Report of the Format Notification and Obsolescence Service (AONS II)

David Pearson (National Library of Australia)
Matthew Walker (National Library of Australia)

http://www.apsr.edu.au/aons2/report

November 2007

Table of Contents

Introduction

This document reports on the outcome of the Australian Partnership for Sustainable Repositories (APSR) AONS II software development project. During the 2007 development cycle, the project aimed:

> "To refine the Automatic Obsolescence Notification System (AONS) developed in an earlier stage of APSR, to a platform-independent downloadable tool that automatically provides information from authoritative international registries to support decisions on preservation action required to retain access to information resources stored in repositories".

The software developed during this project was intended to serve the national interests by providing infrastructure to improve the long-term preservation of digital materials in:

- Local standalone repositories;
- Networked enterprise repositories; and
- Federated large scale repository structures.

The target repositories for the use of this software exist in education, research and library sectors but, as an open source product there are many more potential users. The basis for this project was the 2006 APSR AONS I prototype software project which was a collaboration between the NLA and the Australian National University (ANU), the software developer. The AONS II project refined and extended the work of this earlier project.

Document structure

This document provides more detail on the background and context for the project and then looks at outcomes by answering the following questions:

- What did we want to do?
- How did we approach the work?
- What did we actually do?
- What changes did we make to the original deliverables?
- What lessons did we learn?
- What needs to happen next?

Background and context

File format obsolescence is a major risk factor threatening the sustainability of, and access to, digital information. While the preservation community has become increasingly interested in tools for migration and transformation of file formats, the AONS II project focused on developing mechanisms specifically for monitoring and assessing the risks of file format obsolescence. The AONS II project was undertaken by the National Library of Australia (NLA) in conjunction with the APSR. The project aimed to develop a pilot service allowing users to automatically monitor the status of file formats in their repositories, make risk assessments based on a core set of obsolescence risk questions, and receive notifications when file format risks change or other related events occur.

The most important direct antecedent for the AONS II project was the PANIC (Preservation Webservices Architecture for Newmedia, Interactive Collections and Scientific Data) model proposed and explored by Hunter and Choudhury. This model recognised that there are many elements in the process of providing meaningful

access to digital materials, and that almost all are subject to change. The approach grew out of a perception that it can be difficult for collection and repository managers to keep themselves fully informed of changes that might threaten the accessibility of their collections. The development of PANIC was based on the emergence of three potentially powerful components that could be brought together for the benefit of repository managers in their preservation planning:

- Information registries which store useful information about file formats;
- Preservation action tools (such as migration services, emulation services, etc) that may pre-empt, circumvent or remedy the impacts of these changes; and
- A global information network in which it should be possible to look for relevant indicators of file format obsolescence, and to promptly bring that information to the attention of repository managers so that they might make informed decisions about the need for preservation action. The same network could also allow them to discover and utilise preservation tools and services to address their needs remotely.

The PANIC model was explored by Dr Hunter and her colleagues, who prototyped an environment in which it would work.

Many collecting institutions responsible for managing digital data for long-term accessibility, including the NLA, were excited by the potential of the PANIC model for reducing duplication of effort in managing preservation systems. While format obsolescence was recognised as just one of many risks to be negotiated, it did seem to be one that was both particularly critical and particularly amenable to the kind of approach PANIC was exploring. Using this model as a basis, in 2006, the Australian National University (ANU), the software developer, built the AONS I prototype as part of the APSR project. Some input was provided by the NLA. The AONS I work focused on the "obsolescence identification and notification" element of the PANIC model.

The NLA wished to see further development of the AONS I tool to test and, if necessary, refine the underlying assumptions so that the methodology could reach its maximum potential as a preservation enabler. Thus, in 2007, the NLA and other partners collaborated in the AONS II software development project.

A number of fundamental principles evolved from the development of AONS I. The AONS II software product was required to:

- Support three different business environments: a national federated infrastructure, enterprise business models, and individual standalone repository sites;
- Be open source using Java code;
- Be modular and have a reusable/adaptable design;
- Be platform independent using a decoupled approach;
- Be interoperable, using common interfaces, protocols and standards;
- Provide service interfaces in a Service-Oriented Architecture based on a REST-ful approach (a lightweight methodology for Web Services);
- Provide a core set of functionality, which abstracts repositories and registries functionality away from the core, and would allow new repository and registry adapters to be added without affecting the core; and
- Be demonstrable.

These principles provided a yardstick and reality check for all development work. In line with the above scope and design principles, at the end of the project, AONS II was delivered as a workable product available for download from SourceForge.

What did we want to do?

The 2007 APSR/NLA AONS II software development project was an APSR COSI project (Collections Services and Infrastructures – NLA COSI B) initiative to provide value-added services for repository managers and maintainers.

The APSR project managed all of the deliverables as a part of the COSI project. The original project deliverables are presented in Appendix A. These deliverables were re-assessed throughout the software development cycle. As a result of this process there was an increase from the eight original deliverables specified by APSR in Nov 2006 to the 12 deliverables articulated in the mid-year report cycle in June 2007 (see Appendix B). These changes were brought about as a result of implementation experiences and changes in dependencies.

How did we approach the work?

The project commenced by identifying the team and drafting a project plan. APSR funding allowed for a part-time project manager and AU$75,000 for the development of services. This funding was extended by AU$25,000 for the abovementioned additional deliverables, as well as testing and documentation. There was an expectation that the partners would provide an in-kind contribution to the project.

The resulting project team consisted of a number of NLA and contract staff. For the entire project, David Pearson was the project manager. Matthew Walker from the NLA Collection Infrastructure Branch, in his role as the technical lead, assisted in a part-time capacity with the development of the business requirements and software architecture, and provided project assurance as required. David Levy (Frontier Group) was contracted at an hourly rate as the software developer. At the start of the project David Levy worked in a full-time capacity, changing to part-time during the testing and documentation phase. Throughout the project the team was assisted by various NLA, APSR and other stakeholders. The APSR AONS II Workgroup consisted of: Colin Webb and Gerard Clifton from NLA; Chris Blackall, David Berriman, James Blanden and Scott Yeadon from APSR; Andrew Treloar from ARROW; and Jane Hunter, Matt Smith, Christiaan Kortekaas from University of Queensland (UQ). These workgroup members met on a number of occasions to discuss the development work.

The first stage of the project consisted of gathering requirements and use-cases from NLA, APSR and APSR partners. As David Levy could not start work until February, David Pearson and Matthew Walker worked on the use cases, software architecture and software component requirements. Based on this work they designed an architecture which was compliant with the fundamental principles articulated earlier. Additionally, as no one on the AONS II team worked on the AONS I prototype, the AONS I code was examined to see how much of it could be reused. AONS II aimed to transform AONS I beyond some technical limitations to provide a more robust and decoupled solution that could be utilised in a variety of environments. Little of the AONS I code base could be reused within the new software architecture.

A number of informal meetings with various project partners were held during the requirements gathering phase, between December 2006 and February 2007. These were followed by a workshop with the APSR AONS II Workgroup to check and extend requirements. The guiding principles and software architecture were confirmed as part of this process. The software architecture included a number of common services (see Figure 1) which could operate within two identified deployment modes - Local/Enterprise and Federated (see Figures 2 and 3).

