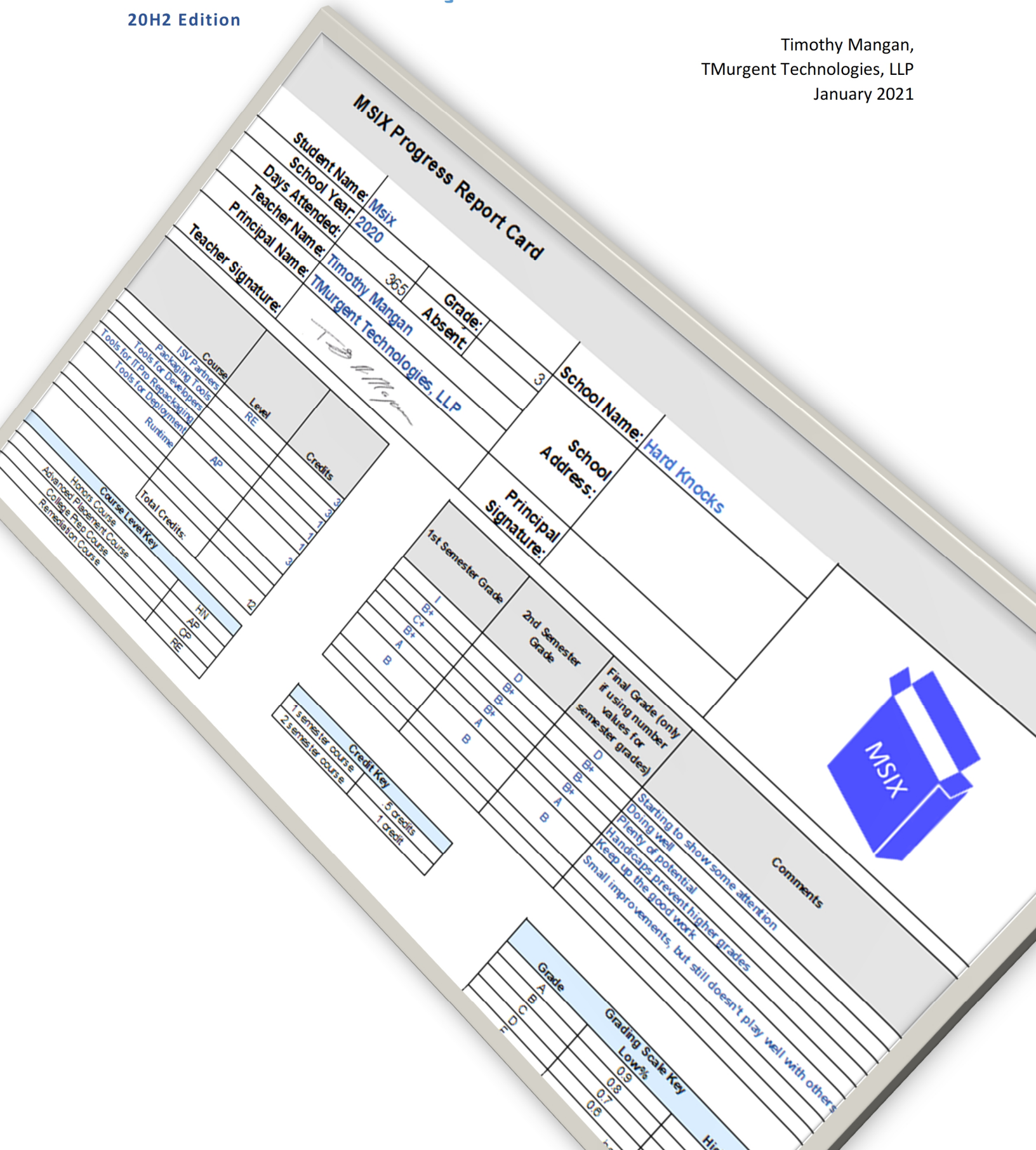
The MSIX Report Card

20H2 Edition

Timothy Mangan,
TMurgent Technologies, LLP
January 2021



Introduction

Nearly three 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 embraces a modified form of the Microsoft Container used by Universal Windows Programs (UWP).

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. While some interpreted that to mean 6-month cycles I always assumed it meant 1-year cycles and openly suggested it might take more than that too. As 2020 ends and we start 2021, it is a great time to take another snapshot of how MSIX looks today. This *Report Card* is an update to our prior views (see http://m6conf.com/index.php/reportcard/46-report-card-2019 and http://m6conf.com/index.php/reportcard/46-report-card-2020)

We can summarize the changes as follows:

- Software vendors have been both interested and wary. Late this year we detected a slight uptick in interest by developers from various forums, but they generally are scratching the surface to understand what they will need to do to be able to release their products in MSIX form. Still, there are few released apps that we can point to. Microsoft's delivery of .Net Core UI3 has the potential to be the bridge they need; we will need to continue watch this space.
- Tooling for IT Pros improved a bit this year, but more is needed.
- With the Windows 10 2004 edition released last spring, Microsoft released some long-awaited updates to the MSIX Runtime built directly into the OS. The addition of Windows Services, better support for HKCU and AppData, plus the addition of Fonts were key.
- The Package Support Framework has also continued to expand, with several new fixups added and improvements to the existing ones.
- MSIX App Attach to support a better dynamic app delivery story.

