

Almost 20 years of Cyclone; past, present and future

Alexandre Torres Porres, Matt Barber, Derek Kwan

Brazil, USA

Resumo: Este artigo discute a história da biblioteca Cyclone, cujo propósito é de clonar objetos do MAX/MSP ao Pure Data e prover compatibilidade entre esses dois ambientes. Este artigo descreve as diferentes fases e versões da Cyclone dos seus primeiros momentos à fase atual e planos futuros.

Palavras-chave: Pure Data, MAX/MSP, Cyclone.

Abstract: This paper discusses the history of the Cyclone library, whose purpose is to clone MAX/MSP objects into Pure Data and provide compatibility between these two environments. This paper describes the different phases and versions of Cyclone from its early days to the current state of development and our future plans.

Keywords: Pure Data, MAX/MSP, Cyclone.

MAX was developed by Miller Puckette in the 80's at IRCAM¹. At first, it could not process audio signals, just control data. After a period where MAX was sold by Opcode in the 90s, David Zicarelli (who was part of the original MAX team at IRCAM) started distributing MAX in the late 90's via Cycling '74, his own company. Before that, Cycling '74 already existed and was distributing their MSP external package for MAX, which was able to generate and manipulate audio signals. By then, Miller Puckette had already started developing Pure Data (or just 'Pd') as a software quite similar to MAX, but open source. Pd had audio signal processing capabilities in its origin, and MSP was based on Pd's original open source code.

At first, MSP was developed independently. When Cycling '74 started selling MAX (still at version 3.0), one could purchase MAX and MSP separately or as a single combined purchase. But when MAX 4.0 was released, MSP was fully incorporated and was released together with MAX as a unified distribution.

The Cyclone project was born at this time in history, when MAX 4 was out and was still a pretty close relative to Pure Data. Its goals were to clone control (MAX) and signal (MSP) objects for Pd as the first step of a larger project aiming to unify computer music tools. Unfortunately, Cyclone was abandoned in 2005 and this goal was not fulfilled. It did not completely clone all objects from MAX 4² and even if it did, MAX has grown and evolved a lot since then. Since MAX 5, many new packages have been incorporated into the MAX environment (now at version 8) - for instance: *jitter*, *gen~*, *vizzie* and *beap*.

Since the compatibility between MAX and Pd has drifted further apart at each new version of MAX, Cyclone's goal to provide compatibility between these two environments made more sense during the MAX 4 era. But Cyclone is a quite successful external package for Pd and has been a great contribution to the community by providing a large set of Pd externals as well as some level of compatibility between MAX and Pd. Even though Cyclone doesn't include all of the objects in MAX/MSP (plus objects from other packages of MAX), it still helps users of both systems to

¹ *Institut de Recherche et Coordination Acoustique/Musique*, see <<https://www.ircam.fr/>>

² It was common to call it MAX/MSP at version 4, but now the 'MSP' part is more of a historical reference as it has long been incorporated into the core of MAX. From now on, we'll use MAX to refer to MAX and MSP (plus other packages it currently includes) – so it doesn't only refer to the original control objects.

develop equivalent or similar patches using a good portion of the tools included in the older MAX/MSP packages. Cyclone also gives MAX users a convenient gateway to Pd, which is especially necessary when adopting tools like Raspberry Pis and designing apps with libpd.

1. Cyclone's first development phase by Krzysztof Czaja (2002 to 2005) and its original structure

Krzysztof Czaja, the original author of Cyclone, announced in the Pd list the development of this project in April of 2002³. This was just after the release of MAX 4.0 and this first phase of development went on until 2005, when it was abandoned still at an alpha stage (version cyclone-0.1-alpha55). By this time, MAX was still at later iterations of version 4, but Cyclone was still only compatible with the original version 4.0.

Cyclone was part of a larger library called miXed, whose original goal was quite ambitious. Cyclone's original readme mentions how the end goal was meant to “*eventually become part of a much larger project, aiming at unification and standardization of computer musician's tools*”. It is unclear exactly what this was supposed to mean, but we can at least deduce that the idea was to better unify Pure Data and MAX, which were still closely related at that point. For reference, check the original readme provided in the APPENDIX, which also describes the original structure of the library, also reported here in the next paragraphs.

In its original format, the Cyclone library could be loaded in different ways, and consisted of different sub-libraries⁴. The main sub-libraries were 'hammer', which included the control objects (part of 'MAX'), and 'sickle', for the signal ('MSP') objects. These sub-libraries could be loaded individually.

