A QUESTION OF CHARACTER

How do we automatically recharacterize data at cloud scales?

Jack O'Sullivan
Preservica Ltd
UK
jack.osullivan@preservica.com
https://orcid.org/0000-0002-0306-761X

David Clipsham
Preservica Ltd
UK
david.clipsham@preservica.com
https://orcid.org/0009-0006-2611-8877

Divyesh Soni
Preservica Ltd
UK
divyesh.soni@preservica.com

Richard Smith
Preservica Ltd
UK
richard.smith@preservica.com

Jonathan Tilbury
Preservica Ltd
UK
jonathan.tilbury@preservica.com

Abstract – Many preservation actions that we undertake on digital content are driven by the format of the content in question. Format information is often determined at the point of ingest and is not regularly updated as our knowledge of file formats improves over time. Periodically re-characterizing all content in a repository would ensure that we get more accurate identifications over time, but a more sustainable approach would be to only re-characterize content that was actually likely to have changed. Preservica's new Automated Active Digital Preservation feature seeks to do exactly this, but even when considering only subsets of the data in our cloud systems, we are faced with significant challenges of scale. In this paper, we describe those challenges, the approach we have taken to implement the feature, and the testing we have performed to verify the viability of this approach.

Keywords – Scalability, Automation, Characterization, Preservation Actions

Conference Topics – From Theory to Practice; Sustainability: Real and Imagined.

I. INTRODUCTION

Characterization is one of the fundamental bases of Digital Preservation. It is the process of identifying the types of digital material we are preserving, and extracting the relevant technical characteristics and significant properties of that material [1]. This understanding of our content drives many digital preservation processes and policies; it might inform how and where we store the content, what normalizations, if any, we perform, what access copies we need to generate, and how we display content to end users. Its importance is such that it is an assumed standard part of our digital preservation processes, with at least the identification part of it even being part of the "Parsimonious Preservation" workflow [2].

Characterization is often treated as part of the ingest process, or preparation for the ingest process [3], and it is true that performing characterization up-front has benefits. Until we know what our digital material is, we can't apply format based policies, or take format based preservation actions such as

iPRES 2023: The 19th International Conference on Digital Preservation, Champaign-Urbana, IL, US.

https://hdl.handle.net/2142/121087

normalization or the creation of access copies. However, our collective understanding and knowledge of file formats changes over time, as do the tools available to identify and validate content, and to perform extraction of technical properties. If all we have is the knowledge of how our content was identified at the point of ingest, and the characteristics we could measure with the tools then available, then our decision making about all subsequent preservation actions may be flawed.

Ideally, our content should be characterized with the latest file format knowledge and most up to date tools at all times.

If re-characterization is a process that must be manually undertaken, this places a burden on the user/s of the system to ensure that this happens. These users are often archivists and collection managers rather than digital preservation experts, and as such are not always the people best placed to determine what needs to be re-characterized and what does not.

An alternative approach would be to automate re-characterization on a periodic basis, in the way that we might perform fixity checks, in order to ensure that our information up to date. However, this potentially requires a lot of compute time, and will, more often than not, result in no changes needing to be made.

Preservica has developed a feature that ensures that the preservation system itself can automatically respond to recommendations made by digital preservation experts to ensure that the correct subset of repository content is re-characterized as appropriate. This removes that burden from non-expert users of our systems, and means we only run processes on potentially affected content.

In this paper, we will discuss how even this approach results in challenges of scale when applied to production systems. In section II we will discuss what these challenges are. In sections III and IV we will discuss our approach and what steps we took to verify that it would work at the scales required and in section V we will discuss how well this matched the performance we saw when taking this feature into production.

II. What Do We Mean By Scale

A. Scale of the Format Problem

A blog post in 2018 [4], investigated the specific case of how PDF identification within PRONOM and DROID had evolved and demonstrated that the identification outcome of a corpus of PDF files changed over time. This is a natural consequence of the fact that PRONOM's data changes over time, usually for the better, as PRONOM's global community of contributors feedback their expertise into the dataset.

