSIX Packaging tool. While there was a bad release last fall, they are adding in new capabilities, including advances on the filtering of unwanted file/registry changes, the conversion of “portable apps”, and even offering a form of the PSF integrated into the tool. Unfortunately the PSF included is about 1.5 years behind what could be available right now so it might not be the best way to fix up the packages.

Third party tools for repackaging have continued to advance. Some pull from the TMurgent fork of the PSF, some from Microsoft, and some from their own fork. Most have been focusing lately on making it easier to repackage and fix.

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.

Meanwhile, the Package Support Framework (PSF) in the Tim Mangan fork has advanced the game with the usual improvements, plus a very significant re-write of the FileRedirectionFixup called MfrFixup. This replacement fixup consolidates all that we have learned about the file needs of applications over the last 5 years into a more capable, simpler to use, and maintainable, version. The goal initially has been to get the MfrFixup initially to be on par with the FRF that it is replacing, but with the inclusion of Ilv-Aware mode (InstalledLocationVirtualization) we already are seeing the advantages

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. It includes support for the new MfrFixup, however because the Microsoft Management Tool creates the AppXManifest, manual editing of the AppXManifest in that tool is required to take maximal advantage.

TMEditx, a paid for app by TMurgent, has expanded the types of detections of application installer intent and fixups to the package, including the new MftFixup and InstalledLocationVirtualization. With more of the core OS changes for MSIX made by Microsoft in the past now part of all supported OS versions, the tool has also started incorporating those as well.

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. Seemingly all are doing more to make it easier for the people doing the repackaging to be successful, and all in quite different ways.   One possible way for a repackager to get the ultimate best level of compatibility may be to try difficult apps utilizing multiple packaging tool vendors.

2.3 Tooling for Distribution

A Software Vendors with support for MSIX Distribution at a solid "A" again this year.

The tools for distribution have been slowly adapting. Microsoft has revamped the hybrid MEM/EMS approach into a full blown Intune solution utilizing WinGet for delivery. While still a work-in-progress we have high hopes for this approach for persistent desktops in the cloud or on-prem.

VMWare and Citrix have added MSIX distribution support to their products this year, and other emerging vendors are making inroads as well.

3 MSIX Runtime Support

B+

The MSIX Runtime remains at a "B+" grade this year. On one hand, we haven't seen a lot that is new this year. But the other hand, some of the work they did previously that we could not use is now coming on line.

The primary reason that we did not use some of the core runtime enhancements previously was that they only worked on newer operating systems. As time has passed on, the OS versions that did not include the support go out of support and companies generally stay on supported OS versions making these new enhancements viable for general use.

The other reason that we did not use some of these enhancements is that we didn't understand them:

- Insufficient documentation on what the features actually due is extremely common.
- Sometimes we see Microsoft make several enhancements that seem to target the same issues and it isn't clear why or which one we should use.
- Often, what documentation that does exist fails to detail OS requirements so we aren't sure what the minimum OS is.
- And then sometimes Microsoft doesn't even tell us about them and we accidentally stumble on their existence and we try playing around with them.

One of the enhancements we can now rely upon being present in supported Operating Systems is InstalledLocationVirtualization (ILV). A rather under-documented enhancement, TMurgent issued a white paper detailing research on the feature in December (Microsoft has since indicated that they will provide better documentation on the enhancement). The feature is a partial answer to the request by many to let MSIX work more like App-V, where package files may be modified in a controlled way. In testing we found the feature to work well, but only if the application made file calls in certain ways. But pairing the ILV with the new MfrFixup in Ilv-Aware mode seems to provide a significant coverage for application file-based issues. There are still problems to be solved in this space, but this looks to be a giant step forward in getting more traditional apps to work under MSIX without having to be the app developer.

Some of the newer enhancements made in the previous year are not fully embraced by the community as these work only in Windows 11. While Windows 11 is rolling out in enterprise customers, hardware requirements and lifecycles prevent a full rollout yet.

The significant improvements specific to Windows 11 include:

- *Shared Package Containers (SPC).* This is the ability to have two or more unrelated packages work together, avoiding container issues between them. We don't have a huge need for this yet; those big apps that are the best candidates are also the hardest to get to work under MSIX properly by themselves first. Eventually this will be huge, for example should Office come out. But…
  - As a potential replacement for Modification packages, testing shows SPC only works about 50% of the time as it depends on how the integration is used by the app.
  - SPC seems to be completely incompatible with InstalledLocationVirtualization.