Cyclone included two other binaries, namely the *cyclone* and *maxmode* libraries, both of which would also load *hammer* and *sickle* internally. More precisely, loading the *cyclone* binary would load *hammer*, *sickle* and the 'cyclone sub-library'. The *cyclone* sub-library would then load a

³ K. Czaja, “[PD] pd-max compatibility” <<https://lists.puredata.info/pipermail/pd-list/2002-04/005949.html>>

⁴ A library in Pd is a set of objects, but it can also mean a single binary that loads two or more objects. By 'sub libraries', we mean Cyclone had separate binaries that would load different subset of components/objects.

small set of operators (!-, !/, ==~, !=~, <~, <=~ , >~, >=~ , !~ , !/~ , %~, +=~). When instantiated as an object⁵, there was a feature for an importing mechanism to load the now very old MAX binary patches with the *.mxh* extension. This was a clear effort to better unify MAX and Pd and worked by clicking on the object, which would open a dialog window for you to select the file to import.

To this day, Pd can still open and save MAX patches with the *.pat* extension as well as MAX's text files with the *.mxt* extension. Both of these extensions are even older than the *.mxh* extension and Pd still being able to load them points us to the early period when Pd and MAX were very closely related. However, Pd was never able to manage the relatively newer *.mxh* files, so this is why the importing mechanism provided by Cyclone was required.

In addition to all these components and functionalities loaded with the *cyclone* binary, there was still another optional *dummies* sub-library binary that instantiated non-functional objects with the names of classes not yet cloned in Cyclone. In principle, this allowed the user to open a Max patch without tripping on missing objects. The idea then would be to replace these dummy objects with functional alternatives.

If Cyclone was loaded via the 'maxmode' binary, it would load all of the aforementioned components (*hammer*, *sickle*, the *cyclone* sub-library and the *dummies* sub-library); in addition, *maxmode* also attempted to enforce strict compatibility to MAX, as some objects would have been incompatible otherwise. An example is the *prepend* object, which would create an extra right inlet when created without arguments, while the MAX original only has one inlet in any case. However, study of the source code for the alpha versions of Cyclone reveals that *maxmode* could not yet accomplish this. The *maxmode* object⁶ had the same import mechanism as the *cyclone* object to load MAX patches with the *.mxh* extension.

Cyclone also provided an executable binary called *cyclist*. This was an alternative to the import mechanism found in both the *cyclone* and *maxmode* objects. The *cyclist* binary converts from the *.mxh* format (binary MAX patch files) into equivalent *.pat* text format files. Cyclone did not provide an export function to save patches as *.mxh*, but the user could employ Pd's built-in functionality to save a patch as a *.pat* file that would open in Max.

⁵ A library binary can also be loaded and created as an object if programmed for it; this is the case with 'cyclone'.

⁶ Like 'cyclone', 'maxmode' can also be created as an object.

MAX 5, released in 2008, introduced a new JSON-based file format with the *.maxpat* extension. MAX 5 and later versions can still read the older *.pat*, *.mxt* and *.mxb* files, but they cannot save them as such anymore. Pd has never had the ability to load or save *.maxpat* patch files, so Pd and MAX have had incompatible files formats for over a decade.

2. The Pd-extended era (2005 to 2013) by Hans-Christoph Steiner

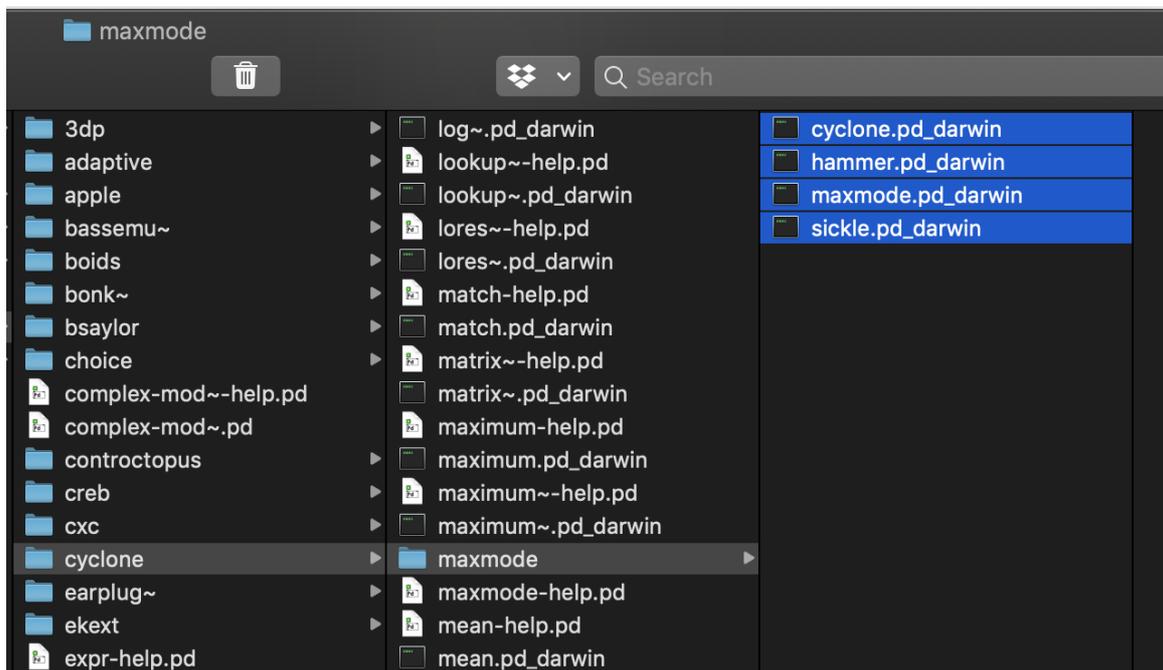
The Pd-extended project was a fork of Pd that provided many pre-installed libraries and Cyclone was part of the package from the beginning. Therefore, at the time, one could either have Cyclone installed into proper Pd (aka “Pd Vanilla”) or obtained as part of Pd-extended. 2005 was the year that the Pd-extended project gained a lot of traction, but also when Cyclone ceased its development (at version 0.1-alpha55 at this point and remaining at it for a long time).