For this year's *Report Card,* we skipped the Community Survey conducted last year as it seemed unlikely to produce significantly different results than what we saw last year. The survey might return next year. I also expanded the number of applications in our test matrix to 70 this year. Testing of paid-for third party repackaging products was not conducted this year.

This year the report is divided into four categories:

1. Support by Software Vendors to release in MSIX format.
2. Support by Tooling Vendors
3. MSIX Runtime Support in the OS
4. Supplement on MSIX App Attach
5. Compatibility Package Testing

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

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

This mark is not intended to be a criticism of the vendors. I am thrilled to be able to grade this category this year with something other than another incomplete. To be clear, after nearly three years we had hoped to see more progress on this front. However, I don't believe the ISVs are not interested and I don't believe that they dropped the ball on this one.

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.

They often only make big decisions on development plans only once a year. An established vendor wants to innovate and be leading edge but is not going to make the switch to something risky, at least not without a backup plan, and not unless they see value to them.

The reason

Until this year, the MSIX story was more about benefits to customers than to the software developers. Sure, it would allow them to bring their old code into the new format and customers would have better systems, but would the app run any better?

Microsoft has done a better job this year of positioning MSIX as the future for software development. Like any change, it takes years for the ISV industry to change. It took nearly 5 years for most vendors to move from Win32 to MSIX (and many still haven't almost 20 years later), so I guess we shouldn't be surprised that this revolution will take some time too.

Microsoft is committed to making most new programming innovations applicable to applications that are built as MSIX. In essence, they are bringing UWP into the MSIX realm. If these vendors want to be able to innovate in the future, they will have to make their way to MSIX.

The market

Unlike the IT Pro case, vendors do have opportunities to modify their source code to adjust to the MSIX container and build great apps that work in that environment. This is only practical when the vendor can expect that the vast majority of their market will use this new version.

The audience of customers able to accept MSIX apps is growing over time. OS versions that have limited backwards support for MSIX have been mostly phased out. Microsoft did attempt to provide a back-rev OS story (allowing MSIX packages to be delivered and unpacked on systems like Windows 7 and run

outside the container similar to what the MSI used to do), but it wasn't practical. As time moved on, this need has shrunk considerably and is probably not a gating factor for the ISV. Those customers still on the outdated platforms will be well served using the un-updated version of the app also. However, we do need to keep in mind that support of the OS for MSIX is not actually a binary thing. In each new OS Runtime, additional capabilities appear and many of these vendor applications would need the latest OS. So, some vendors must consider what their effective market is if, for example, they require the 2004 OS be used by the customer so that Windows Services work.

The market for MSIX based apps continues to improve over time, but when is it enough for the vendors? Is this this year, is it the next?

The tools

The tools were largely in place last year, and they continue to expand and get better. I do not see this as a blocking factor for the software vendors. Microsoft appears currently to be investing in support for automated pipeline builds of software (CI/CD) preferred by DevOps, but largely without consideration of the need for the Package Support Framework (PSF). While the need for the PSF can theoretically be eliminated from all apps through source code changes, the reality is that unless the app is being written from scratch it has a high probability of needing the Psf.

Meanwhile, in Visual Studio we now have a built-in Windows Application Package project type to build MSIX packages inside Visual Studio. Much like the ability to natively build MSI setup projects, this should be considered only for very simple situations and Microsoft seems to expect ISVs to utilize one of Microsoft's many partners to help with setup projects. I find the WAP lacking reasonable workflows to add and configure the Psf. The third-party vendors that supply setup program capabilities generally do a much better job at this and ISVs should probably be looking at them.

The current state

It remains quite difficult to know just how much MSIX is out there. If you can find a previously Win32/.Net program up in the Microsoft store, it is probably MSIX. I've noticed a few appear this year, notably Vlc Player and Blender, but given a choice of installing their latest release in either MSI or MSIX form – I'd currently prefer the MSI to get full functionality of these products.

We believe that there may be a number of games up in the store that are MSIX, but Microsoft intentionally hides whether apps in the store are UWP or MSIX based so we don't have a good feeling for how much there is really out there. (To determine, you would be required to obtain a copy of each, download and inspect the manifest). At this point, from the customer standpoint, I am no longer interested in even categorizing UWP versus MSIX as long as we can get an app working in the container. The Microsoft store continues to be dominated by games and have few enterprise apps, so this might not be the best place to look for the current state.