    - Microsoft does not supply a runtime API (like AppID) to detect that SPC is in use which prevents the PSF from working correctly.
- *Support for Certain Context Menus.* Sometimes Microsoft refers to these as “classic” context menus, and most now work but we still don’t have support for all of them.
- *Possibly Other Shell Extension support.* We see work was done, but lacking documentation we believe these to also be Windows 11 only.

Microsoft could do a much better job communicating with their partners and customers than what the current documentation provides. When the goal is to allow your customers to do more, it doesn’t mean they should guess at what might work.

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 86 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. The applications that were added into the test suite this year were all apps that would have a higher probably of failing that those included in past years.

These applications were repackaged and tested three times, all on Windows 10 22H2:

- Using Microsoft App-V.
- Using the Microsoft MSIX Packaging Tool (2021.118 release) without the PSF.
- Using the Microsoft MSIX Packaging Tool (2021.118 release) with TMEditX 3.1.0.0.

Additionally, the last set of packages were retested on Windows 11 22h2. All packaging was performed on Windows 10 21H2.

PsfTooling was not tested this year, with an assumption that the results would be closer to, but not up to that of TMEditX test. Other third party packaging tools were not tested.

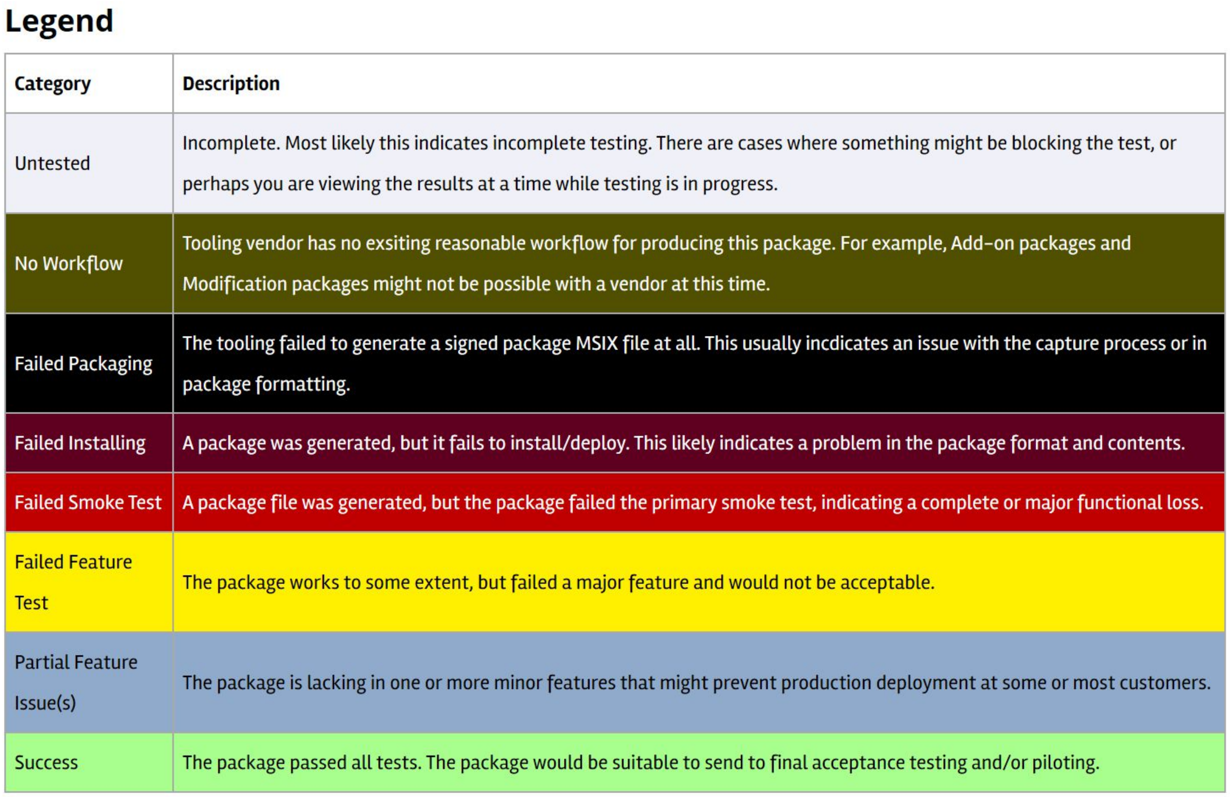
For each application, I categorized the results of testing into one of five buckets:

- **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 exsiting 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 incdicates 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.