Hans-Christoph Steiner was the main force behind Pd-extended, so he took over the maintenance of Cyclone and distributed it as part of it. Cyclone 0.1-alpha55 was included in Pd-extended from versions 0.39-4 to 0.42-5 and got a minor update to version 0.1-alpha56 at Pd-extended 0.43-0. This was the last version of Pd-extended, released in January 2013. Pd-extended was then abandoned and Cyclone was left unmaintained as a result. Unfortunately, this 0.1alpha-56 version meant to fix a bug reported by Alexandre Porres in both *cartopol~*/*poltocar~*, where the sign of the phases were inverted, but this error was only corrected in *cartopol~*. Hence, the conversion in *poltocar~* still considered an inverted phase information, which lead to errors in signal reconstruction from the now correct *poltocar~*, ruining many spectral processing patches.

Like all its included libraries, Pd-extended provided each of Cyclone’s objects as a separate binary instead of preserving its original *hammer*, *sickle*, *cyclone* and *maxmode* sub-libraries organization in which many objects were included in a single binary. Because of this, Pd-extended users were not aware of these sub-libraries and the ways they were intended to be loaded. This changed the structure of Cyclone significantly and resulted in some hidden and missing features. One can then argue that the versions provided in Pd-extended are actually parallel versions of the original Cyclone project. Nonetheless, Pd-extended was the vast majority of users’ introduction to Cyclone, so any hidden or missing feature from the original project was simply lost to oblivion.

Although Pd-extended loaded Cyclone's objects from separate binary files, it still provided the library binaries *hammer*, *sickle*, *cyclone*, and *maxmode* (the *dummies* binary along with its intended functionality were excluded, for some reason). These binaries were hidden inside a 'maxmode' folder for seemingly no reason (refer to Figure 1).

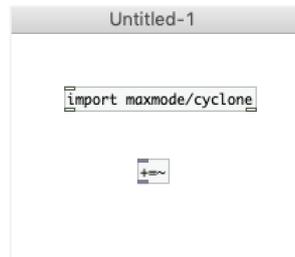
FIGURE 1 – The file structure of Pd-extended, showing objects as separate binaries and a 'maxmode' folder including the library binaries for cyclone, hammer, maxmode and sickle.



Source: the authors.

However, the *hammer* and *sickle* binaries were non-functional, as attempts to load them caused Pd-extended to report errors. Thus the only way to load most of Cyclone's objects was from the individual object binaries. Loading the *cyclone* binary would load the 'cyclone' sub-library (set of operators: +=~, >~, etc.) but wouldn't load hammer and sickle and produced the warning '*creating cyclone object without loading cyclone components.*' Unfortunately, users were simply unaware that they could load the operators by loading this hidden library (see figure 2).

FIGURE 2 – Loading the cyclone sub library in Pd-extended via the [import] object, and calling cyclone inside the maxmode folder. This allows us to create operator objects like [+~=].



Source: the authors.

You could also load the *cyclone* object in Pd-extended with its importing mechanism, enabling the user to choose a *.mxh* file to load when clicking on the object. The *maxmode* library was not fully functional, as it could not load the missing *dummies sub-library*, but loading *maxmode* as an object still also allowed the user to open MAX's binary files. Pd-extended also included the *cyclist* binary and actually incorporated this import functionality into its core code, making it possible for Pd-extended to load *.mxh* files directly via the Open menu, which converted them to *.pat* and opened them in one step.