This was explored further in a poster for iPres 2019 [5] which additionally examined historical changes to the GIF, TIFF, and JPEG PRONOM-based identification.

However, PRONOM contains details of over 2250 file formats as of March 2023, so it is necessary to evaluate changes across the entire dataset to get a complete understanding of the impact of these changes.

Carrying on from the Lightning Talk last year [6], we investigated changes in PRONOM going back to the very earliest versions, with the PRONOM v10 update in 2006 chosen as a starting point as this was the first release where every single format entry had a persistent 'PRONOM Unique Identifier' (PUID) assigned.

Of an initial assessment of 1,089 unique file formats represented across the Preservica Cloud estate as of March 2022, we found that 489 format definitions (approximately 45% of those assessed) have changed at least once in such a manner that they warrant a re-identification event.

All of these recommendations have been made publicly available and as new recommendations are made these will continue to be published for the benefit of all.

Format definitions change in PRONOM for a few reasons:

Name or version updates: These are often relatively trivial, so a format name might be updated to correct a misspelling or to match official branding. A format version might be adjusted to cover multiple software releases or adjusted to a default 'generic' entry that is used in the event of a format being unable to be identified as an exact, specific version. There can be more impactful changes, however.

In the case of the database preservation file format, the Software-Independent Archiving of Relational Databases format, or SIARD, when the

format was originally added to PRONOM in 2009 the entry was given the version number 2, although version 2 of the format wasn't formalized as a standard until 2015. In 2014, on the advice of the Swiss Federal Archives who created the original file format, the original entry in PRONOM was adjusted to version 1.0. Subsequently in 2016 SIARD version 2.0 was added to PRONOM. As such two separate PRONOM entries have been called 'SIARD 2.0' at separate times, therefore file instances that were most recently identified before the 2014 correction will need to be re-identified to ensure they have the correct identification and to avoid confusion and ensure proper management.

In a separate case, the image file format '3D Studio,' introduced in one of the earliest versions of PRONOM before version 10, had its name changed to 'Paint Shop Pro Image' for reasons unknown around 2012. This was likely a mistake, as it was changed back to '3D Studio' in 2015 but this means that any file instances identified as such during this time period will need to be re-identified.

Up to the version 109 PRONOM update in November 2022, 301 updates to format name and/or version number have taken place.

Deprecations: Once a PRONOM entry has been created, it is intended to persist, so entries are not permanently deleted for any reason, however sometimes an entry may no longer be suitable for use, at which point it is deemed 'deprecated' and disassociated from identification mechanisms such as extension or file format signatures. Particularly in the early days of PRONOM there were several entries added that really related to specific software versions rather than file format versions and subsequent research deemed many of these unnecessary and with the potential to cause unintentional and unwanted identification clashes.

In the case of the Tagged Image File Format, or TIFF, PRONOM originally had distinct entries for versions 3, 4, 5, and 6, however each entry shared a single identification signature, meaning a file format identification tool would identify a file instance as each of these four formats, which could cause confusion or uncertainty, however it wasn't clear how to distinguish between these format versions reliably. A decision was made to deprecate these four entries and create a single general one. As such any file instances that were identified before these deprecations were made, should be re-identified to ensure they get the current correct identification outcome.

As of PRONOM's version 109 update, 68 file format entries have been deprecated.

Changes to format priorities: Further significant sources of change within PRONOM are 'priority relationships.' Many file formats are based on other file formats and some formats share certain characteristics of others. In these cases, it may be the case that these shared characteristics, where used for file format identification, will clash and would result in a file format identification tool matching against each format rather than a specific one. This situation is handled through setting a 'priority relationship,' where the more specific format is given priority over the more general one.