Additionally, this year the result graphics offers an additional summarized version of the results in bar form that combines the first five categories into a failed status. This creates a simpler subjective view where the reader can apply their own subjective view of compatibility with categories of “Good”, “Maybe”, “Maybe not”, and “No way”.

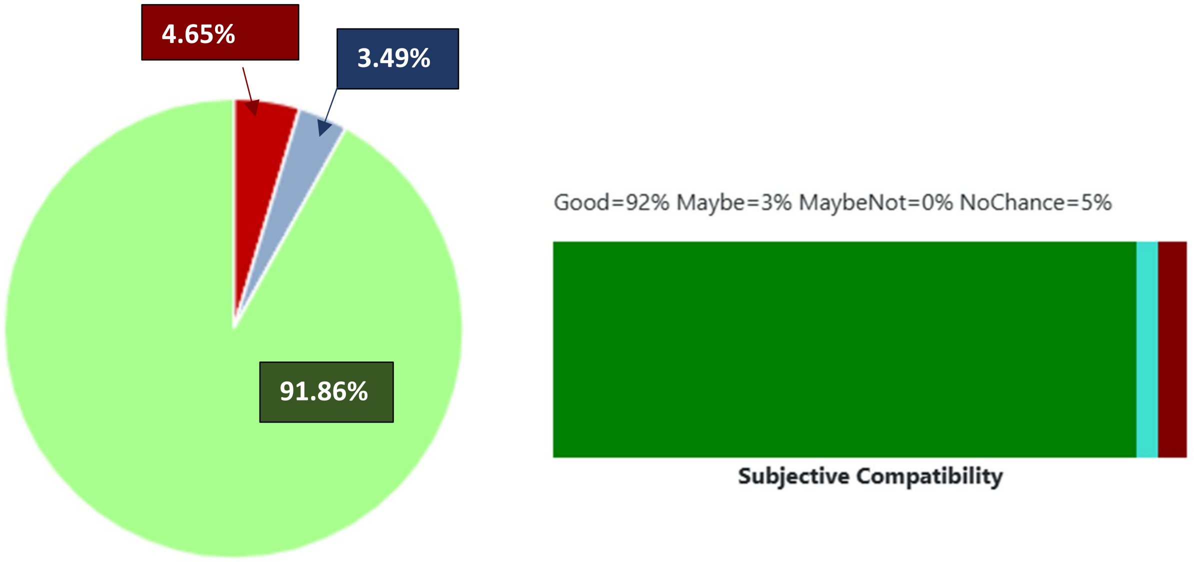
Finally, it is worth noting that the testing performed was using recapturing techniques. There may be ways to build a MSIX package using traditional install builder tools from these same vendors that have been extended to support MSIX, however that was outside the scope of this testing.

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 86 packages were packaged on Windows 10 22H2 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 86 packages using Microsoft App-V from 2004 ADK Sequencer and Win10 22H2