So, what about one of the biggest software vendors out there. What has Microsoft released? We have the MSIX Packaging Tool, and each OS release seems to have more app either built into the OS image or delivered as "regional apps" shortly thereafter. I counted 7 MSIX apps last year (I didn't count the UWP)

and I see about 145 this year (with UWP) so they are starting to deliver some things themselves. But I still use MsPaint over the modern equivalent, SnipIt over the intended modern replacement because I find those apps lacking the complete features I want. Not because they are modern, only that someone missed the boat on why I use that app. But what about the more major business apps?

Certainly no MSIX for the flagship Office product, which would become a huge signal to the partner software vendors for the future of this technology. Will the next version of Office be MSIX based? We don't know. That is a huge undertaking! At least Microsoft already has experience in getting Office running under App-V but running under MSIX brings in lots of additional challenges because so many external products need to integrate. Perhaps they'll have to settle on parts of Office in MSIX, and part not. Either way, Office in MSIX is a huge shoe that, if dropped, will signal a shakeup to this industry.

Another place to look is activity in forums where developers ask questions. In past years I had seen little sign activity there outside of those just starting to kick the tires. There is a definite uptick in queries recently, especially around how to deal with activities normally performed at installation time. Only time will tell if this indicates that they are far enough along to be finally concerned with installation or if these queries are part of an overall evaluation before they take on a project that turns into an MSIX product.

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. This year Microsoft has done what it needed to make MSIX seem like the inevitable future for software vendors, so they are paying attention. But these vendors know that just because Microsoft is temporarily committed to a direction, doesn't mean they won't change direction in the future. The vendor must believe it in their best interest.

2 Support by Tooling Vendors

B+ Tooling Vendors stayed steady at "B+" this year. The vendors, especially the most popular vendors, have been very active in the MSIX community.

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 "B-". Many of the vendors have products available and you can build and deliver applications using these, as long as the applications do not run afoul of the MSIX Runtime restrictions. In each of the last two-year vendors earned an "A-", and I suggested that these vendors will need to up their games to earn such a high mark. While these tools continue to improve, I have not seen significant improvements, and they no longer deserve that mark.

Most traditional installer tooling vendors support building MSIX packages. Several new companies have also sprouted up in the last couple of years as well. These challengers are offering generally simpler products at a lower price, but completion in any form tends to make everyone work just a little harder.

In addition to that tooling, there is the open source MSIX Packaging SDK on GitHub. This really is some lower-level code that others can build upon, and I suspect that even though some of the early vendors built their own code initially, they may be migrating to using at least some of this. Supporting the open-source community with pre-built tooling is needed and it would be great to see these vendors also contribute back as well.

Going forward, all vendors will need to consider how to help developers create applications using a mixture of traditional and modern components. It isn't necessarily enough to say you can build the package if everything conforms. Some are starting to provide better guidance and analysis on what is

needed; however this analysis is still pretty rudimentary. The tools usually provide great information on the limited items built into the tooling, but the unknowns are vastly greater than the knowns at this point.

2.2 Tooling Vendor Support for Repackaging

B+ Software Vendors supporting IT Pro Repackaging of applications into MSIX earned a "B+" this year. After a phenomenal early start, we dropped down the grade last year due to limited progress. This year we again see good progress in application compatibility.

Microsoft delivered multiple releases of the Microsoft MSIX Packaging Tool (MMPT), reacting to feedback from the early version we were testing a year ago. It is a bit more mature and easier to use than a year ago. Significantly, my testing indicates that issues with creating some form of package have largely been eliminated, but there are still many API features used by traditional apps that should be able to work under the MSIX runtime that it does not support. Meanwhile a few issues crept into this tool affecting this report. There were a few packages that failed to package in this snapshot in time; packages which did complete last summer (even if the app itself wasn't completely successful).

Meanwhile, the Package Support Framework (PSF) has had significant improvements. This is an open-source project hosted on GitHub that includes Detours (some of the technology behind App-V and other vendor products) for Windows API interceptions.

As a contributor to the PSF, I added significant improvements this year to PsfLauncher scripting, to the FileRedirectionFixup, and added three new fixup types this year to address different problems that we have seen in traditional packages:

- **DynamicLibraryFixup** – addresses dll not found issues that occur due to changes in working directory, lack of App Paths support, and inability to affect the Path variable.
- **RegLegacyFixups** – addresses two types of registry requests that work natively but cannot under MSIX.
- **EnvVarFixup** – Brings environment variable support to MSIX.

Additionally, 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 started to make PSF suggestions based on analysis of the static components, but it can only report on a subset of the issues commonly faced when repackaging.