A new priority relationship being introduced will usually necessitate some form of re-identification as the previously general format identification outcome may now result in a more specific outcome if reassessed. A common case is where camera image formats, such as the Nikon NEF, the Pentax PEF, and similar file formats which are often based upon the TIFF file format, are introduced. Since these would have previously been identified as TIFF, it follows that any previously identified TIFF files should be re-identified as these may now get a more specific identification outcome. This is an instance that would need to be handled with care however, as many digital preservation repositories will store many millions of TIFF files.

In a separate case, when the Video Object Format (VOB) was introduced to PRONOM in 2012, it was given a *lower priority* than the MPEG Program Stream video formats from which it was derived. This was a mistake, as VOB is the more specific format so it should have been given a higher priority. This mistake was corrected in 2014 but means that any file instances that were identified as MPEG-1 or MPEG-2 Program Stream during this time need to be re-identified as they may have instead been VOB files.

As of the version 109 PRONOM update there are 1,054 priority relationships in-place, with 191 formats set as 'lower priority' than one or more other formats.

Changes to identification signatures: The final major trigger for file format re-identification will be

where file format identification signatures are changed.

This usually happens where a previous signature has been found to be a little loose in order to tighten the signature, however it can sometimes be the opposite, where a previous signature has been a little too strict. This could also be correcting a prior mistake.

A signature update will not necessarily require a new re-identification as in many cases optimizing a signature will not adversely affect a prior identification outcome, but mistakes will usually necessitate them.

In a recent instance, an attempt to slightly loosen up the signature for Encapsulated PostScript (EPS) version 2.0 went awry – the intention was to replace three specific bytes with wildcard bytes (bytes that can have any value) to allow for a little variance that had been observed in some file instances. Mistakenly the sequence was replaced with two wildcard bytes rather than three, which meant that affected files would then erroneously identify as standard PostScript rather than Encapsulated PostScript. This issue was quickly rectified within two months, but once again, any file instances that were identified as PostScript during this time will need to be re-identified.

From version 10 to the version 109 PRONOM update, 594 signature sequences have been altered.

B. Scale of the Content Problem

Preservica has been running commercial, cloud-based digital preservation systems for over a decade; starting with a single, multi-tenant system in the US, we now operate tens of systems across multiple regions of the world. Some of these are "private cloud" systems, hosting services and data for a single organization, others are multi-tenant, with tens, hundreds and even thousands of organizations sharing resources. We have customers who have been using these systems continuously for the entire lifetime of the service, meaning that we have production data that was ingested over ten years ago.

As of October 2022, we have over 116 million digital objects stored across our cloud estate. Our largest individual tenancies each have over 10 million assets stored.

Of these files there are approximately 1,350 file formats represented across the estate. The top ten most common file formats present make up over 90 million assets, approximately 77% of files stored. The most common types of file format present are images, documents (including PDF), and email.

We have over 32 million TIFF files stored, and a similar number of the various JPEG file format variants. There are over 20 million PDFs, including over 2 million PDF/A files. There are approximately 4.5 million emails.

However, the long tail is very real and very long. 664 file formats have 100 or fewer assets stored. 1,056 file formats have fewer than 1,000 assets. The 1,000 least populous file formats make up just under 110,000 files stored, less than 1% of the total, and although 1% seems like a very small number, 110,000 is more files than many of our individual tenants have in total.

The diversity of file formats present truly reflects the diversity of our user-base. Among these file formats we see rare and interesting eBook formats such as Broad Band LRF, or the Rocket Book eBook format. We see many different variants of Flash, which was once extremely common but due to security issues is no longer supported by most mainstream content platforms. We see ancient image formats such as PCX and TGA, but also extremely modern ones such as HEIF and JPEG XL.