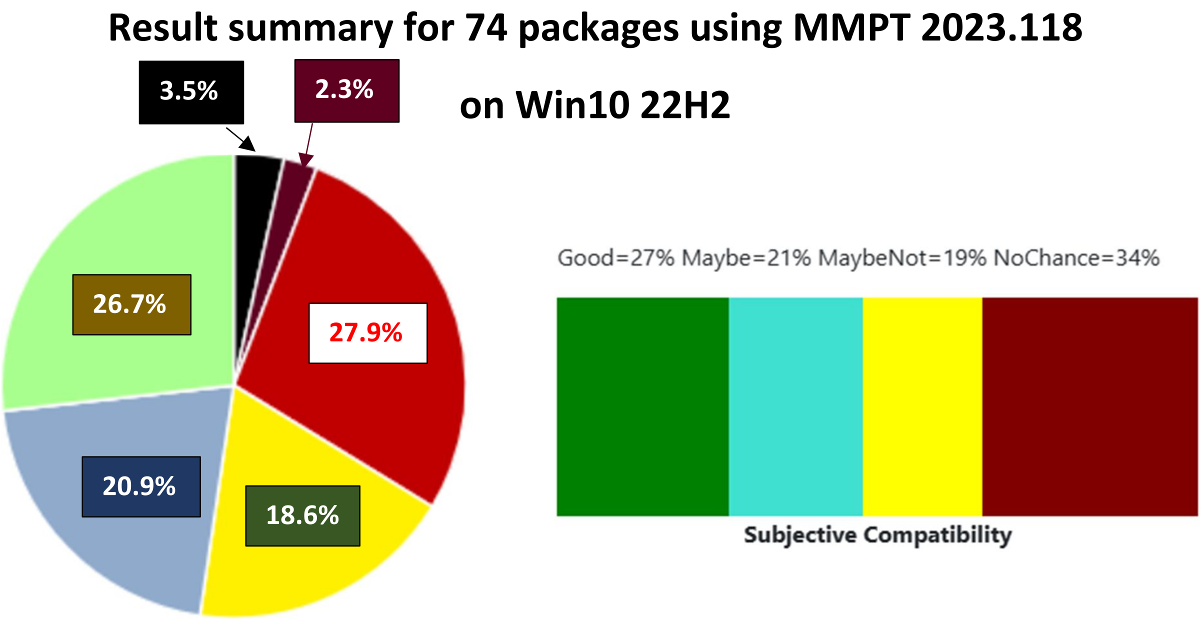


Not surprisingly, these results were very good using seasoned tooling and collective wisdom on techniques to package with App-V.

The testing value for high-fidelity increased from 89.2% last year (even with the additional applications) as more time was spent in preparing the apps for this test this year.

4.2 Results from Packaging with Microsoft MSIX Packaging Tool

The same 86 applications were packaging 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.



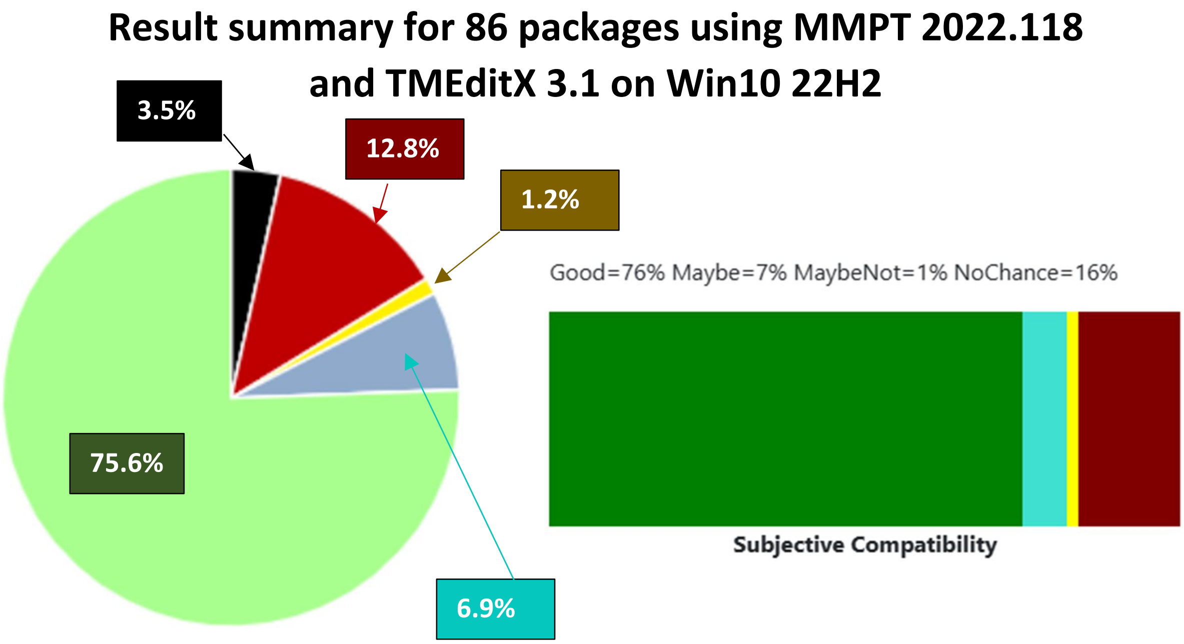
The applications that did not repackage were new applications in the test this year, or new versions of apps previously tested OK. Otherwise, I consider the results to be about the same as per last year's results for the same unchanged apps. In the past I have cautioned that the results will be lower on back-rev Operating System versions due to the runtime improvements of the tested OS, but that (mostly) no longer applies as those systems are now out of support.