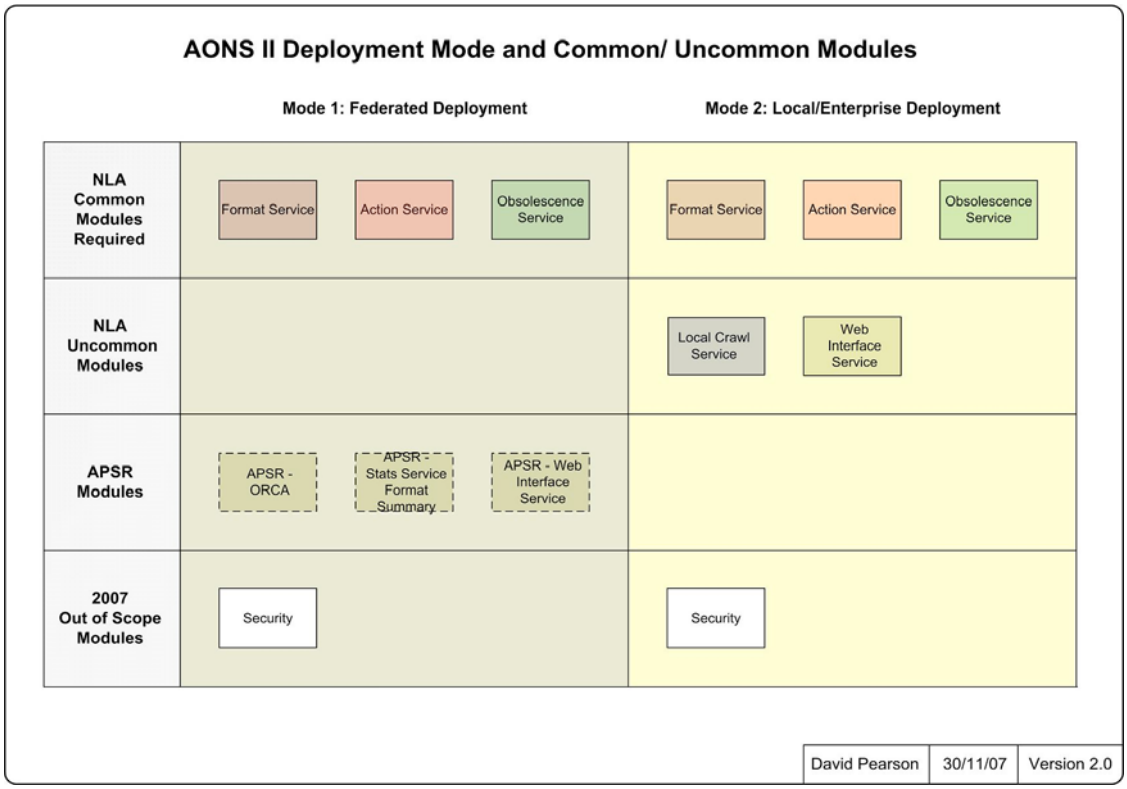


Figure 1 shows the overlap between the two deployment modes. The common modules are the Format, Action and Obsolescence services. The major difference between the two modes is that a Crawl service and Local Web interface service are built into the Local/Enterprise mode whereas the Federated mode uses the COSI Web Interface and interacts with a number of other COSI products (ORCA, BEST) to fulfil its need for repository content information.

Mode 1: Federated (APSR) Deployment deliverable characteristics

The principle deployment mode specified in the original deliverables was the Federated (APSR) deployment (Mode 1). In this deployment scenario, a number of repositories share common services. The services are supported by the AONS software which responds to a set of standard service requests. The responses are processed by the invoker of the service as required. For example, in the COSI context, services allow registration and display of repository format information as well as display and search of various format registries. In addition, other APSR-developed services could play a part in a national-level federated service infrastructure. For example, a national collections registry (ORCA) could be utilised as a source of collections to crawl or locations of collection/repository format summary files. A national research statistics service (BEST) could be a source of format information for national repository holdings.

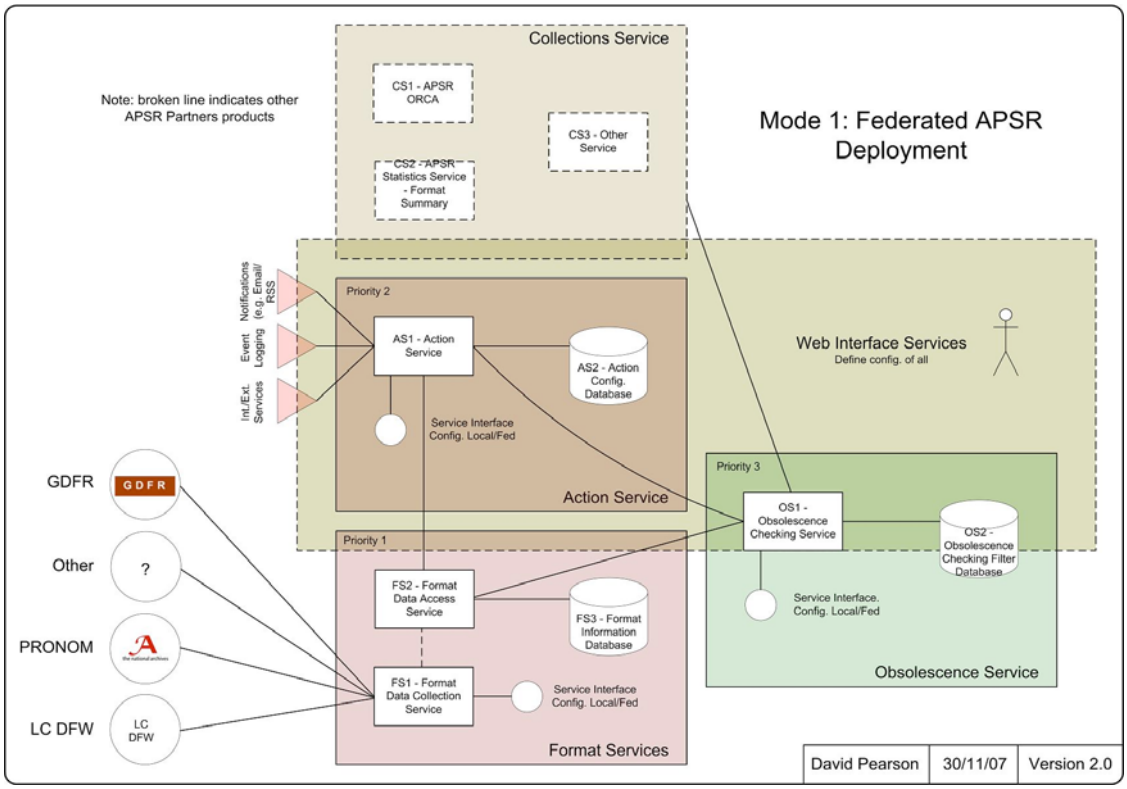


Figure 2 shows the Federated APSR Deployment Mode.

Federated (APSR) deliverable characteristics:

- Public anonymous access for read only operations;
- Protected access for system owners who can change state of the application;
- Access to public data repositories via the format summary service and ORCA (Online Research Collections Australia - APSR Collection Registry Product) which will be delivered by APSR;
- Public viewing of read only obsolescence reports;
- Public access to static, non-contextual rules for evaluating document obsolescence within public repositories;
- User interface operating on a separate server to main module deployment;
- Access to external format registries; and
- No federated repository crawl due to bandwidth and security factors.

Mode 2: Local/Enterprise Data Deployment

During the development of the requirements and use cases it was identified that a federated deployment was not the only useful purpose for the software. As a result the project sought to make the software accessible to a broader user-base. The Local/Enterprise mode was designed to run all AONS II services in a standalone environment. All AONS II functions could be accessed in this deployment mode including a local repository crawl. It was envisaged that summary repository information could be uploaded to a Federated deployment of the software if desired.

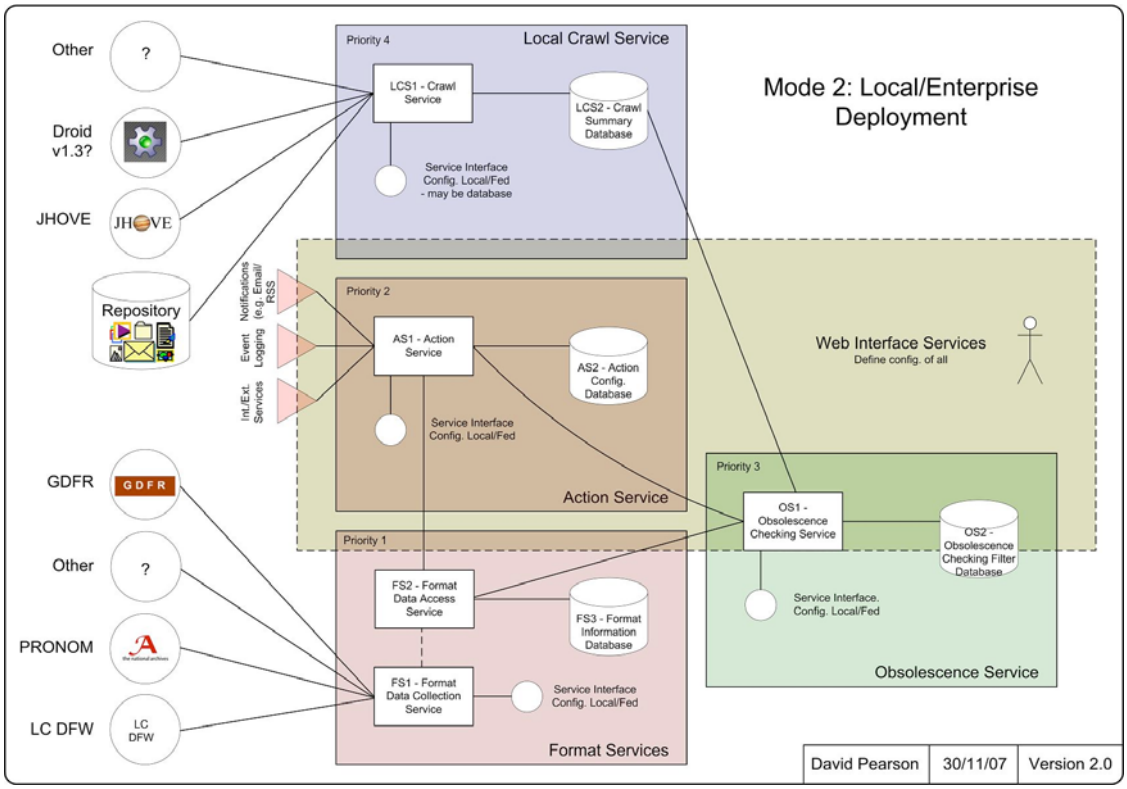


Figure 3 shows the Local/Enterprise Deployment Mode.