Cyclone 0.1-alpha56 (the last version of Cyclone from Pd-extended) is still available via Pure Data's external manager (a.k.a. 'deken'), but listed as 'cyclone-v0-0extended'. Other libraries that were also part of Pd-extended are equally found with the same 'v0-0extended' version name, which indicates that it is the version of that library as found in the last version of Pd-extended. The cyclone-v0-0extended version installs with the same structure as found in Pd-extended, like hiding the non-functional libraries *hammer* and *sickle* inside a folder named *maxmode*, and omitting the *dummies* binary. The *cyclist* binary is missing in this download because the *cyclist* binary was not found in Pd-extended inside the 'cyclone' folder, but in a dedicated folder for binaries alongside the binary of Pd-extended itself.

3. Fred Jan Kraan era and the nilwind fork

In December 2014, Fred Jan Kraan started developing for Cyclone again. It had been almost 10 years since Cyclone received much development attention. The first release of this era was 0.1-

alpha57 and other new releases were provided by him as Cyclone 0.2-beta versions. This maintenance era included many bug fixes, most of them reported by Alexandre Porres. The cartopol~/poltocar~ issue was finally fixed, for instance. In this phase of development, Cyclone objects were provided as separate binaries as in Pd-extended, but the original library binaries (*cyclone*, *dummies*, *maxmode*, *sickle* and *hammer*) were completely removed.

Pd cannot properly load single binaries with non-alphanumeric object names (such as +=~) due to constraints of OS file systems. This is why these needed to be loaded as part of the *cyclone* sub-library. In this development phase, a new library called *nettles* was created to provide the operator objects. The *nettles* sub-library did not include the functionality of importing files by clicking on an object, as was possible with the original *cyclone* object. The *cyclist* binary was also removed (but eventually brought back as a Windows-only executable).

In February 2016, Kraan abandoned development of Cyclone and, in response, a fork by Alexandre Porres was announced as the current development branch, maintained by Alexandre Porres, Derek Kwan and Matt Barber. At the time, the last Cyclone version released was 0.2-beta-1 (released in December 2015). Kraan's repository was renamed to miXedSon and kept releasing new maintenance versions. The last releases from Kraan's repository were 0.2-beta-2 (from July 2016) and 0.2-beta-3 (from September 2016). This was before Porres, Kwan, and Barber provided a new stable 0.3 release.

Fred Jan Kraan created a new fork of his miXedSon repository in 2019, carrying the name *nilwind*. It is not in much active development, making it almost the same as the previous Cyclone 0.2 versions. The development of *nilwind* is mostly motivated by the concern that the current and further developments of Cyclone (0.3 onwards) by Porres, Kwan, and Barber could provide instability and regression bugs (an unfounded worry, as we discuss later).

4. Cyclone in Pd-l2ork/Purr Data

In 2009, Pd-extended was forked and served as the basis for Pd-L2Ork 1.0 (for Linux only). Purr Data started as Pd-L2Ork 2.0 as a cross-platform software that utilizes nw.js instead of tcl/tk (utilized by Pd Vanilla, and Pd-extended). Currently, Pd-L2Ork is now also at a parallel 2.0 version

utilizing the nw.js port from Purr Data, but still including original functionalities from Pd-L2Ork 1.0 that didn't make into Purr Data plus other improvements. This means that both Pd-L2Ork and Purr Data are now distinct projects not compatible to each other.

Both Pd-L2Ork and Purr Data inherited the last version of Cyclone from Pd-extended (0.1-alpha56) and still include it. Hence, they inherited the issues from Pd-extended (such as lacking 'dummies'). However, the version of Cyclone they carry does include some bug fixes from the Fred Jan Kraan era, like the *cartopol~/poltocar~* conversion issue. There are also other changes, like *prepend* becoming an abstraction that is not compatible with the original version in MAX and a new argument for the *coll* object to operate in 'threaded mode'. Therefore, the Cyclone version included in Pd-L2Ork/Purr Data is a parallel version from the one found in Pd-extended (which can be considered yet another parallel version on its own of the original Cyclone).

Moreover, this version has issues of its own as Cyclone has not been fully ported to nw.js yet (at the time of this writing). In Pd-L2Ork, this corrupts the objects *mousestate* and *mousefilter*. More drastically, the *comment* object was simply removed.