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, or installers that refused to automatically install.

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

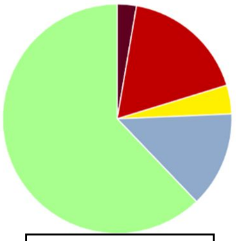
For these tests, the MMPT version 2023.118 was also used, but the packaging process was enhanced by using TMEditX 3.1.0.0 to inject and configure the Package Support Framework (PSF) into the package as well as make additional improvements available in MSIX but not through the MMPT. (Information on TMEditX is available https://www.tmurgent.com/AppV/en/buy/tmeditx/tmeditx-about) .

For the most part, TMEditX was used via automation as well. When analysis indicated the need, the new MfrFixup in Ilv-aware mode and InstalledLocationVirtualization was added. In a few of the packages the automation was scripted to add the RegLegacyFixup or make other overrides to the analysis. This version of the PSF used contains a build of the latest PSF source code from GitHub available as of the beginning of February 2023 (v.2023.2.5). The same set of 86 Apps were tested.



Even with the more difficult application mix, the high-fidelity result managed to pop above the 75% mark! 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 20% to 16%.

But more striking was the amount of effort that it took to produce those packages due to improvements in the analysis and available fixes.

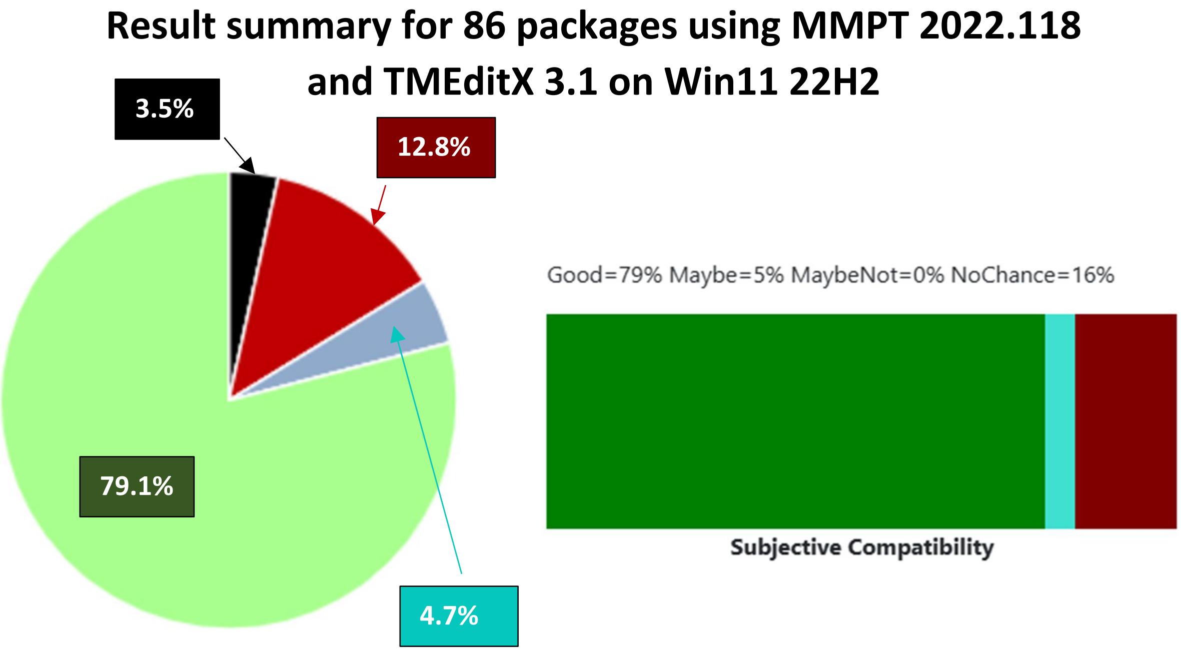


21H2 Results

automatically install.

4.4 Results from Packaging with Microsoft MSIX Packaging Tool and with PSF on Win 11

For these tests, the same packages created in the prior test, using MMPT version 2023.118 enhanced by using TMEditX 3.1.0.0, were retested on Windows 11 22H2.