Local/Enterprise Data deliverable characteristics:

- No mandatory public access – repository owners can configure this;
- Access to local and probably unshared data repositories via format summaries produced by a local crawl service;
- Obsolescence rules will be heavily customised based on client environment;
- User interface bundled with main components for deployment simplicity;
- Still utilise access to external format registries, but may augment this information with own data; and
- Some organisations may wish to integrate the obsolescence tool with their own software.

What did we actually do?

The AONS II project developed a software tool which allows users to automatically monitor the status of file formats in their repositories, make risk assessments based on a core set of obsolescence risk questions, and receive notifications when file format risks change or other related events occur. This was in spite of the limitation that automatic risk metrics were not available from the international registries - Library of Congress Sustainability of Digital Formats (LCSDF) and PRONOM - as expected at the beginning of the project.

The following description of the functionality of AONS II is drawn from a paper by David Pearson called 'AONS II: continuing the trend towards preservation software 'Nirvana' '. The paper was presented at the International Conference of Preservation of Digital Objects (iPRES2007) in October 2007.

AONS II can be deployed as a part of a workflow or as a stand-alone application to:

- Check files as they are ingested; or

- Check files some time after they have been ingested, either on a one-off basis or on a regular monitoring schedule.

Like its predecessor software, AONS II is intended to work by identifying the file formats found in a digital repository and seeking information on obsolescence risk indicators by referencing file format information in external registries. Where relevant indicators are detected, the tool generates a notification to a designated person. Unlike its predecessor software, AONS II recognises the need to refer to internal information as well, and engages the repository manager more actively in determining an apparent level of risk based on both external and internal indicators.

Once a risk profile has been established for a particular repository format profile, the software can be configured to look regularly for changes in the targeted indicators, generating an automatic notification that either a new risk assessment should be carried out, or that preservation action may be needed.

Recognising File Formats and Building Collection Profiles

AONS II builds a profile of the formats in a repository or a subset such as a collection or even a single file. The profile is constructed as an XML metadata summary, which can be created from any existing compliant metadata summary, or from a repository crawl using purpose-built AONS adaptors for a given repository type (DSpace, Fedora, etc) [see Appendix C]. Crawl results may be obtained from existing repository metadata or automated format recognition tools (such as DROID, JHOVE), or both.

This approach differs from other format profiling systems which rely on downloading content files in order to identify them and build a format profile, or which use generic harvesting tools.

Format Identifiers

A comparison tool like AONS II depends on being able to distinguish accurately between different formats, and between different versions of formats, in order to identify relevant risk levels. Format identification is not necessarily an unambiguous exercise. Files may be labeled with misleading extensions; different sources may refer to the same format under different names. So that it can bring together relevant information from disparate sources, AONS II creates a internal format identifier for each apparent format found, and then tries to map it to the likely matching format identifiers used by external registries.

Based on the repository formats found, AONS II may classify formats as ‘identified’, and matched with format information held in external registries, or as ‘unidentified’. As part of this classification process, a repository manager could:

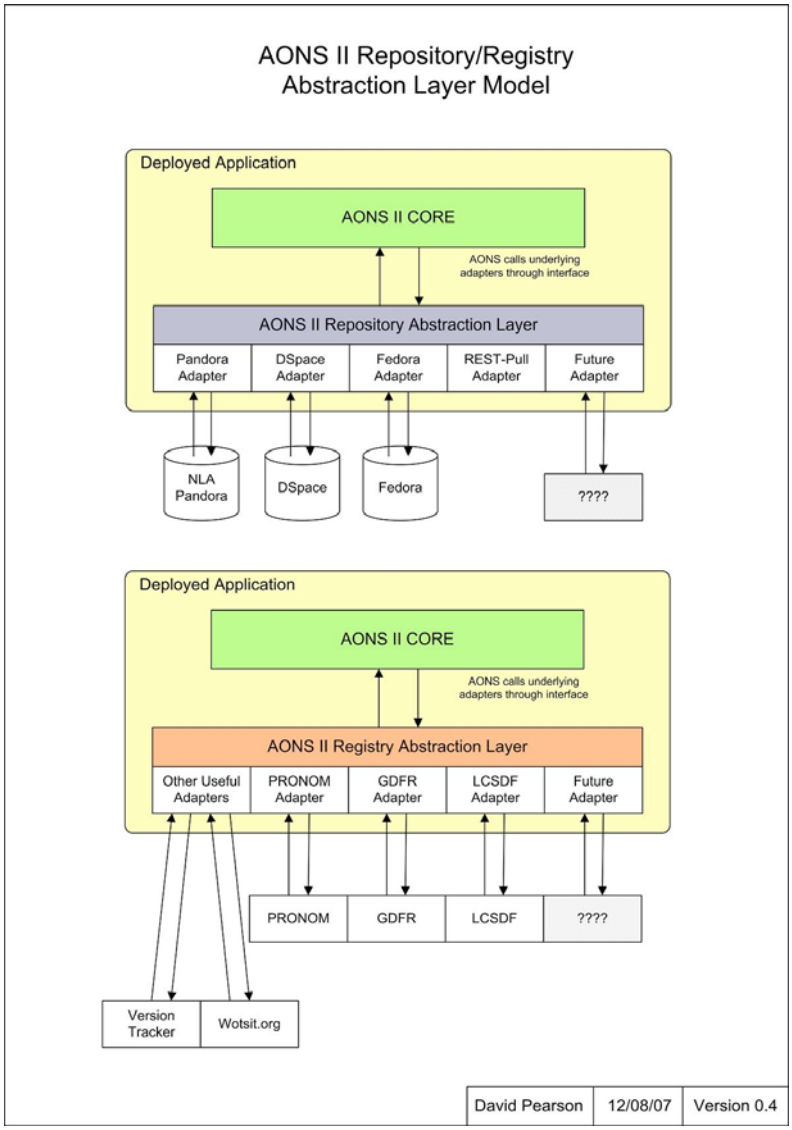
- Decide to link an unidentified format to an existing AONS internal format;
- Create a new internal format with links to external format information;
- Create a new internal format with no links (not a particularly desirable option, but a valid use case because a format might not yet be recorded in external registries, given the ever expanding superset of file formats); or
- Simply leave the format as unidentified.

Once the formats have been established in the repository or collection profile, the AONS II core software compares the list of formats and versions with information derived from external registries on formats mapped as equivalents. For efficiency purposes, AONS II stores format information from the target registries in local databases. Unlike the AONS I tool, the current software keeps the locally stored registry information from each target registry separate, so that it can be updated, synchronised, replaced or complemented by information from new sources without disrupting the entire database. Users can also add other useful links and access them through the GUI, without using a local cached copy.

A feature of AONS II is its adaptability. Users can configure it to target authoritative sources of format information as they emerge or are found to be useful. Currently the external target registries include LCSDF and PRONOM. As these registries change over time and as new registries are created and become stable, such as Global Digital Format Registry (GDFR), new adapters can be created with minimal effort. This ability to configure the targeting of registries is considered critical; during the development of this tool it became apparent that there was still no single definitive source of information on file formats.

Adapters

AONS II uses repository/registry adapters which are abstracted from the core software for interfacing to different repository and registry types. This keeps the core code isolated from the adapters so that the basic business logic does not need to be modified when creating or modifying adapters [Figure 4]. Having a decoupled approach which uses a new adapter for any new implementation has proven to be very successful in the open community. Potentially anyone with a new repository type can write an appropriate adapter and share it with the user community on SourceForge. Currently the repository adapters which have been written include generic file system, REST-ful pull, DSpace version 1.4, Fedora version 2.2, and NLA Pandora. Similarly, registry adapters include LCSDF and PRONOM.



[Figure 4] Diagram showing the Repository/Registry Abstraction Layer Model. This diagram illustrates that the AONS II core code is separated from the various adapters so that the basic business logic does not need to be modified when creating or modifying adapters.

Notification

The notification part of AONS II is configurable and based on change in state within the system. Examples of these changes in state are: end of a repository crawl; change in the information about a format in an external registry; or the expiry of a time-sensitive trigger, such as a format risk re-assessment period ending. Notification can occur in a number of forms: via email; RSS feed; and task boxes via the GUI.

Checking for Obsolescence Risk Information

Critically, AONS II software aimed to help in assessing levels of obsolescence risk, with a view to informing decisions about the need for preservation action. While we remain committed to this aim, it has been necessary to modify its interpretation in light of experience in developing the tool beyond its first prototype stages.

An initial business driver for the project was a perceived need for a tool which could automate much of the assessment process, using standardised metrics that would support machine-formulation of

recommendations on risk levels. This approach presupposed access to relevant authoritative and machine-usable information about a wide range of file formats, including information that might offer warnings about format obsolescence risks. Behind this was an assumption about the state of development of format registries, that they might offer warnings about format obsolescence risks. Development of the project had involved close study of the information that known target registries offer, and their likely ability to support automated format risk judgments.