Some proportion of this content will have been tentatively identified. This means that it didn't match any byte sequences for any file formats, and was assigned an identification on the sole basis of the file extension. Whilst we know this must be true for some content (e.g. the plain text file format x-fmt/111 has no byte sequences to match), the raw format data we have analyzed does not tell us this for other formats where byte sequences do exist. Some of the changes made to PRONOM in the time since any such content was ingested might mean that today we would be able to provide a firmer identification on the basis of matching byte sequences.

For example, the OS/2 Presentation Manager Metafile file format was originally added to PRONOM in 2005, and was associated with the .met extension so any file instances with that extension will have received a tentative identification outcome. In the v108 PRONOM update in 2022, a new identification

signature for this file format was created, meaning we can now re-identify these file instances and either definitively and positively identify them as OS/2 Metafiles, or for those that are not OS/2 Metafiles, focus file format identification research efforts to further improve PRONOM.

We also have approximately 700,000 (approximately 6% of the total) 'unidentified' file formats stored, that is files for which we were unable to positively assert the file format identity *at the point of ingest*. Since the time period for these ingests stretches many years and PRONOM coverage is continually improving, the real current number is likely to be lower, but this can only be measured through re-identification.

These counts are only looking at the cloud services that Preservica actively manages. We have a number of "on-premise" customers who themselves manage similar sized repositories. Our on-premise offering pre-dates our cloud offering by around a decade, and so some of these customers have content ingested over even longer timescales.

III. Process

The approach we have taken to this problem of re-characterization at such scales is to separate responsibility for determining what content needs to be re-characterized from responsibility for actually running the process. Further, we have removed both responsibilities from the typical non-expert users of Preservica.

A. Identifying Changes

Preservica now allows a Digital Preservation expert to produce "Recommended Processes" [7], which describe the type of process to run and filters to describe the subset of content to run against. These filters include:

- lists of file formats to specify that only content matching one of the formats should be processed;
- event/date ranges to specify that only content ingested or last characterized between certain dates should be processed;
- whether unidentified, or tentatively identified content should be processed.

These recommendations are written in JSON format, consistent with the Preservation Action Registries (PAR) data model [8], and published to a Preservica Registry using an API that is consistent with the PAR API definition [8].

B. Executing Processes

Once published, these processes will be automatically executed by Preservica's *Automated Active Digital Preservation* (ADP) feature.

Preservica's architecture allows for individual "mini-services" to be containerised and deployed as consumers of specific messages brokered by a message queue. Specifically, these are implemented as Docker containers, and can be deployed in a scalable manner using a service such as Kubernetes.

As well as allowing for the independent scaling of each mini-service, this deployment model also means that each mini-service can be deployed in an isolated manner, allowing us to avoid resource contention with other parts of the system.

The orchestrator for Automated ADP is one such mini-service, whose function is to watch the Registry for new Recommended Processes, and then query the repository to get a list of Assets that match the criteria in the recommendation. Once this list is generated, it posts a message for each Asset, requesting a re-characterization. These messages are consumed by a separate mini-service, dedicated to performing characterization.

This means that during periods where large numbers of re-characterization processes are requested, we can scale up the number of mini-service instances dedicated to running them. Conversely, once the demand has died down, we can scale back down, meaning that we only use computing and memory resources as we need.

The execution of these processes is explicitly designed to be a background activity that does not necessarily surface to the users of the system. However, it is still useful to be able to track them as they happen, and so each process that is executed is also monitored. This allows us to record general progress updates that detail how far through the list of Assets we are, as well as data and/or process specific error messages (such as forwarding error messages from the characterization tools themselves).

IV. Scale Testing

In order to ensure that this process would be viable, we undertook a program of scalability testing, with a view to replicating the typical scales seen by our cloud systems. This was largely achieved using two distinct testing regimes. The first was "code-level" integration tests, which gave us tests we could spin up one demand on local development machines, and where we could actually debug into individual processes. The second was to create an actual cloud environment using production level hardware specifications, and populated with large volumes of data.