Other packaging vendors also include the PSF in their own repackaging products, combining the equivalent of PsfTooling and the MMPT into a single experience. Although testing was not performed this year on these products to compare their level of application compatibility to that of the MMPT/PsfTooling combination, I believe these tools generally fall at or below the results shown in this report due to the timing of this snapshot. However, if application compatibility is of upmost importance, I expect that the third-party vendors will be better at certain apps, so using a combination of tools would result in the best set of outcomes.

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. This new support leverages the embedded update processes inside of MSIX, allowing the vendor update rom 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.

This pretty much means any vendor using the AppX PowerShell commands can probably handle MSIX without any changes, with the possible exception of understanding the new file extension.

A long-awaited entry into this space is also from Microsoft; *MSIX App Attach.* This report will cover App Attach as a separate section.

I have been cautioning customers that MSIX is coming and they need to prepare their organizations for the operational aspects of MSIX. Taking a few apps to repackage and deploy would help the organization prepare, not only for deployment but other aspects such as how the help desk handles these apps. Even with the necessary tools in place, we are already hearing of queries from customers asking how to convert a vendor MSIX package back into another form; obviously these customers did not heed the caution and are now panicking.

3 MSIX Runtime Support

B The MSIX Runtime moved up to a "B" grade this year, up from a "C" last year. Ultimately what is important is what can I deploy into production now, so progress has been made but we still have a long way to go in App Compat.

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

- *Support for Services.* The 2004 OS release included support for Windows Services being delivered in the MSIX Package. Unlike App-V Services, these services are not virtualized but are natively installed at the time of package installation. The service installation does require an elevated privilege level. So even through most packages may be installed by a standard user without a UAC prompt, those including services (including disabled services) do. This isn't an issue for automated deployment tools as they already use privileged processes to install MSIs.
- *Fonts.* The 2004 OS also included support for deploying fonts. The MMPT (as of the end of year 1209 release) still does not support them, but they may be manually added, or a third-party tool used.
- *HKCU Access Changes.* Many applications can now make modifications to Current User hive registry settings in the application. Previously these were immutable if delivered as part of the package, but most apps may now make changes that are handled and isolated by the runtime. There are apps that make certain kinds of HKCU requests that are not supported, and the PSF now has support for those cases. Of course, the HKLM hive remains immutable, but this affects only a small set of apps.

The list of things not yet supported remains large. An announced future feature to support letting multiple packages run in a single container is needed. Microsoft has been telling us that they are working on this for a year and a half, and we still do not even have details on how this would work. Shell Extensions are critical to certain applications. And while there is support for COM and Services, there is still a lot of work to be done to achieve broader compatibility.

To date, Microsoft has not been as forthcoming with roadmaps for MSIX support and features as we are used to seeing out of their product teams. Even when features in the product are added or problems are addressed, we sometimes must "discover" the changes ourselves. In conversations with customers, I constantly hear questions from them regarding how committed Microsoft really is to MSIX. Increased transparency would go a long way to make customers more comfortable with the direction.

4 Repackaging Testing

At the end of the day, we need to be able to deploy the apps. Without a log 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 70 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.

These applications were repackaged and tested six times:

- Using Microsoft App-V.
- Using the Microsoft MSIX Packaging Tool (2020.1209 release) without shimming.
- Using the Microsoft MSIX Packaging Tool (2020.1209 release) with PsfTooling 3.8.0.

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

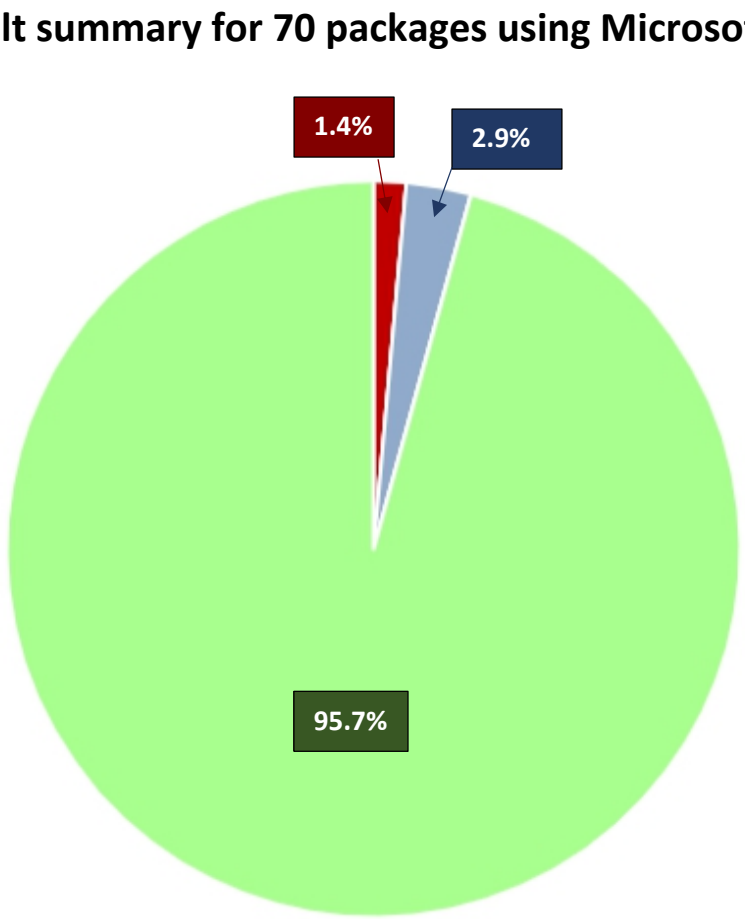
Category	Description
Untested	Incomplete. Most likely this indicates incomplete testing. There are cases where something might be blocking the test, or perhaps you are viewing the results at a time while testing is in progress.
No Workflow	Tooling vendor has no exsiting reasonable workflow for producing 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.