To make matters yet more complex, at least at the time of this writing, Purr Data has the same and yet more issues with Cyclone objects. It hasn't ported the click functionality that loads text windows in some objects, which are: *coll*, *table*, *seq*, *capture*, *capture~*. Pd-L2ork 2.0 has these fixed and you can also click on the *cyclone* object to import MAX files. Pd-L2ork 2.0 can also open (from the Open menu) *.pat*, *.mxt* and *.mxb* files. The import mechanism from the Open Menu doesn't work yet in Purr Data, neither you can click on the *cyclone* or other objects.

5. Cyclone 0.3 by Porres, Kwan and Barber (2016-2019)

In February 2016, Porres forked a new repository for Cyclone from Kraan's, at version 0.2-beta1. Derek Kwan and Matt Barber joined Porres to form the development team from this point on. In that same year, Porres, Kwan and Barber wrote a paper for PdCon16 [1] proposing an update to the Cyclone library.

This new development effort would attempt to bring the compatibility level from the then quite old MAX 4.0 to the current version at that time (MAX 7). The resulting version, Cyclone 0.3,

was released on February 21, 2019, exactly three years into the project. MAX 8 had already been released in 2018, but Cyclone 0.3 targeted compliance to the last version of MAX 7 (7.3.5). Cyclone 0.3 proved to be a humongous overhaul of previous versions of the code. In total, 60 objects were updated to be compatible with versions of MAX up to 7.3.5. In addition, we wrote a new documentation set from scratch, added 40 new objects, and provided numerous bug fixes – many crucial – that affected the pre-existing 65 objects. Finally, we simplified and improved the build structure and updated the code to comply with better C programming practices and removed many dependencies making the structure much simpler.

The release of Cyclone 0.3 was made available after an extensive period of testing, only after we were satisfied enough with its stability to release without ‘alpha’ or ‘beta’ designations. At the time of this writing, Cyclone’s development is still active, and is now at version 0.6. Newer versions have fixed yet more bugs and issues, further improving the library’s stability. Hence, the concern that the *nilwind* fork aims to address is unfounded. We had minor regression bugs as a result of our work. Just two such issues have been reported in almost 6 years of development (one of which was opened by ourselves) and both have been fixed rather quickly.

We inherited Cyclone with some missing functionalities. The *maxmode* and *dummies* binaries had been removed for good and *cyclist* was no longer part of the build target. Instead of ‘nettles’, we have partially restored the *cyclone* sub library in Cyclone 0.3. Like *nettles*, it only carries the small set of operators. This can also be created as an object, but we did not include the original importing mechanism in it. Also, Cyclone 0.3 provides aliases for these operator objects as single binaries (for instance, *plusequals~* as an alias of *+=~*). These aliases are also present in MAX and provide an alternative to using the *cyclone* sub library. On the other hand, as of Cyclone 0.5, the *cyclone* object now has a new functionality that can output the library version, which can be used to check if you have the correct compatible version needed by the patch.

As an update to our 2016 paper, some of the new objects that were promised to be included in Cyclone 0.3 did not actually make it: *biquad~*, *filtercoeff~*, *freqsbift~*, *cascade~*, *hilbert~*, *saw~*, *rect~* and *tri~*. This reflects a new mindset that pursuing cloning all or too many objects from MAX is not desirable anymore, as we’ll discuss next in this paper. The 2016 paper also mentioned good candidates to be cloned in the future, namely *radiogroup*, *matrixctrl*, *rslider*, *kslider*, *multislider*,

spectroscope~, *dict*, *transport*, *groove~* and *gizmo~*. Because of this new mindset, we can forget about incorporating this or any other new object in Cyclone. Which means we'll stick to fixing bugs and eventually include new functionalities from new versions of MAX.

6. Current/last phase of development (Cyclone 0.4 to 0.6)

Newer versions of Cyclone since 2019 do not include the same amount of changes and improvement that were achieved in Cyclone 0.3. In this new phase, comparatively minor changes have amounted to a new version number. One of the criteria that grant an increment is the compatibility to newer versions of Pd. Cyclone 0.3 needs at least Pd 0.49, whereas Cyclone 0.4 needs Pd 0.50, Cyclone 0.5 needs Pd 0.51 and Cyclone 0.6 needs Pd 0.52. One thing Cyclone 0.6 “needs” from Pd 0.52-0 is that it now carries italic variants of the DejaVu Sans Mono font. Pd does not need these variants to function, but the *comment* object from Cyclone does. The other thing it needs from Pd 0.52-0 is that the *buffer~* object in Cyclone 0.6 allows you to load *.caf* files. This is just an example on how usually just minor features require a minimum Pd version.

Cyclone 0.6 is officially out at the time of this writing, but the final version of 0.52-0 isn't ready yet. It was released when Pd 0.52-0 was still at a 'test-3' version. Our releases usually follow updates of Pd quite quickly, but this time we went ahead and released it while the new Pd version was still at a test phase.