A. Integration Performance Tests

At the lowest level, we created performance testing at a code level, writing integration tests that configure and deploy the relevant set of mini-services, populate a test database with data, and then trigger background re-characterization processes. For the sake of simplicity, these provide dummy implementations for dependencies like archival storage; only create data that we intend to re-characterize; and use the same input content for each database record.

This means that we are not using them to derive realistic or expected production performance metrics, but they do allow us to quickly run tests with increasing volumes of content to determine where bottlenecks may emerge.

They prove exceptionally useful in replicating issues uncovered in the more realistic test scenarios, allowing us to diagnose those issues, and have some confidence that we have actually resolved them.

B. Production Like Test System

The second and main testing mechanism we used was to create an actual cloud environment using production level hardware specifications, and populated with large volumes of data. From here we could publish realistic recommendations and allow the system to run through re-characterizations in a real world scenario.

This system was loaded with close to 345,000 pieces of content in 763 different file formats (plus around 29,000 "unidentified" formats). As with our production systems, this was heavily weighted to common formats, with over 53,000 JPEG 1.01 files and over 24,000 Word 97-2003 files. The top 10 file formats accounted for over 55% of all content.

By combining formats in our recommendations, we could create processes that would target an arbitrary number of assets to re-characterize. We published a series of recommendations, triggering re-characterization processes on increasing numbers of assets, from tens at a time up to just under 100,000. By querying the monitoring API and underlying database, we could calculate the rate at which these re-characterizations were performed.

For this initial round of testing, we did not perform any scaling of any of the mini-services involved, so at any given time, there was only instance of a mini-service running.

C. Results

The predominant finding from this was that over increasing scales, the rate at which we were able to process re-characterizations did hold relatively constant.

In the majority of test cases run, the rate, as measured by the overall running time of the process divided by the number of assets processed, was less than 1 second per asset. (varying between around 0.1 and 0.7, but averaging around 0.25). In the final iteration of the code, this held true up to the largest dataset we tested, which was in excess of 96,000 assets being re-characterized in a single process.

This is not to say that characterization of any given asset took less than 1 second, since, even though there was only one instance of a mini-service, internally it runs up to 8 threads simultaneously, so 8 assets, each taking 8 seconds to process would still result in a rate of 1 asset per second.

This parallelism benefit could in fact been seen in one of the smallest tests we ran where just 17 assets were being processed. The rate for this test was 5.6s per asset. On closer examination we determined that this was essentially a "small sample effect"; one of the test files was orders of magnitude larger than the others (around 3.5GB), and the overall process time was dominated by the retrieval of this content.

At this rate of less than 1s per asset, processing of up to around 100,000 assets will run for approximately a day, which is well within the comfort zone of being able to generally assume full system uptime.

D. Issues Uncovered

The first issue we encountered was at around 15,000 assets being processed. The rate jumped from less than second per asset to over 3s per asset. The process reported a lot of errors that were ultimately due to calls to Third Party characterization tools being timed out (i.e. cancelled when they took >30s to return). Although this initially seemed like it might be to do with overwhelming the mini-services, the actual root cause was discovered to be a scalability limit in our "working area" shared storage.

In order to run characterization tools against the content in the repository, we take a copy of the content from its archival storage location (in this case an AWS S3 bucket) and place it in some storage that is accessible to all mini-services (in this case an AWS EFS drive). The throughput on the EFS is throttled by default, giving you an allowance that you use when performing reads or writes to disk, and which replenishes over time when no activity is taking place. At this scale, we were using up all of the allowance without it being able to recover. At that point, all I/O operations became slower than we were able to tolerate. This is relatively trivial to fix, albeit at increased service cost.