It became apparent that in the short-term – certainly within the funding life of the AONS II project – the intended international target registries would not provide any format obsolescence risk metrics. One of them, PRONOM, has been declared by its owner institution, The National Archives (UK), to have a relevant long-term intention:

> "TNA intends to develop a holistic risk assessment methodology for electronic records that will enable us to identify risk factors at an early stage, predict their impact, and plan appropriate mitigation strategies"
> [http://www.nationalarchives.gov.uk/aboutapps/pronom/#future].

This functionality was not available during the 2007 development cycle.

Similarly, the current registries had not evolved to the stage where they are a good fit-for-purpose for a tool like AONS II. The data is not sufficiently structured to be useful in a system-automated context without considerable human intervention. Human intelligence was required to understand the content, and often little or no information is available.

Given that the target registries were not designed with tools like AONS II in mind, it is not surprising that there are some frustrations in automatically deriving risk metrics or even consistent, machine-usable information from them. However, it would be pleasing to see file format registries interested in automated obsolescence notification as a critical use case.

Therefore, the AONS II project involved deriving a series of questions which it is believed provides an effective basis for judging the level of obsolescence risk for a file format at a particular time. At this time, the rule set has not been automated. As a consequence of having to cater for potentially thousands of possible file formats, the questions have to be generic and somewhat simplistic. However, the questions aim to allow a repository owner to build a risk profile of an individual file format. At this stage they are a series of questions with corresponding free-text entry fields. Information from PRONOM, LCSDF as well as any other user-defined web sites can be made available for the operator to help answer these questions. At the completion of the assessment, based on the answers to the series of questions, the operator assigns a subjective risk level to each format. The results of all the format risk assessments are presented in the main format summary screen of the application. For practical reasons, there was a decision to wait on community feedback about the usefulness/appropriateness of the questions before hard coding workflows metrics into the software.

The full paper is also available for reference on the APSR website at http://www.apsr.edu.au/aons2/pearson_ipres_2007_text.pdf. For more detail refer to the as-built software specification and the user guide.

Testing

The testing phase involved both deploying a pilot AONS service and releasing the AONS software on SourceForge. Both the pilot service and beta release of the software were announced on mailing lists and in the APSR newsletter. APSR partner organisations were asked to expose their repository format metadata in an AONS-compliant format and were registered with the federated APSR AONS service. Other users who downloaded and installed AONS locally were asked to provide feedback to the SourceForge site.

From the outset, it was recognised that a decoupled local deployment mode would be problematic due the many different registry/repository products. To get around this problem, AONS II used registry/repository adapters, abstracted from the core software, for interfacing to different repository and registry types. This adapter abstraction layer has allowed the creation of generic adapters for a number of given repository types. However, in some cases repository products can be used in different ways which impacts the applicability of the generic adapters provided. For example, Fez requires an alternate adapter due to the specific way in which it uses the underlying Fedora repository. This example highlights the fact that the generic adapters provided may only work for where repository technologies are deployed and operated in the standard manner. Where the repository technology is customised for the local environment a specific adapter might need to be engineered. Testing also highlighted that the graphical user interface in the local mode still required some attention.

In the federated deployment mode, testing verified that the project delivered a number of services including repository registration, and viewing of format and risk summary information. It also demonstrated the benefits of a managed service implementation to reduce the workload and responsibilities of the institutions using it as opposed to running and maintaining their own instance of another software package.

What changes did we make to the original deliverables?

Additional NLA deliverables were added for the following reasons (see Appendix A for original deliverables and Appendix B for mid-year report):

AONS registry and repository adapters (mid-year review deliverables 6 and 9)

A number of new repository adapters were identified and specified - generic REST-pull, generic file system and TRIM. These were added to make the software useful to a broader audience. REST-pull and generic file system adapters were successfully implemented. The TRIM adapter was not built because information gathering sessions could not be arranged with the vendor within the timeframes required.

Similarly, delays in the GDFR project schedule meant that a GDFR registry adapter could not be developed during the AONS II project timeframe.

Configurable risk recognition and risk prioritisation modules and rules (mid-year review deliverable 4)

The risk rules to define file format obsolescence and how it can be determined involved a substantial effort which was not earlier anticipated in the original deliverables. Early in the project, it became apparent that risk metrics were not available from international registries such as PRONOM and LCSDF. While risk assessment information was available in these registries, to some extent it was not in a machine-actionable form. Therefore, establishing a method for quantitatively measuring the risk of obsolescence became as important as the development of the AONS II software. One of the outcomes was the first version (1.0) of a set of

questions for determining risk metrics against file formats which was also incorporated into the software with a logical workflow. Because of the effort invested in the questions there were less resources available for integrating them into the software. The result was a more manual workflow than originally intended within the software.

Local Graphic User Interface (mid-year review deliverable 13)

AONS I had no Graphical User Interface (GUI). It was envisaged that APSR would develop a user interface for the federated deployment of AONS II. However, it was also recognised that a GUI would be necessary for the local deployment mode. As a result some project resources at the NLA were assigned to meet this deliverable. There was no web interface design resource available to the project; therefore the GUI in this mode requires further refinement.

What lessons did we learn?

A number of lessons were learnt in the AONS II project:

- As the NLA did not have spare business analysis and software development resources available, these functions had to be found elsewhere. The business analysis function was fulfilled by the project manager. This has delivered a cost benefit but, due to skills required and other commitments, has meant that many of the specifications have been developed as required rather than at the beginning of the project. Also, the work could not be done to the same degree and quality as a qualified BA would achieve.
- Due to the dynamic of the team and the skills members possessed a successful outcome was obtained.
- Obtaining software engineering resources in February was a very frustrating undertaking (given that we started the process in December). Apparently the job cycle starts in October-December for filling in January, February and March. Perhaps if the process was started in October we might have been able to attract a larger field of resource options.
- It would have been better to have two software developers working together for the majority of the coding phase of the project as suggested by the agile project methodology.
- Large cumbersome project methodologies are not required for this kind of agile software development.
- Part-time software development work at the end of a project is not conducive to the best outcome.
- It would have proved advantageous to have more structured time for testing and project assurance. Due to the limited testing the software is on SourceForge as a beta release.
- The funds provided at the beginning of the project work were optimistic. While additional funding was provided by both the APSR and NLA partway through the project, it still did not offset the size of the task and the new deliverables that were introduced. Evidence of this is in the quality of the end-product – a beta release of the software and less than comprehensive documentation. The outcome meets the original desire of the APSR project to have a demonstrable product. However, it does not fulfil the NLA's need for a robust software module that is ready for production deployment.
- At the beginning of the project the APSR software products Online Research Collections Australia (ORCA) Collection Service Registry and the Benchmark Statistic Service (BEST) were supposed to provide AONS II with file format information. Subsequent developments of ORCA changed this requirement

and at the end of the APSR 2007 development cycle the BEST software product had not been built. These changing requirements had an impact on the AONS II product.

- External format registries: It has been found that their purpose is not yet closely aligned with AONS II business needs (i.e. to help a repository owner understand their collection) rather than just supplying some objective and mostly subjective information about a format. Therefore, additional web tools and configurable risk tools (to obtain metrics) have become a more important component of the software. Due to the registry limitations there will be a need to develop this tool further.
- The project's success can partly be attributed to the benefit of having a dedicated project manager. While the project manager took on that role for two projects, it was nonetheless his primary responsibility. This is often not the case in other similar projects.
- One of the good things about the project was having time allocated communicate and promote the project and its outcomes. This is key for the awareness and benefit of the community surrounding such a project. It is also a foundation for future collaboration relationships with other like-minded institutions.
- The successful outcome of the project showed that a small project team, working cohesively, can achieve very good results. It could be argued on this basis that earlier references to insufficient resources are irrelevant. However, in this project a trade-off was made in favour of time and cost to the detriment of quality.
- In addition, the GUI has proven to be one of the most important parts of the application as it has driven the other functions. This highlights the necessity to incorporate the development of the user interface and workflows early in the project to ensure the underlying software can support the requirements in this area.

What needs to happen next?

The current work is delivering a Java-based, repository/registry agnostic tool which can be deployed in any Service Orientated Architecture. Therefore these foundations could be amenable for future development improvements.

Developing the AONS tool into a central web service based on feedback from multiple AONS expert repositories could:

- Provide machine- and human-harvestable Format Risk profiles for file formats (see Appendix D).; and
- Provide machine- and human-harvestable Format Identification metadata profiles for file formats (see Appendices E and F).