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!

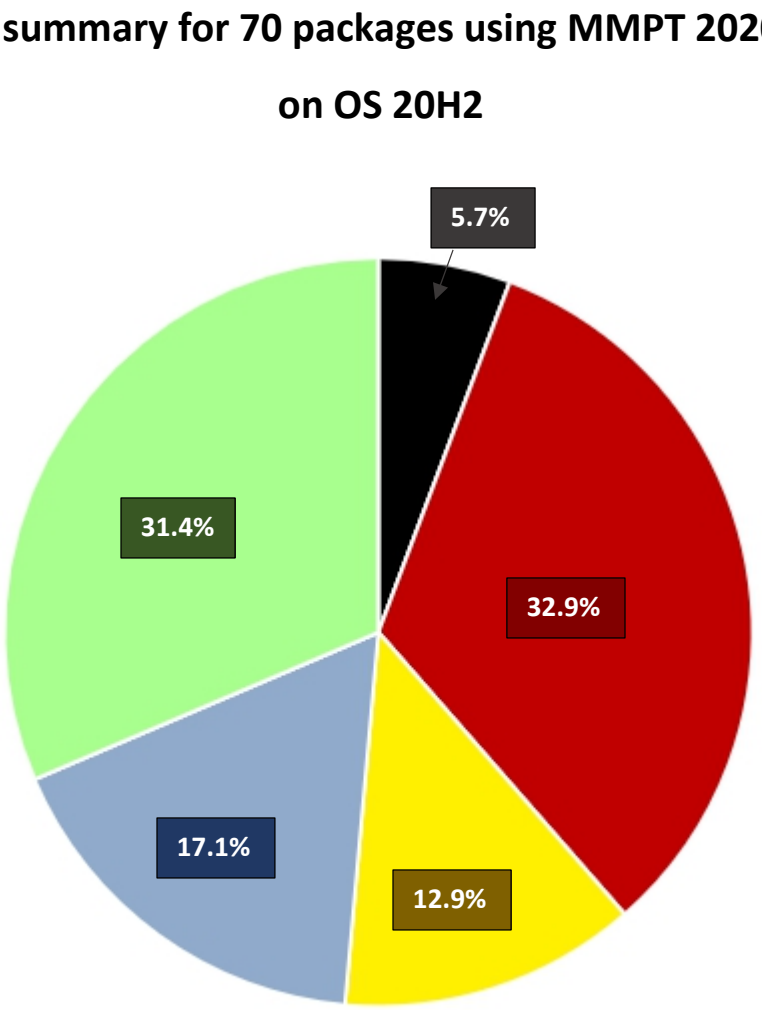
The 70 packages were packaged on Windows 10 20H2 and tested on the same OS. Although many customers continue to use the 1803 Sequencer and then TMEdit to overcome the short-name bug and other issues, Microsoft finally addressed the short-name bug with the 2004 ADK. While skipping TMEdit completely has a small impact on package quality, for this testing I used only the ADK for 2004 Sequencer on 20H2 OS without additional tooling.



Not surprisingly, these results were very good using seasoned tooling and collective wisdom on techniques to package with App-V. If anything, I was harsh in the objective judgment on those 3 apps.

4.2 Results from Packaging with Microsoft MSIX Packaging Tool

The same 70 applications were packaging using the 2020.1209 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 results are better than last year, due to a combination of improved runtime support and updates to the tool. Customers are cautioned that a 2004 or above operating system on the systems that the apps are deployed to are required to achieve these improved results. Of note:

- I performed some preliminary testing using the 2020 summer version of the MMPT and did not see the package failures seen. These issues seem to have started with the 2020.1006 build; Microsoft is aware of the issues and I would expect them to be addressed in a future version of the tool. The apps affected would not likely have fallen into the “Success” category anyway.