The second issue was also due to the same EFS system, or at least, how it was "mounted" in the mini-services, and hit at around 25,000 assets. To reduce network costs, each client connecting to the EFS drive maintains a local cache of what is on the drive. In real time terms, these caches are short-lived and so once a client has written content to the drive, all other clients will "see" the content very shortly thereafter. In our case however, the messaging between mini-services was quick enough that the code that should use the content was trying to read it before its cache updated, then compounding this issue by storing this "not found" result in cache for long enough that eventually the process was timed out. Whilst this was likely happening on smaller scale tests, only at this point did it cause an appreciable impact on our results.

The final major issue that we encountered was to do with the way the processes were being monitored. This presented as an inelastic threshold in our testing. The rate of processing held constant up to around 80,000 assets, at which point, the Automated ADP orchestrator service became very unstable, restarting frequently, causing monitoring to go awry and process requests to be re-sent multiple times.

The limit here was essentially that each time a process completed, we were attempting to update the monitoring information to indicate how far through the process we were. In doing this, we were retrieving a list of the requests, then aggregating them by their process status so that we could update these numbers in the database. There were two issues with this, the first is that at some point, the volume of data contained in the list of requests became large enough that the SQL query to retrieve it would take a long time to complete. The second is that because we were operating 8 processes in parallel, we would often have 8 threads making that call simultaneously. This combination caused contention for database resources, which ultimately cascaded into a series of timeouts and errors.

The issue of counting lots of simultaneous updates in a transactional manner is a common problem in large scale systems, and the general solution is to reduce the number of times you actually update progress, caching all the updates in memory in between. The updates in question here were purely for monitoring, and in large scale processes it is generally acceptable to see updates at longer discrete intervals, so we were able to solve this issue by a combination of performing the status aggregation in the database query (thus reducing the volume of data we needed to transfer), and by only updating periodically (thus reducing the number of database calls we needed to make).

E. Testing Limitations

The system we ran our testing on was configured as a production system would be, with the same hardware specifications, so the direct performance results should be comparable. However, we were limited in how far we could fully replicate a production system in the time available.

At over 345,000 pieces of content, this system was larger than a number of our production systems, but at least an order of magnitude smaller than the largest systems we have. The data also contained many more duplicated items than we would reasonably expect a production system to contain. This introduces some uncertainty into validity of the process. Some data will cause issues with third party tools that other data in the same identified format will not, possibly due to the use of features of that format, or just whether it is valid content. If our dataset contains lots of replicas of problematic data,

then this might mean that our measured rate is over-estimating how long a truly heterogeneous data set of the same size would take. Similarly, if it is replicating more "clean" data than would exist in a truly heterogeneous set, then we might be under-estimating how quickly we would process that set.

The final limitation we have identified is that our test system was configured to be single-tenant, whereas many of our biggest systems are multi-tenant. The system is designed to run processes on a tenant by tenant basis, which means that the number of tenants in the same system should be irrelevant, however this set of tests was not designed to explicitly verify this.

V. Into Production

Following on from this successful scale testing, we have started to roll this feature out into live production systems. At the time of writing, this has been limited to around 10 recommendations, across two production systems, reaching scales of up to around 15,000 assets being re-characterized in a single process. Taking the same rate measurements as we did for the testing processes, our performance has been between 0.2 and 0.25 seconds per asset, which is perfectly in line with the results from the test systems.

We will be continuing to enable this feature on more systems, and publish more recommendations over the coming months.

VI. Conclusions

We have reported on a large-scale issue that that affects the users of Preservica's cloud systems, namely that it likely that some proportion of the content they have ingested has outdated characterization information. The result of this is that we are likely to make poorly-informed decisions as to how to treat this content; particularly we may repeatedly attempt to perform processes, such as rendering or migration, that have no prospect of success, which will harm our efforts to preserve information efficiently.

We have discussed the general approach we have taken to implement functionality within Preservica to address this issue. We are allowing Digital Preservation experts to publish machine actionable recommendations for re-characterization processes that should be run, and then automatically executing those within a scalable architecture.