Both of these undertakings would involved the development of new web services. If this were to occur, future development work on the AONS software might also include:

- Development of automated risk workflow; and
- Development of export functionality to a central web service
- Further development of REST Web Services (only a limited set of functionality is available via Web Services)
- Deployment as a production service with full governance
- Collaboration with major Format Registry owners to improve registry services and relevance to AONS requirements

An example of the above web-based service is provided below in Figure 5:

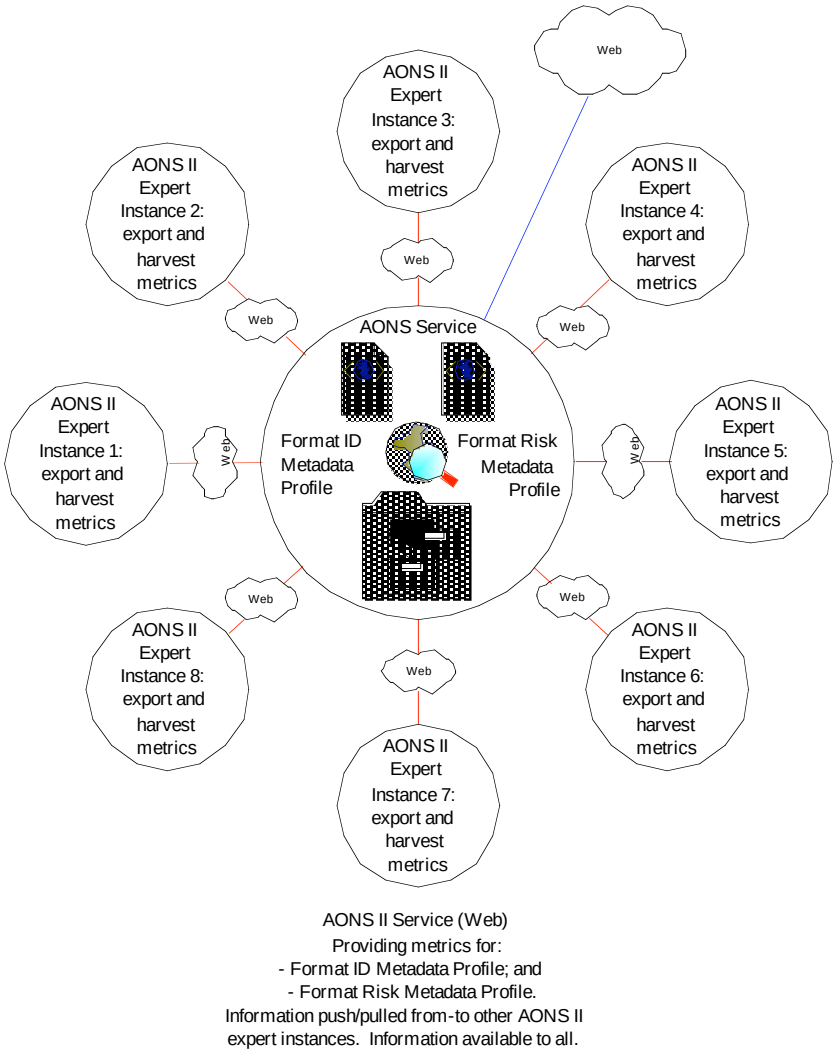


Figure 5 provides a high level example of how this web-based service could work.

Conclusion

In summary, the project met it's objective of refining the AONS I software into a platform-independent downloadable tool for obsolescence notification. AONS II provides authoritative information from a number of international registries to support decisions on preservation action required to retain access to digital objects stored in a range of repository types.

The success of the project is testimony to both the skills of the project team and stakeholders, and the profound value of the PANIC model. The development of the AONS II software shows, in particular, that it is possible to implement the automatic obsolescence notification element of the model. However, some additional work is required in the areas of risk assessment and machine-actionable risk metrics. Thus, the project has also focused on the development of draft risk questions towards this goal.

The AONS II user interface has shown what can be achieved by making format registry information more accessible and meaningful to the repository manager through associating it with the contents of the repositories being managed. This latter point is critical in that it is only through an awareness of the repository contents that meaningful action can be taken.

The architecture of the AONS II software has made an extraordinary impact on the way in which the APSR partner projects have been approached, as well as the future possibilities for expanding the functionality of the software. It has also impacted thinking in a number of other projects currently being undertaken at the NLA. Decoupling the key components has provided the means to build the federated implementation through composition of the building blocks required in a Service-Oriented Architecture environment.

Since the AONS II software is currently in a beta state, it is recognised that some further investment is required to transition it into a production-ready application. As the APSR project is drawing to a close, no further effort can be supported through the original project sponsors. It is anticipated that parties with a commitment to digital preservation, including partners in the AONS II project, will take the necessary action to evolve AONS II into a fully-tested production-level application and beyond.

The interest generated by the AONS II work has shown that it is an area of critical importance in the digital preservation community. A number of international institutions have shown interest in the AONS II software and the concepts behind it. Therefore, as an open-source product, there are many opportunities for further collaborative development of the software in the future. We hope that this is realised through the future initiatives of the digital preservation community.

Acknowledgements

The authors wish to thank APSR and DEST for supporting this project. At the NLA the authors would like to recognise the assistance given by Colin Webb, Gerard Clifton, Douglas Elford and Nicholas del Pozo on the definition of file format obsolescence, and David Levy for his work on AONS II. We are also indebted to Dr Jane Hunter for the PANIC model and for providing project assurance. We would also like to thank our colleagues at APSR, ARROW, ANU Division of Information's Digital Resources Service, and The Library Technology Service at the University of Queensland Library for their input on this project.

Appendix A

Original APSR Project Proposal - Obsolescence Notification Services Proposal NLA (COSI-B) (dated 14 Nov 2006)

Title	Format Notification and Obsolescence Service [AONS Generic Extension project – "AGE"]		
APSR Partner	National Library of Australia	**Component ID**	COSI B
Collaborations & Leverage	Australian Partnership for Advanced Computing Australian National University University of Queensland Leveraging off related work: • Global Digital Format Registry (Harvard University Library) • PRONOM (UK National Archives) • CASPAR (EU) • Library of Congress Digital Formats Web • PANIC (Dr Jane Hunter) • AONS (Automatic Obsolescence Notification System) (Joseph Curtis/APSR)		
Objective	To refine the Automatic Obsolescence Notification System (AONS) developed in an earlier stage of APSR, to a platform-independent downloadable tool that automatically provides information from authoritative international registries to support decisions on preservation action required to retain access to information resources stored in repositories.		
Description (Numbered task list)	1. Evaluate existing AONS prototype and develop detailed functional specification for work required, and clean up existing code 2. Develop configurable mechanism for specifying risk and priority rules 3. Uncouple existing AONS prototype from DSpace repository model and re-code for repository independence, and re-code for target registry independence 4. Work with GDFR, PRONOM, CASPAR and other appropriate format and access risk services to refine services and interfaces that the AONS tool can exploit 5. Evaluate, refine and develop existing collection characterization/metadata extraction; risk notification and feedback elements 6. Develop a target-independent services port to provide an interface to preservation processing services when they are available 7. Develop integration interfaces for DSpace, Fez-Fedora, ARROW/VTLS and NLA Pandas repositories 8. Test on range of repository types 9. Develop as downloadable product with full documentation 10. Investigate applying preservation process services directly through the AONS tool. 11. Investigate applying quality checking processes for preservation services.		
Deliverables	1. Functional specification		

(Keyed to task list)	2. Configurable risk recognition and risk prioritization modules and rules 3. Re-coded repository- and target registry-independent AONS tool 4. AONS user specification for partner format registries and interfaces 5. Collection characterization, notification and repository feedback modules 6. Target-independent preservation services interface 7. AONS integration interfaces for DSpace, Fez-Fedora, ARROW, and Pandas 8. Report on AONS tests on a range of repository types 9. Full documentation; AONS tool available in public domain 10. Ongoing issues paper	
Timeline (Keyed to task list)		**Month**
	To be completed after 16 Nov 1. 2. 3.	
Personnel and Responsibilities (Keyed to task list)	1. NLA (DP) + business analyst (APSR funded) + APAC, ANU, UQ 2. NLA (DP) + business analyst + programmer (APSR funded) 3. NLA (DP) + programmer (APSR funded) 4. NLA (DP) + business analyst (APSR funded) + NLA staff 5. NLA (DP) + business analyst + programmer (APSR funded) 6. NLA (DP) + business analyst + programmer (APSR funded) 7. NLA (DP) + APAC + ANU + UQ 8. NLA (DP) + NLA staff 9. NLA (DP) + technical writer (APSR funded) + NLA staff	
Funding	$75,000 (+ NLA APSR funded Project Manager) + in-kind contributions by NLA, APAC, ANU and UQ $10,000 Architectural Consultant (Jane Hunter) Travel or teleconferencing component also may be required	

Appendix B

Updated APSR Project Proposal and mid-year report - Obsolescence Notification Services Proposal NLA (COSI-B) (dated 22 June 2007)