In the recent versions, we have consolidated the goals from Cyclone 0.3. Regarding the main goal of upgrading Cyclone to be compliant to MAX 7, we could still improve the upgrade we did for the *comment* object. Actually, this is an object that will only be partially upgraded. It will still miss functionalities from newer versions of MAX such as the ‘bubble’ format. There's no real impediment for implementing such features other than an understanding that it is not worth the trouble. Nonetheless, several improvements were made to *comment* and the only thing it needs now is a better properties window (as we did for *scope~*), with features like color picking.

Another missing feature is that we simply skipped porting new functionalities to the *mtr* object. We only realized recently that we've missed this, so this is still on the “to do” list. In the latest versions we did fix some bugs from Cyclone 0.3 where we didn't get all of the new functionalities

that we added correctly. Some old known bugs from Cyclone are still on the “to do” list as well. Regression bugs have also been fixed and now there are no known ones.

In our development of Cyclone versions 0.3 and later, we have striven to not break backwards compatibility. However, we have removed capital-letter aliases (such as *MouseState* and *Scope~*) from Cyclone 0.3 due to compilation issues. We began restoring the functionality with Cyclone 0.4 and completed the process with later versions of 0.5 with the caveat of producing an error suggesting the use of their lower-case versions. We don't have significant backwards compatibility issues now and more about this will be discussed in the next section.

7. The future of Cyclone

Cyclone's development has slowed down a lot since 2019. The project has lost steam, but it is still active and with plans of future work. We're mostly still dealing with bugs and will work on maintaining the library. On the horizon, we'll offer binaries for the new “M” chips for macs and also double precision compilations for double precision Pd. Future development plans will consider new functionalities introduced in newer versions of MAX for existing objects in Cyclone, if possible.

We already have new functionalities to port from MAX 8, the current version of MAX. There aren't that many new functionalities, and they are all already part of our “to do” list, but we'll wait until a MAX version 9 is out before porting them all at once. MAX 8 has introduced a new Multi-Channel feature called 'MC'. This would be a nice feature to have in Pd but it'd be just too complicated for us to implement this in Cyclone at the present moment and would probably require core changes to Pd. Alongside Pd's comparative GUI limitations, this is another example of how MAX and Pd are diverging more and more with every new MAX release, so who knows what waits for us in the MAX 9+ releases.

As for now, there are no more ambitious plans for Cyclone. The current concern is to keep it stable and backwards compatible. We also do not anticipate any backwards compatibility issues. This means we also don't have plans to change the current state of the library. We can argue that the current Cyclone is not fully backwards compatible with the original phase of development by

Krzysztof Czaja. That is true, but so it is true for any other version of Cyclone since it was incorporated into Pd-extended. Most of these compatibility issues we have now were inherited, like missing *maxmode* and *dummies* binaries in Pd-extended or the import mechanism found in the *cyclone* and *maxmode* objects being removed for good in the Fred Jan Kraan era.

We did take things further and removed the unused code for the import mechanism, so we got rid of it all for good as a final nail to the coffin. The parallel development of Cyclone by Fred Jan Kraan and the now parallel development of *nilwind* have brought back the *cyclist* binary. However, *cyclist* is only provided for Windows. We did not resurrect *cyclist* for different reasons. First, it seems like a rather arbitrary step back. It'd make more sense to bring back the import mechanism in the *cyclone* object to open *.mxh* files as the *cyclist* binary is not convenient nor friendly to use. You have to run it via a command line and it only gives you the converted text in the terminal window, so you have then to manually generate the *.pat* file.

The other reason is simple: the *.mxh* files are too old now and you need MAX 4 to save them as such. The reasonable thing to do these days would be to update the import mechanism Cyclone had to be able to open (and maybe even save) a more modern MAX file format, like the *.maxpat* extension based on JSON. This could have been part of our goal to update Cyclone, but it wasn't considered worth the hassle due to the complexity of the new format and its major differences with previous formats. For last, the main point is that the little MAX and Pd compatibility we had back in the day got lost throughout the years and we just have to accept this.

Our 2016 paper also mentioned how there were plans to update the Cyclone library in Pd-L2Ork/Purr Data, which are now two separate projects and both still carry an old and parallel version of Cyclone 0.1-alpha56. Currently, we cannot make this commitment anymore. Porting cyclone to Purr Data involves dealing with nw.js and none of us are well acquainted with this tool, nor do we have the available time and resources to invest on this. We haven't even been able to provide a decent properties window for the *comment* object in tcl/tk yet, let alone learn a new language. However, we are open to outside contributions for those willing to tackle the project and would hope maintainers of Pd-L2Ork and or Purr Data had motivations of their own to help and finish the GUI port and include and ship a more up to date version of Cyclone.