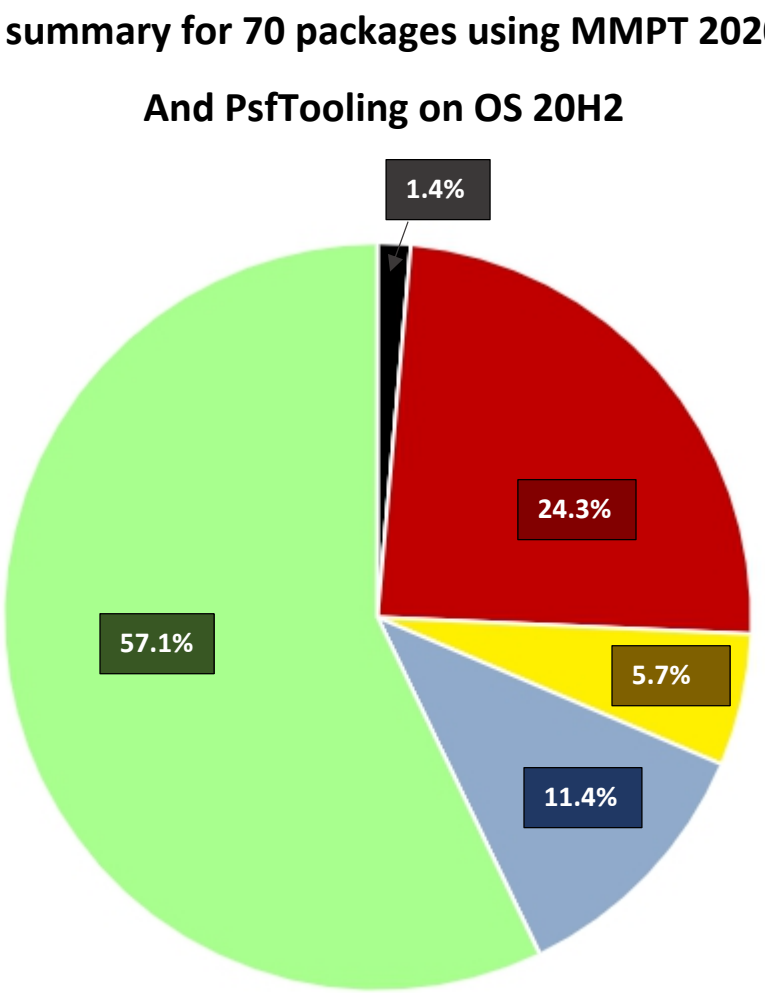
- Based on preliminary testing done on 2004 OS, I do not believe these results would be significantly different if packaged with the 1219 build of the MMPT and tested on 2004.
- Those on 1909 or earlier Oss should look to last year's report, even if using the latest packaging tool.

These improvements are encouraging, but not likely to convince an enterprise that MSIX is ready yet for their apps, at least not without additional help.

The MMPT includes a remote option now, however it assumes that you use an external tool to control the state of the capture virtual machine, so I did not use it. It seems appropriate only for a packaging house that builds their own tools.