COSI-B	**Format Notification and Obsolescence Service (AONS II)**		
APSR Partner(s)	National Library of Australia		
Project Specification	http://sts-bscw.anu.edu.au/bscw/bscw.cgi/d17250/AONS2_Specification.doc		
Public information	APSR Website: http://www.apsr.edu.au/aons2/index.htm APSR Wiki: http://pilot.apsr.edu.au/wiki/index.php/AONS		
Project Leader(s)	David Pearson (NLA)		
Technical Leader	Matthew Walker (NLA) and David Levy (Frontier-Group)		
Objective	To refine the Automatic Obsolescence Notification System (AONS) developed in an earlier stage of APSR, to a platform-independent downloadable tool that automatically provides information from authoritative international registries to support decisions on preservation action required to retain access to information resources stored in repositories.		
Commencement	Jan. 2007 (note: D = the original deliverable No.)		
Deliverables and deliverable date	1. Project Specification	14 February	Done
	2. APSR AONS Workgroup Meeting	21 February	Done
	3. Functional specification (D1)	15 April	Done
	4. Configurable risk recognition and risk prioritization modules and rules (D2)	23 July	Done
	5. Re-coded repository- and target registry-independent AONS tool (D3)	15 June	Done
	6. AONS registry adapters for use with LoC DFW, PRONOM (GDFR is dependent on GDFR progress) (D4)	23 July	Done
	7. Collection characterization, notification and repository feedback modules (D5)	29 June	Done
	8. Target-independent preservation services interface (to COSI framework) – REST SOA – (COSI framework is based on APSR collaboration) (D6)	21 Aug	Done
	9. AONS repository adapters for use with REST/push-pull, generic file system, DSpace, Fedora, NLA's Trim and Pandora repositories (D7)	23 July	Done
	10. Report on AONS tests on a range of repository types: Test Report - 1 page from DSpace, Fedora, (partners report), Trim and Pandora (NLA report) (D8)	21 Aug	Done

	11. Full documentation (D9): As built specification & how to documentation AONS tool available in public domain (D9): Software has been put on SourceForge	21 Aug	21 Aug Done
	12. Ongoing issues Paper (=APSR Wiki and Developers Blog) (D10) Currently writing a paper on AONS and format obsolescence for iPRES2007 (Oct 2007) and DCC conference (Dec 2007)	Ongoing	End of 2007 year
	13. Local GUI	23 July	Done

	Mid-Year Report
Start Date	Start date of the Project Nov 2006 – Development March 2007.
Completion Date	Under current funding the development phase of the project is due to be finished half-way through August 2007. Due to testing constraints NLA has requested allocation of additional funding for 80 hours work. If this is approved the development completion date for the NLA will be mid September 2007. The end of the promotion of the project will be Dec 2007.
Adjusted milestones and rationale for changes	See Project Progress Summary (additional deliverables) for detail. 5. & 10.- wording change to reflect current state. 9.- new adapters added. 12.- added - some extra NLA deliverables. 13.- added – extra NLA deliverable.
Evidence of Deliverables	1. Project spec on BSCW and Wiki. 2. Functional spec on BSCW and Wiki. 3. All other documentation on Wiki including: • workshop outcomes; • deployment modes and adapter diagrams; • AONS II Deliverables; • AONS II Functionality Breakdown; • AONS II Motivations. 4. Risk Question to be posted when complete. 5. Release schedule documents at NLA.
Public information	APSR Website: http://www.apsr.edu.au/currentprojects/index.htm APSR Wiki: http://pilot.apsr.edu.au/wiki/index.php/AONS AONS II developer Blog: http://aons2dev.blogspot.com/
Project Progress Summary	This project has currently gone to plan with additional funding supplied by NLA. The current deliverables still seem to be attainable (although the risk workflow and comprehensive testing are potential problems). Additional deliverables Additional NLA deliverables have been added for the following reasons: • AONS I had no Graphic User Interface. It was envisaged that in AONS II APSR would do this development work. However,

due to the requirements that the software works in both a federated (APSR) mode and in a local mode, a GUI has proved to be critical. In addition, the GUI has proven to be one of the most important parts of the application as it has driven the other functions.

- The risk rules set defining format obsolescence and how it can be determined is a world-first development. As risk metrics are not available from registries such as PRONOM and LoC, defining how the risk can be determined has become almost as important as the AONS II software. We have currently specified what the most likely questions are, and developed a possible logical workflow. We believe that the questions and a manual workflow can be integrated into the software before end of project. However implementing an automatic risk workflow with the current resources will probably prove to be difficult. The original deliverable (D2) stated we would build a "Configurable risk recognition and risk prioritization modules and rules". This has proven to be more problematic than originally estimated (initially the registries were considered more usable and potentially automatic than they have proven to be).
- There are a number of new repository adapters which have been specified: These include generic REST push/pull, generic file system and TRIM. These were added to make the software more useful to a broader audience.

<u>Limitations</u>

A number of limitations of the current AONS II application need to be articulated:

- External registries: It has been found that their purpose is not yet closely aligned with AONS II business needs (ie. to help a repository owner understand their collection) rather than just supplying some objective and mostly subjective information about a format. Therefore, additional web tools and configurable risk tools (to obtain metrics) have become a more important component of the software. Due to the registry limitations there will be a need to develop this tool further.
- Scope and timing. As the NLA did not have the BA and development resourse available, these functions have had to be found elsewhere. The BA function has been fulfilled by the project manager (this has delivered a cost benefit but, due to skills required and other commitments has meant that the specification have been developed as required rather than at the beginning of the project). The money provided has also only allowed for 5 months of a contract software developer (additional funds have been provided by the NLA).
- It would be advantageous to hire a technical writer to work on the as-built specifications and help text. This would provide a consistent APSR look and feel (in line with the other projects).
- Testing the code. At this stage unless additional funding is available there will be limited testing conducted.

<u>Future work</u>

The current work is delivering a Java based, repository/registry agnostic tool which can deployed in any Service Orientated Architecture. Therefore these foundations could be amenable for future development improvements, including:

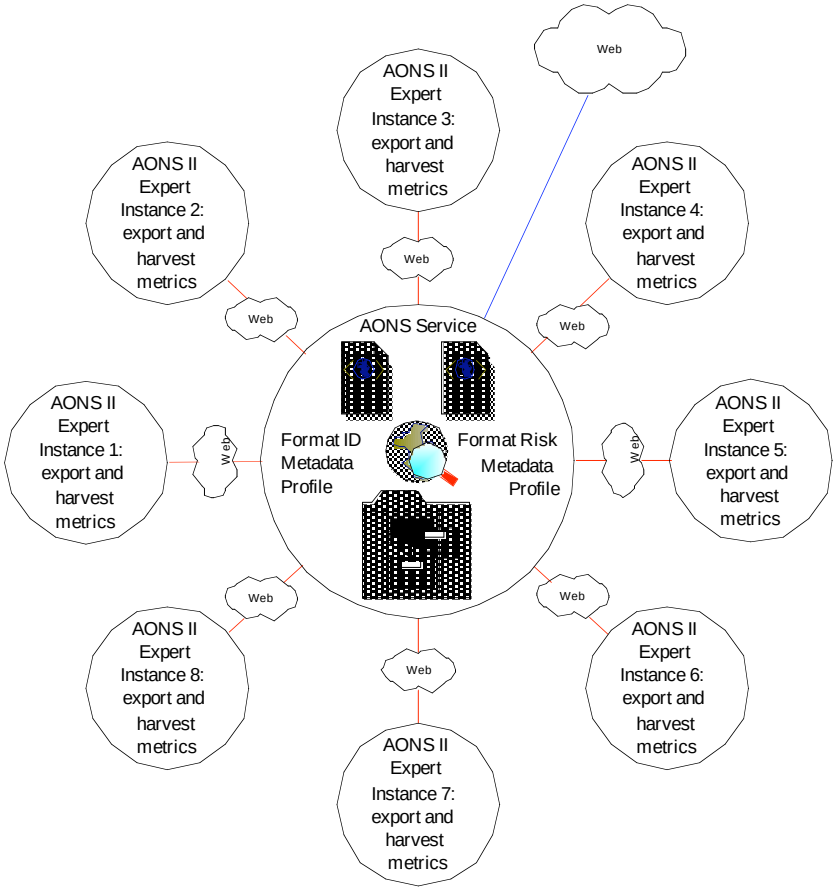
Developing the AONS tool into a central web service based on feed back from multiple AONS expert repositories. This could:

- Providing harvestable Format ID metadata profiles (new web service); and
- Providing harvestable Format Risk profiles (new web service).

If this was to occur, future development work on the AONS software might also include:

- Development of automated risk workflow; and
- Development of export functionality to a central web service.

An example of the above Web based service is provided below:



AONS II Service (Web)
Providing metrics for:
- Format ID Metadata Profile; and
- Format Risk Metadata Profile.
Information push/pulled from-to other AONS II expert instances. Information available to all.