8. Final Considerations

Cyclone is one of the most important legacy libraries for Pure Data. It started as a quite ambitious project and aimed to offer excellent portability between MAX and Pure Data. Unfortunately, the initial goal to provide full compatibility between these two environments was not only unfulfilled but this goal has also become more and more unrealistic and unattainable.

One way or another, Cyclone has served as an important library, independent of its original purpose of being a MAX-compatible library. Pd Vanilla has a very limited set of internal objects and relies a lot on the usage of external libraries. Cyclone being a very large library and offering many convenient tools has been a very important factor in its popularity. Because Cyclone was included with Pd-extended, it has been part of countless Pure Data projects. It was also common in the Pd-extended world to not be fully aware of which library an object belonged to and sometimes people would use objects from Cyclone without even being aware of it (an occurrence magnified by the potential lack of overlap between Pd users and Max users). Cyclone filled in a gap that was needed for Pd aside from the MAX compatibility aspect.

Because of its focus on convenient high-level objects, Cyclone has left its footprint in the Pd ecosystem, providing an extensive toolkit that warrants its use in Pd projects on its own. As a consequence, if you don't care about the MAX-compatibility, you may be stuck with Cyclone and its design, meaning that you have to use objects that have the exact same functionality that you'd find in MAX. Sometimes you may wish things were a bit different. Working with Cyclone throughout these years made us realize that Cyclone can be both a blessing and a curse, as there is no freedom to do things differently in a way we consider better. Also, its adherence to the Max API may prove limiting (and counterintuitive to those accustomed to Pure Data's approach). We as maintainers have long struggled with this conflict but have mostly adhered with Max's approach as in the usage of attributes and not allowing floats to be promoted to signals in secondary inlets.

Cyclone still fails to provide 100% compatibility. We're missing some features like being able to set a delay time based on a global transport. We'd need to create one for that, but it didn't seem worth it as this did not provide any considerable advantages and there are other ways to keep a master tempo with other objects receiving a relative tempo.

Alexandre Porres, one of the current maintainers of Cyclone got really frustrated with the compatibility constrains and started including alternatives for Cyclone into his new external library called ELSE. At first, ELSE was supposed to offer missing objects and features in Pd's ecosystem, as to not overlap with other existing libraries (Cyclone included). Eventually, “the more the merrier” idea won and, currently, ELSE provides numerous alternatives to Cyclone. Not only that, but missing objects in Cyclone that would be useful to clone from MAX were also used as inspiration for new objects in ELSE. One example is the use of 'transport'. Porres did expect that this mechanism in MAX would keep objects in sync when changing values, but it doesn't, so he created more sophisticated objects based on this idea, namely *metronome* and *clock*. This is an explanation why we stopped cloning new objects from MAX, and also a reason why Cyclone's development has slowed down, as Porres's development efforts have focused mainly on the development of ELSE.

Despite this, Cyclone still deserves attention. It is a legacy library that Pd users have relied on for almost two decades now. It is part of numerous Pure Data projects and it was an incredible effort and achievement from our part to have fixed so many bugs, worked on new documentation and upgraded this library to keep up with newer developments of MAX, which now allows a much better gateway to Pd from the MAX side than before.

ACKNOWLEDGMENT

Krzysztof Czaja's for his incredible initiative that is living on for so long. Hans-Christoph Steiner and Fred Jan Kraan for keeping it up before us. Katja Vetter for pd-lib-builder, the new standard for building Pd externals. Everyone who collaborated with us in our repository, specially Lucarda for the work on building for Windows. Last but not least, Miller Puckette, of course, who can't be left out simply for creating both Max and Pd.

REFERENCES

PORRES, Alexandre; KWAN, Derek; BARBER, Matt. Cloning Max/MSP Objects: a Proposal for the Upgrade of Cyclone, *Proceedings of PdCon16*, NYU, New York, 2016.

ABOUT THE AUTHORS

Porres is a composer, performer and computer music teacher from São Paulo, Brazil with a PhD in computer music. He's a Pd enthusiast since 2005, organized the PdCon09 in São Paulo and runs EL Locus Solus, which organizes cultural events and music courses. E-mail: porres@gmail.com

Matt Barber is a composer, performer, and teacher currently residing in Rochester, NY. His music has been performed by the Colorado Symphony Orchestra, the Juilliard Orchestra, the Ossia New Music Ensemble, Musica Nova, the New York Miniaturist Ensemble, Tetrawind, the Arapahoe Philharmonic, and other ensembles across the United States. He has avoided adopting a particular style of composition, and every new piece represents a different and original compositional interest. Matt is an engaging performer specializing in bassoon and recorder, and is in demand as a conductor for music from Bach to Xenakis. E-mail: brbrofsvl@gmail.com