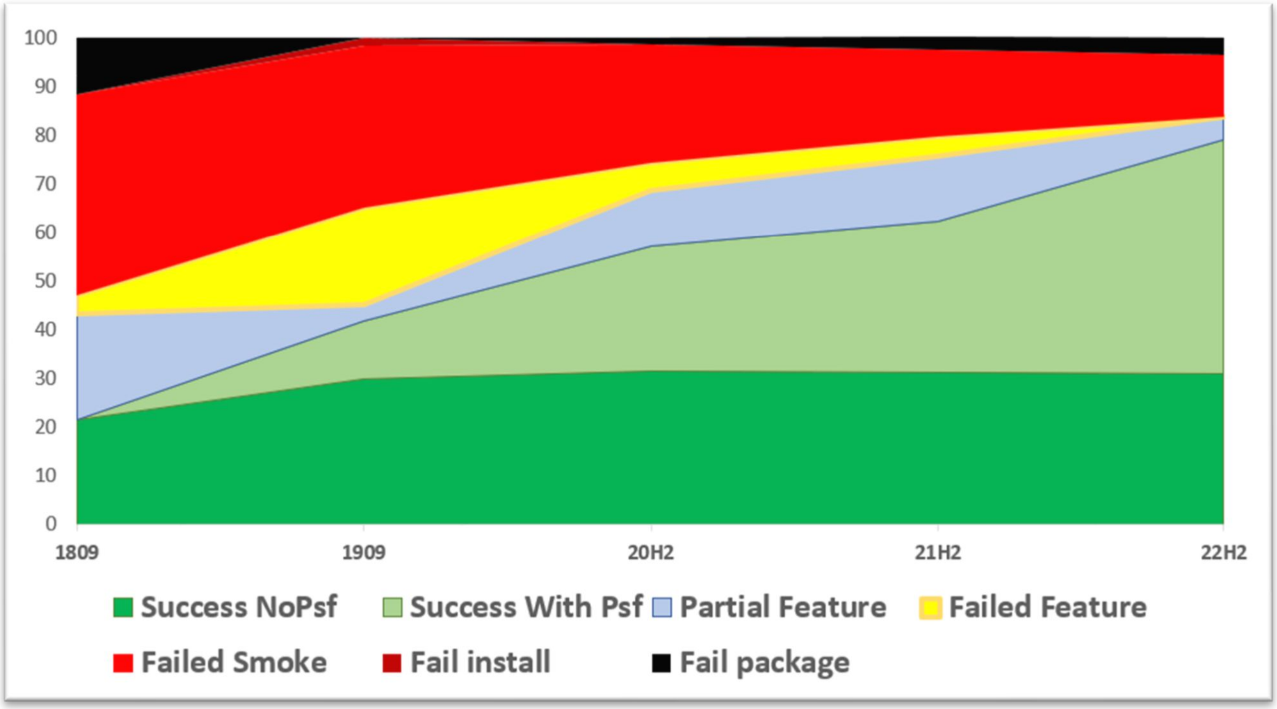


Oh so close to the 80% mark. Next year, for sure!

4.5 The trend

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

Trend of MSIX Compatibility



With the PSF continuing to be enhanced and 3rd-party tooling getting better at utilizing the core OS enhancements faster than the MMPT, the compatibility levels that best match typical enterprise UAT testing have been significantly improved.

It is still more work to repackage for MSIX than for App-V, and with less compatibility at the output. So expect many enterprises to require App-V for years to some. For those who use App-V today and are not looking at MSIX, I would recommend starting to move some applications over today so that your staff is better trained for the future.

My focus in 2023 will be in efforts to increase the application compatibility further and make it easier to achieve those results. More is to be learned about leveraging *InstalledLocationVirtualization* by further fine-tuning of the PSF. Additionally, certain classes of apps (especially electron based apps) are problematic and require much more research to solve. And finally there are more under-documented enhancements that Microsoft has made that we have yet to leverage, so more research and testing is needed there.

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 16 years, and by Citrix for a dozen as a CTP/Fellow, Tim is generally known as “The Big Kahuna”, having led the effort to build the original version of App-V at Softricity and builds many free and paid-for tools for packaging under both App-V and MSIX.

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 application 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)
- **For the Developer:** A Developer's Guide To MSIX (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.