Appendix C

AONS XML metadata summary


```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
 Copyright 2004-2007 the original author or authors.

 Licensed under the Apache License, Version 2.0 (the "License");
 you may not use this file except in compliance with the License.
 You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

 Unless required by applicable law or agreed to in writing,
software
 distributed under the License is distributed on an "AS IS" BASIS,
 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 See the License for the specific language governing permissions
and
 limitations under the License.
--> 
- <collection crawled="2007-08-02T15:16:55" name="Demetrius">
  <format name="Tagged Image File Format" version="6" puid="fmt/10"
    count="10105" />
  <format name="Extensible Markup Language" version="1.0" puid="fmt/101"
    count="2624" />
  <format name="Portable Document Format" version="1.3" puid="fmt/17"
    count="1354" />
  <format name="Exchangeable Image File Format (Compressed)" version="2.2"
    puid="x-fmt/391" count="17311" />
  <format name="JPEG File Interchange Format" version="1.01" puid="fmt/43"
    count="15916" />
  <format name="Portable Document Format" version="1.4" puid="fmt/18"
    count="936" />
  <format name="Windows New Executable" version="" puid="x-fmt/410"
    count="2" />
  <format name="Portable Document Format" version="1.5" puid="fmt/19"
    count="183" />
  <format name="Waveform Audio" version="" puid="fmt/6" count="526" />
  <format name="Portable Document Format" version="1.2" puid="fmt/16"
    count="684" />
  <format name="Exchangeable Image File Format (Compressed)" version="2.1"
    puid="x-fmt/390" count="6062" />
  <format name="JPEG File Interchange Format" version="1.00" puid="fmt/42"
    count="86" />
  <format name="JPEG File Interchange Format" version="1.02" puid="fmt/44"
    count="1057" />
  <format name="Portable Document Format" version="1.1" puid="fmt/15"
    count="56" />
  <format name="Microsoft Word for Windows Document" version="97-2003"
```

```
    puid="fmt/40" count="2" />
  <format name="Graphics Interchange Format" version="1989a" puid="fmt/4"
    count="138" />
  <format name="Hypertext Markup Language" version="" puid="fmt/96"
    count="136" />
  <format name="Hypertext Markup Language" version="4.0" puid="fmt/99"
    count="27" />
  <format name="Hypertext Markup Language" version="4.01" puid="fmt/100"
    count="108" />
  <format name="Graphics Interchange Format" version="1987a" puid="fmt/3"
    count="26" />
  <format name="Exchangeable Image File Format (Uncompressed)"
    version="2.2" puid="x-fmt/387" count="80" />
  <format name="Extensible Hypertext Markup Language" version="1.0"
    puid="fmt/102" count="39" />
  <format name="Portable Document Format" version="1.6" puid="fmt/20"
    count="230" />
  <format name="Hypertext Markup Language" version="3.2" puid="fmt/98"
    count="9" />
  <format name="Portable Network Graphics" version="1.0" puid="fmt/11"
    count="67" />
  <format name="Portable Network Graphics" version="1.1" puid="fmt/12"
    count="6" />
  <format name="MS-DOS Executable" version="" puid="x-fmt/409" count="3" />
  <format name="PostScript" version="2.0" puid="x-fmt/406" count="3" />
  <format name="Raw JPEG Stream" version="" puid="fmt/41" count="5" />
  <format name="Exchangeable Image File Format (Uncompressed)"
    version="2.1" puid="x-fmt/388" count="4" />
  <format name="ZIP Format" version="" puid="x-fmt/263" count="2" />
  <format name="Unknown" version="Unknown" count="644" />
- <collection crawled="2007-08-02T15:16:55" name="Demetrius">
  <format name="Exchangeable Image File Format (Uncompressed)"
    version="2.1" puid="x-fmt/388" count="4" />
  <format name="ZIP Format" version="" puid="x-fmt/263" count="2" />
  <format name="Unknown" version="Unknown" count="300" />
    </collection>
    </collection>
```

Appendix D

AONS Future Extension - Centralised Gathering of Format Information and Risk Assessments

David Levy

The Problem

- Registries of format data do not change as fast as the pace of the digital world
- Registries often include only data on common formats
- Registry data is often sparse, many listed formats are simply placeholders for data when there is time to insert it
- Community involvement in the upkeep of registries is minimal: any external updates are done in an adhoc basis, there are few clear means for user driven interaction
- We should be seeking to create communities of interest groups with tools to let people contribute their skill in an area

The Solution

- Create a service which accepts community generated content
    - Format Information
    - Risk Assessment Information
    - "What this information helpful" style votes
- Re-expose this information as a source of input into AONS application
- Depending on the involvement of a repository, AONS installations may fulfil one or both of the following roles:
    - Data Population: Feeding format information back into the central service
    - Data Receiver: Utilising community created format information and risk metrics to help them understand their own Repository

Similar Solutions

- Wikipedia: That is the absolute in terms of 'free form' user contribution. We would probably have a bit more focus in terms of what was contributed (keeping ourselves to Format information and risk assessments), but it is a similar model. Also, we would face similar issues as Wikipedia if there were conflicting views on a format's risk. Coping strategies could be put in place from the outset whereby:
    - A format can have multiple risk assessments
    - Multiple risk metrics driven by multiple risk assessments have some sort of average
    - In addition to performing risk assessments, people would also be able to 'rate' a risk assessment on how helpful it has been to them, which would in turn affect that risk assessments contribution to an average.
    - Risk assessments could also have a diminishing value over time: this would force the community to 're-assess' a format so it would not return to it's unassessed 'high risk' value.
- Wotsit.org: This website is driven by user generated content. Information on different files is created pretty much 'free form', not only in the sense of unstructured text but also even in the format of the files uploaded: it is often a mix of HTML, links and plain text. Hopefully we could have the same kind of user participation yet keep some structure in the user added content: at the very least we'd try and have a few common fields (name, version and mime type) as well as free text and a quantifiable risk metric for a given format.
- IMDB.com: This is a nice site driven both by user generated content but also with a bit of an overarching administration role fulfilled by the site owners. The functionality to look at in this site which we'd try and bring across is the 'rating' mechanism on user generated reviews: anyone is free to review a movie, but only the reviews with the top ratings will show up. This would be a relatively simple mechanism to emulate for risk assessments and even potentially information about a format itself.

Appendix E

AONS Future Extension - Plugin/Service/Community Driven Identification of Digital Formats

David Levy

The Problem:

- Droid, JHove and to an extent *nix File utility, currently the best in the field, are relatively static identification tools
- Often the people who know most about what makes a format a format are left out of the loop.
- Often the identification tools have a minimal set of analysers for file formats, specialist file formats are often unidentified or identified badly
- Often the identification tools are relatively fragile and need quite a lot of coaching to get the 'right' analyser to analyse a file, which implies we already know what type of file it is.
- Currently analysers are relatively slow
- Current analysers are painful to setup, often requiring many extra library dependencies which have to be compiled in at deployment time. Updates to the analysers full require redeployments.
- Tools are either platform dependent or require 'analysers' which are platform dependent or the actual applications in which the digital document was created.

The Solution

- Create a service which allows users to input their domain knowledge in the analysis of digital data
- This service does not directly identify files but gives information about 'how' to identify files (like what analyser to use when a certain amount of information is gathered).
- Next generation of digital identification tools access this service to retrieve 'analysis pathways' and 'analyser plugins' which they use to identify files.
- Digital identification tools utilise 'analyser plugins' to update their functionality at runtime.
- Write the tool and analysers in Java so it is platform independent

Similar solutions

- Magic numbers in *nix. This is actually very similar to what this is proposing… but hopefully this is a little more extensible.
- Anti-Virus Definition Update Processes: Virus detection software not only sells the current set of virus definitions, but also ensures there will be mechanisms in the software to update these virus definitions and potentially the core engine for identification. We would have similar processes in the identification tool by which we would retrieve new 'analyser plugins' (similar to the virus definitions) and retrieve 'analysis pathways'.
- Eclipse IDE: By itself is a relatively minimal set of software, but has an elegant API for plugin creation and retrieval. This allows extension of the IDE relatively easily and means that vertical interest groups can customize the IDE depending on their function: Data Centric people download entity mapping plugins, C++ developers can download their plugins and people who bridge areas can download combinations. The point to remember here is we could have similar 'analysis packages' for different vertical markets. Financial focused organisations could have one set of plugins and engineering a different set.

Appendix F

Concepts and Initial Ideas for an

Online File Format Identification Service

History:

action:	agent:	date:
document created	N. del Pozo	11.10.2007
added diagrams to explain proposed identification paths	N. del Pozo	15.10.2007
clarified some of the formulas.	N. del Pozo	20.11.2007

Some Initial Issues:

At present, an efficient methodology for file format identification and verification has been held back by a number of issues. For the purpose of this paper, primarily:

- while there are many file validation tools, there is very little that exists in the way of providing a system for cooperation or interoperability between them.
- many of the available file validation tools only cover a small range of formats, relative to the total number of known (and unknown) file formats
- many of these tools were developed for a specific operational environment, and as such may display limitations when utilised elsewhere.
- some tools provide functionality that exists elsewhere, but to a greater or lesser capacity.

These issues can be seen as contributing to a general environment where, although file format identification and validation is possible, it is either a slow, or incomplete process. In some instances, even with these limitations, the currently available collection of tools can be considered operationally sufficient. For example, Stanford's Digital Repository recently conducted tests using an "Empirical Walker" software package, that utilises only JHOVE as its format identifier; so long as they restricted their ingest to files that are readily identified by JHOVE, the software requirements did not exceed their capabilities.

Stanford's approach for this test highlights one of the current 'traps' that an institution wishing to implement a file verification/validation step into their digital archiving process can easily fall into. If they use a single program to identify files, because that program sufficiently meets their needs, their capacity to handle the implementation of any arising requirements may be dictated by the capabilities of their elected software.