4.3 Results from Packaging with Microsoft MSIX Packaging Tool with PSF

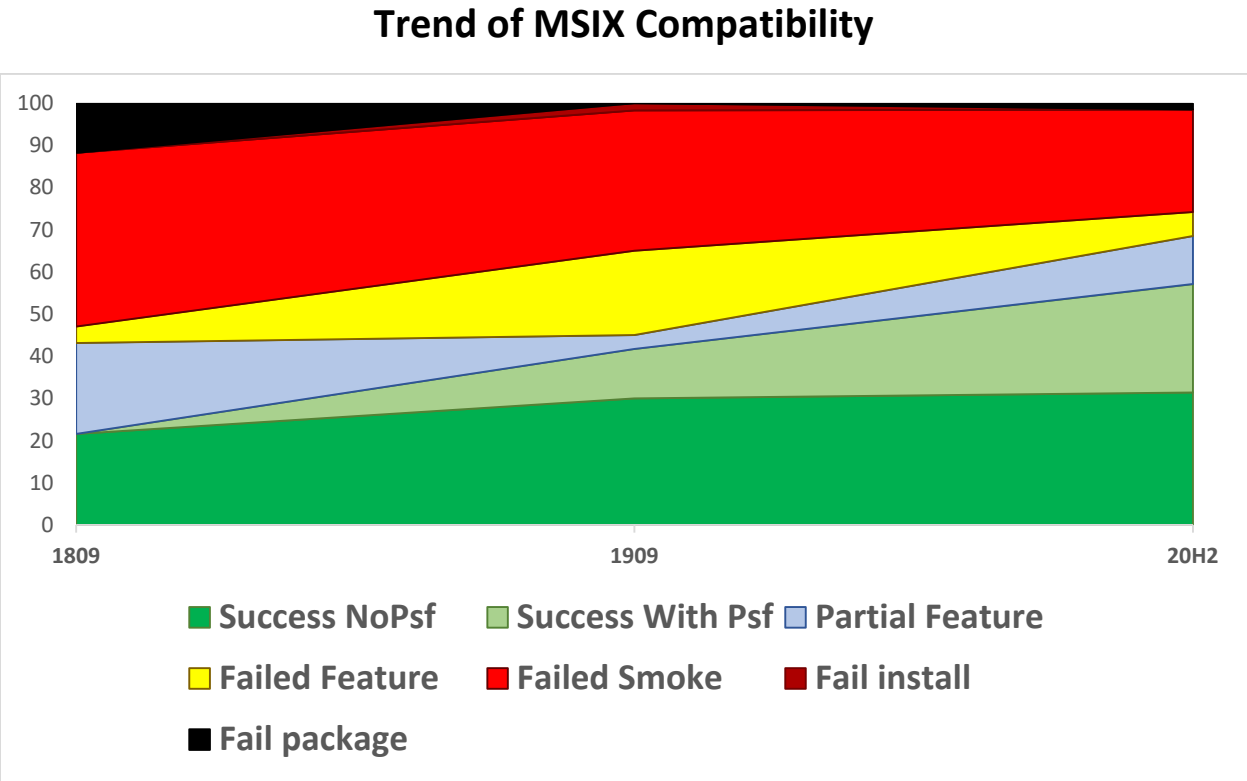
For these tests, the MMPT version 2019.1209 was also used, but the packaging process was enhanced by using PsfTooling 3.8.0 to inject and configure the Package Support Framework (PSF) into the package. Additionally, there were cases where manual manifest editing was needed to achieve compatibility success. This version of the PSF used contains a build of the latest PSF source code from GitHub available in January of 2021. The same set of 70 Apps were tested.



Much of my own efforts on the Package Support Framework were made specifically to target issues highlighted in last year's report. While additional work on the PSF is needed, until additional improvements to the runtime are made there are limits to what may be accomplished in the short run.

4.4 The trend

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



Overall, the trend has been improving. Of most concern to me currently is that achieving these results requires a significant amount of packaging expertise and time; by far more than is required for other types of repackaging.

My focus in 2021 will be in efforts to make achieving these results easier for most customers through improved tooling, and other packaging tooling vendors appear to be headed in this direction as well. For now, depending on those inside Microsoft to improve the platform capabilities, and those outside to make the possible more palatable seems like the right direction to take.

What's missing in MSIX

Below is my short summary of support missing in 1909 from last year, with updates to this year. There are more items that should be added to these lists, but these are what I believe to be the key ones.

Item	1909	2004/20H2
Support for Services	No	Yes (multi-user issues)
Apps modifying registry in package	No	HKCU only (with help from PSF)
Fonts in the package	No	Yes, with help
Environment Variables in pkg	No	Yes, via PSF
Plug-ins for native apps	No	No
Shell Extensions	Some	No improvement
Software Client System	No	Partial
Exposing COM	No	No
Scheduled Tasks	No	No
Run Keys	No	No
Layer Hiding (hiding native install)	No	No
Deletion Markers Ref & File	No	No
WMI Providers	No	No
ETW Providers	No	No
Object Spoofing	No	Maybe a little

For some of these items, Microsoft might choose to not support older applications that need these items, preferring to wait it out for vendors to rewrite their software to not require them. Given the long shelf life of software in the Enterprise, such a strategy will be a pain point for customers that want to move everything over to MSIX.

5 MSIX App Attach

The long anticipated MSIX App Attach was finally released to production for Windows Virtual Desktops at the end of the year and it deserves some discussion, even if it is too early to "grade".