It is noted that for now, assessing updates to the PRONOM dataset as they are formally released is a task that is carried out manually by digital preservation experts using the tools and approaches created in-house for the task.

The additional workload this requires will scale with the number of file formats in the PRONOM database, and the number of types of underlying digital content these represent. This is independent of the volume of content in any given system. For context of the current scalability of this task, PRONOM updates are comparatively infrequent (2 to 3 per year), which limits the frequency at which such analysis has to be performed, and although the sizes of updates vary, they are comparatively small, affecting tens to a few hundred formats each. This makes it possible for a single individual to assume responsibility for this task at each update.

This work is currently being performed by Preservica staff as part of our ongoing digital preservation activities. The output from the analysis is being published for the benefit of the community at

Since the types of data changes that may warrant a re-identification recommendation have so far proven to be relatively systematic, it would likely be possible to augment this process through further automation, perhaps through machine-assisted or machine-learning-based approaches, however exploring these approaches is beyond the scope of this paper.

Over time, it may be necessary to partition this workload so that experts in different types of digital content are responsible for making recommendations related to their expertise (e.g. one expert assessing the impact on images, whilst another assesses the impact on Audio-Video content). Again, this is beyond the scope of this paper.

The mechanism of allowing digital preservation experts to publish recommendations written in a PAR-like data model, and using a PAR-like API means that it should be possible to extend PAR to encompass this in the future. This would enable experts and practitioners from across the digital preservation community to publish their own advice, and access that of others, in a machine actionable

way. This would further extend the benefit of this work and enhance knowledge sharing for the entire community and not just Preservica users.

We have presented a description of the testing we have undertaken to validate that this approach will indeed be able to meet the scale of the challenge, summarizing the key results of that testing, and highlighting the key issues uncovered. We have also reported some initial confirmation from production implementation of this feature that our test findings are in line with the performance we are able to achieve on live systems.

We clearly have further work to do in rolling this feature out more generally across our cloud estate, and this work is currently in progress.

The next step in our Automated ADP feature implementation is to enable similar automation of expert derived recommendations around migration functionality. Much of the testing we have already performed will be valid for this also as much of the triggering and monitoring mechanisms are shared. Typically however, migration itself is a more compute and memory intensive process then characterization, so there are still outstanding questions of scaling these processes to answer.

VII. References

[1] M. Hutchins, "Testing Software Tools of Potential Interest for Digital Preservation Activities at the National Library of Australia," National Library of Australia, Canberra, 2012.

[2] T. Gollins, "Parsimonious Preservation: Preventing Pointless Processes!," in *Online Information 2009*, Online, 2009.

[3] The National Archives, "Digital Preservation Workflows > 2. Ingest," The National Archives, [Online]. Available: https://web.archive.org/web/20221202160658/https://www.nationalarchives.gov.uk/archives-sector/projects-and-programmes/plugged-in-powered-up/digital-preservation-workflows/2-ingest/. [Accessed 02 12 2022].

[4] Y. Tunnat, "Sherlock Carriage – PRONOM's blind spot on (some) PDFs from 2010 to 2014," 25 July 2018. [Online]. Available: https://openpreservation.org/blogs/sherlock-carriage-pronoms-blind-spot-on-some-pdfs-from-2010-to-2014/. [Accessed 19 June 2023].

[5] Y. Tunnat and M. Lindlar, "Time-travel with PRONOM: The 4th dimension of DROID," in *iPres 2019 Conference*, Glasgow, 2019.

[6] D. Clipsham, "PRONOM & Preservica's Auto-Preservation functionality," in *iPres 2022 Conference*, Glasgow, 2022.

[7] J. O'Sullivan and J. Tilbury, "Using preservation action registries to automate digital preservation," *Journal of Digital Media Management,* vol. 9, no. 3, pp. 240-252, 2021.

[8] Open Preservation Foundation, "PAR Overview," [Online]. Available: https://parcore.org/. [Accessed 20 06 2023].