Derek Kwan is a percussionist and a computer science graduate student who now takes a graduate course also in computer science. His creative output includes live coding electronic music, interactive audiovisual installation art, physical computing, acoustic and electronic music composition, and percussion improvisation. As a performer, Derek has appeared with the Bang on a Can Marathon, Make Music New York, and soundSCAPE festival and has performed with Raphael Mostel's Tibetan Singing Bowl Ensemble and Iktus Percussion. He has had music compositions featured at the New York City Electroacoustic Music Festival, the UC Davis Shinkoskey Noon Concert series, and Glasslands Gallery (Brooklyn, NY). E-mail: derek.x.kwan@gmail.com

APPENDIX

Original cyclones readme from the 0.1 version era:

Cyclone is a library of PureData classes, bringing some level of compatibility between Max/MSP and Pd environments. Although being itself in the early stage of development, it is meant to eventually become part of a much larger project, aiming at unification and standardization of computer musician's tools.

In its current form, cyclone is mainly for people using both Max and Pd, and thus wanting to develop cross-platform patches. In this respect, cyclone has much in common with Thomas Grill's flext, and flext-based externals. See Thomas' page. While flext enables developing new cross-platform classes, cyclone makes existing classes cross-platform.

Cyclone also comes handy, somewhat, in the task of importing Max/MSP patches into Pd. Do not expect miracles, though, it is usually not an easy task.

The entire cyclone library, which might be preloaded with either `-lib cyclone` or `-lib maxmode` option, consists of:

- * the main hammer and sickle sub-libraries, containing Pd versions of, respectively, Max and MSP classes;
- * cyclone sub-library, taking care of loading hammer and sickle, and which itself contains: a small set of operators (`!-, !/, ==~, !=~, <~, <=~~, >~, >=~~, !~-, !/~, %~, +=~`); an importing mechanism provided by the cyclone class.
- * optional dummies sub-library, which contains a large set of dummy classes, serving as substitutions for Max/MSP classes not (yet) implemented in cyclone;
- * maxmode sub-library, which imposes strict compatibility mode, and loads all the other components, including dummies.

The two main sub-libraries might be loaded separately, by using `-lib hammer` and/or `-lib sickle` options. There is also a possibility of loading any single class from hammer or sickle library dynamically (this feature is only available in the linux snapshot).

Currently, the hammer part contains: accum, acos, active, anal, Append (more info), asin, bangbang, bondo, Borax, Bucket, buddy, capture, cartopol, Clip, coll, comment, cosh, counter, cycle, decide, Decode, drunk, flush, forward, fromsymbol, funbuff, funnel, gate, grab, Histo, iter, match, maximum, mean, midiflush, midiformat, midiparse, minimum, mousefilter, MouseState, mtr (more info), next, offer, onebang, past, Peak, poltocar, prepend (more info), prob, pv, seq (more info), sinh, speedlim, spell, split, spray, sprintf, substitute, sustain, switch, Table (more info), tanh, thresh, TogEdge, tosymbol, Trough, universal, urn, Uzi,

xbendin, x bendout, xnotein, xnoteout, and zl.

The sickle part contains: abs~, acos~, acosh~, allpass~, asin~, asinh~, atan~, atan2~, atanh~, average~, avg~, bitand~, bitnot~, bitor~, bitshift~, bitxor~, buffir~, capture~, cartopol~, change~, click~, Clip~, comb~, cosh~, cosx~, count~, curve~, cycle~, delay~, delta~, deltaclip~, edge~, frameaccum~, framedelta~, index~, kink~, Line~, linedrive, log~, lookup~, lores~, matrix~

(more info), maximum~, minimum~, minmax~, mstosamps~, onepole~, peakamp~, peek~, phasewrap~, pink~, play~, poke~, poltocar~, pong~, pow~, rampsmooth~, rand~, record~, reson~, sah~, sampstoms~, Scope~, sinh~, sinx~, slide~, Snapshot~, spike~, svf~, tanh~, tanx~, train~, trapezoid~, triangle~, vectral~, wave~, and zerox~.

Cyclone comes without any documentation. All the included patches were created merely for testing.

Caveats:

* The binaries provided in this snapshot release are not expected to run inside of a pre-0.36 version of Pd, without prior recompiling.

* If a single -lib cyclone startup option is used, cyclone in turn loads its two main components: hammer and sickle. If a single -lib maxmode startup option is used, all the remaining library components are going to be loaded: cyclone, hammer, sickle, and dummies. In these cases, all the required libraries should be accessible by Pd.