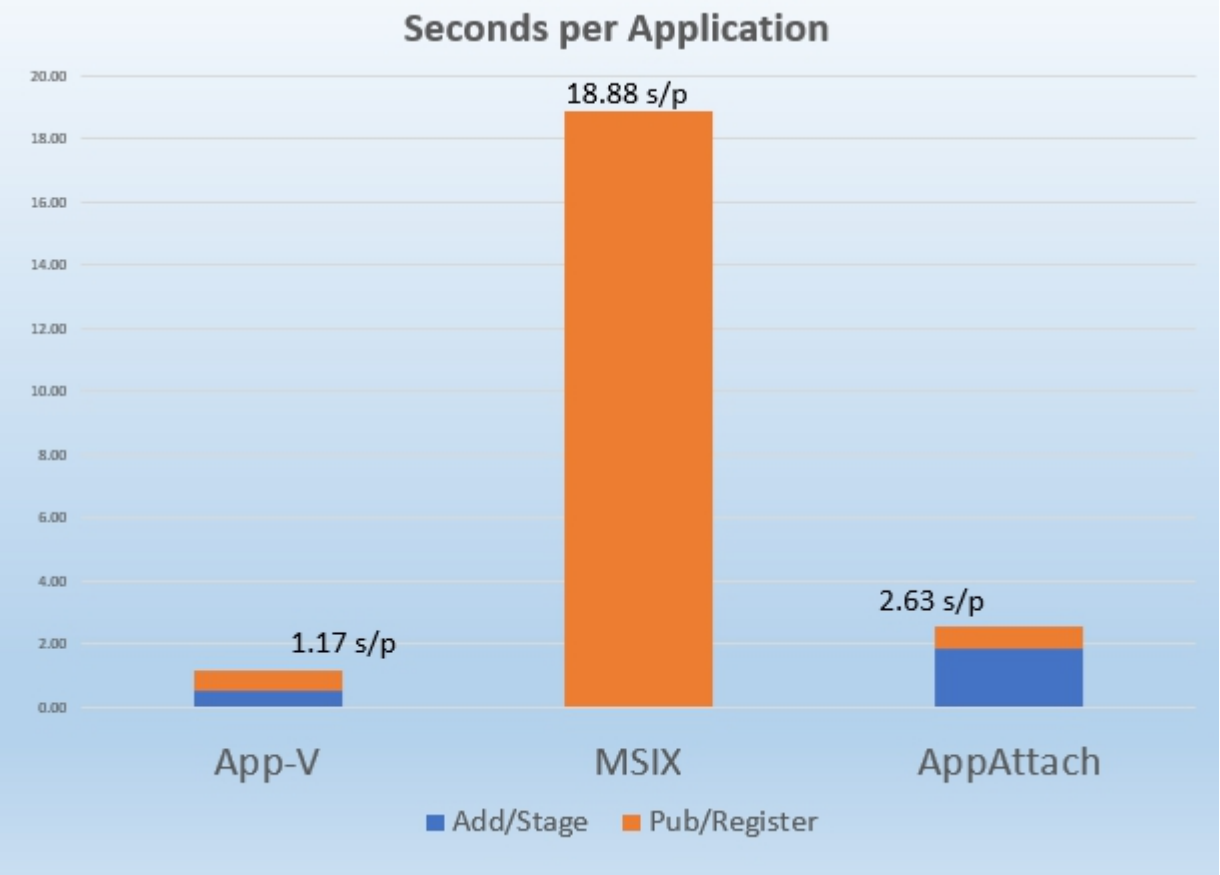
Microsoft's focus for MSIX App Attach is to support MSIX in a non-persistent environment in general, but specifically for Windows Virtual Desktops (WVD). Built by the folks they acquired from FsLogix, it offers a fast installation of MSIX apps for scenarios where the user logs into a "fresh" copy of Windows each day and dynamically delivers assigned apps on the fly. This type of support is critical if MSIX is to replace App-V in the non-persistent and semi-persistent environments.

Microsoft's efforts should be viewed as being two parts. There is an underlying technical part, plus specific work done for integrating into the Azure WVD experience. The underlying part is also useful in scenarios outside of WVD, and most of us that have tested pre-release versions of App Attach did so by only using the underlying parts. We expect additional vendors will embrace this part of App Attach as part of their products. Indeed, several have already announced current or future support for App Attach outside of WVD.

While the technology looks good, there are some performance concerns to be addressed, plus there is the question of how to deploy apps not currently compatible with MSIX into this WVD environment. For some this means a combination of baking MSI installers into the golden image or utilizing App-V and MSIX; but if you already have your apps working in App-V outside WVD there is little incentive to move any of them over until all can be migrated to MSIX.

I did release some preliminary performance results last summer on publishing time for App Attach. This testing showed that for the non-persistent environment, MSIX App Attach greatly outperforms the traditional MSIX installation time at logon, however publishing times were still significantly longer than the App-V benchmark. Better app-compat from MSIX in general, and improvements in this publishing performance will be needed this year.

This data comes from some simple testing of 25 apps against pre-release App Attach software. As seen in the following chart, I found that the time for MSIX App Attach to publish these apps was significantly faster than for regular MSIX installation. It was, however, not as quick as for Microsoft App-V on those same apps and this is probably the benchmark that App Attach should be trying to beat.



Because this testing was performed using pre-release software, there is the likelihood that Microsoft has already improved up this. We know that improvements, such as moving from ISO to CimFS, will impact publishing performance time in the future. I expect CimFS to improve scalability (the ability to handle publishing under load from multiple users) rather than having a great impact on the single user testing performed above, but we need both kinds of performance improvement.

6 Community Surveys

The community survey conducted last year was not repeated this year as I did not believe the responses would be significantly different. I anticipate running the survey again 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 than a dozen years, and by Citrix for a half 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.

As an independent contractor, we 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.