Alternatively, institutions may find that their current approach to dealing with the problem of file format identification is a product of the software they have available, and may not entirely address their requirements.

In both these cases, to implement a new program to cover additional cases presents an unwelcome additional expense.

An alternative to this problem is that institutes may choose to utilise modularise their file identification system. For example, a library may choose to run both JHOVE and DROID, and from the results of each program, decide on the 'most likely' format.

This approach, however, is similarly troublesome. Although it may provide more accurate results, the certainty of each program's accuracy is to a large part provided by the program itself, and as such the problem of aggregating results either becomes mathematically complex, or (as is more likely the case) irrelevant. For example, DROID provides some indication of its certainty, but this is calculated using a different set of protocols than JHOVE. This is compounded when we consider that the user may not necessarily agree with DROID's level of certainty for any given file format. As such, they may be required to implement rules for certainty based on the individual format level, rather than the program level.

Additionally, to send each file through each file verification/validation program is problematically slow, especially for systems processing large amounts of information. Worse, the entire system will become progressively slower with time, as new modules are added, or a greater volume of files is processed.

From these brief conditional descriptions, it is possible to raise some more issues that digital repositories should take into consideration:

- Any methodology used by a specific tool to calculate the 'certainty' with which it identifies any given file format is specific to that tool, and difficult to incorporate in any approach that relies on the results of more than one identification program.

- The user may not share the confidence of their identification tool in all situations, meaning that the amount of information needed to accurately aggregate the output of various file identifiers into a meaningful final figure must take place at the individual file level.

- To use more than one form of identification on each file and aggregate the results into a meaningful and *accurate* result is computationally complex, and will become more so as more tools are incorporated into a system.

- To use more than one form of identification on each file is a slow process, and will become slower as more tools are incorporated into a system.

From the issues currently identified, it is evident that when it comes to identifying particular file formats, not all tools are created equal. Although the seemingly obvious solution is to simply use the best tool for the job, we must also acknowledge that we are, above all, restricted by the seemingly tautological problem:

- When a file format is unknown, the best tool for identifying that format is also unknown.

Some Potential Solutions:

A solution to this problem should adequately deal with all the issues as presented above, and do so in a way that does not specifically trap the methodology to any one piece of software. Additionally, retrofitting an existing application to handle new file formats is

possible, but not a desirable solution, as it presumes too much about the program's capacity to handle different file formats. Additionally, this type of solution would devalue the worth of any other program, and as such may not achieve optimal results.

From a logistics perspective, it must be acknowledged that the chances of the various file identification tools being unified, or agreeing on a shared format and system for identifying files, is unlikely. Additionally, it may be the case that this kind of amalgamation of services may not be desirable for other reasons; it could potentially stifle productive innovation in a particular tool, or discourage the development of new technologies.

As such, we can make some initial assumptions about the nature of the system that would provide a solution to these problems:

- it should be capable of utilising a variety of tools and methods, in such a way that when new technologies surface, they can be integrated so that they improve the output of the fundamental system, without significantly changing its methodology.

- it should be capable of scaling upwards without changing the basic form of algorithm used to calculate certainty of file identification, nor should the processing time be significantly increased. Although it is assumed the overall algorithm may become more complex, in that it may contain more units, this will be within a set of predefined parameters - the overall shape of the algorithm should never change.

- in order to accommodate a variety of different institutions, it should not be directly implemented, but it should facilitate, or provide a guide for implementation, that is designed to accommodate the various needs and capabilities of different institutions, whatever they may be.

The Online File Format Identification Service

The particular solution proposed by this paper is an online file identifying service. Described most tersely, this would be a system for ranking, and suggesting operation order for already existing tools, and new tools as they become available.

The service would operate on the following principles:

- There are many applications for identifying files

- Not all tools are as useful as others for identifying some file types

- Not all tools need to be run in order to identify some file types

- The order in which tools are run will vary from file to file

- The order in which tools are run is dictated by the overall trust placed in a tool's capacity to identify a given format

- the overall trust that is placed in a tool comes from an open and discursive community

Basic Premise:

The basic premise for the proposed online file format identification service is that it will take the form of an online community web site, that displays the following functionality:

- Primarily, the site will act as a repository of identified file formats: as file formats are discovered (even if there is no method for identifying that format), they are added to the 'library'.

- For each file format contained in the library, where possible there will be an entry for each file identification tool that currently claims to accurately (or moderately) identify that particular format.

- It is assumed that each tool's methodology for identification either works, or doesn't work (either a tool can identify the specific file format it claims to, or it can't).

- Each tool's capacity to identity a given file format will be assigned a numeric ranking, based on how many people in the community believe that the tool is accurate in its claims.

- Each tool will be given a percentage of 'participating users' who believe the tool to work, relative to those who do not believe it to work. Participating users are those users who have voted

- based on the numerical ranking, and level of trust given to the identification tools, the service will provide, upon request, a list that outlines the order in which identification tools should be run, and at which point in that order the user can be satisfied that a particular file format has been correctly identified.

- Institutions can use this information in any way they wish, though it is expected that it could provide a basis for developing the file format identification portions of their digitisation workflows.

How "Trust" is Calculated

The driving principle behind the order in which the web service suggests users deploy their file identification resources is based on the concept that for each tool, the overall community will have a certain level of trust that they place in that tool to return plausible results.

To quantify this level of trust, the following system is proposed. Initially, allow four conditions:

1) Any person, or organisation of people, can register as a user.

2) For each file format contained in the library, for each program that claims to accurately identify that file, a user can make a 'claim' against that program. The user making the claim is referred to as 'the claimant'.

3) A claim is comprised of:

c.i) An overall judgement against the capability of the program to identify the specific file format ("works" or "doesn't work"),

c.ii) a written rationale for that judgement.

4) Users who are not the claimant can either "agree, or "disagree" with a claim, and optionally provide a public rationale for their own decision.

Trust is now calculated as follows:

4.i) For each claim, that claim is identified as a Positive Claim, if the claimant proposes that the tool in question 'works', and a Negative Claim, if the claimant proposes that the tool in question 'doesn't work'.

4.ii) For each claim, set a multiplier M that is initially 0.

4.iii) For each user that is not the claimant, if that user "agrees" with a claim, the multiplier M for that claim is increased by 1.

4.iv) For each user that is not the claimant, if that user "disagrees" with a claim, the multiplier M for that claim is decreased by 1.

4.v) For each Positive Claim, a total P is calculated using:

$P = (1 \times M)$

4.vi) All P values (the P value for each Positive Claim associated to a single tool) are summed, and this becomes the total positive value, TP

4.vii) For each negative claim, a total N is calculated using:

$N = (1 \times M)$

4.viii) All N values (the N value for each Negative Claim associated to a single tool) are summed, and this becomes the total negative value, TN

4.ix) We derive from the sum of TP and TN the overall number of user actions against claims, value O.

4.x) We calculate a percentage of users who trust the application to work accurately, value T, as:

$T = ((P \div O) \times 100)$

How a Numerical Ranking is Calculated:

The numerical ranking of a tool tells us how many people overall believe the tool to work, which is a different value from the amount of trust placed in the tool by the community.

1) Value R, is expressed:

$R = (TP - TN)$

Calculating Order of Deployment

The web service will provide a list that outlines in which order tools should be called, and provides information about general community trust for requested file formats. The point of this service is to give users the most efficient sequence for identifying files, based on which files they are likely to ingest (or for all identified files).

1) Rank tools for specific file formats

Before calculating call order, each tool that claims to identify a specific format is ranked relative to that format. For example, if JHOVE, DROID, and XENA all claim to accurately identify Word 97 documents, we first calculate the numerical ranking, and community trust in that tool. Tools are then arranged within that file format in preference of *numerical ranking*. In this case, we rank according to *what most people*

believe works accurately.

2) Optionally identify a subset of file types to preference

We then require a list of the sorts of files a user believes they will encounter most often. Although in most cases this will necessarily be *all* possible types of files, there may be specialised cases where so broad a range of tools is not required.

For example, some government institutes have very specific business rules about what sort of material is archived, and so only a small subset of all potentially discoverable file formats require identification.

Alternatively, an institute might keep internal records about which sorts of files they encounter most frequently, and specify that these are the files they are most interested in identifying.

In both cases, specifying a subset of files does not excluded the identification of files formats that are not included in the set, but skews the order of execution to favour tools that are more likely to capture those specific types of files.

The default in this case is to specify *all* files.

3) Create the operating order based on the relative rankings, and the file type preferences

For each tool registered against the service (e.g., JHOVE, DROID, etc), provide a variable T*n*, and each time that tool is ranked highest inside of a file format that is included in the subset of files, increase T*n* by 1.

The order of execution for this subset is then described by presenting the tools in order of value T*n* descending.

This method can also be used to calculate the order of operation against all files, and these results can be appended to the initial order (excluding tools already identified in the initial results).

Calculating when to stop identifying:

Calculating when to declare a file identified is not the responsibility of the web service. The web service is designed to present the user with enough information to make their own implementation decisions.

However, the web service may also suggest the following methodology, and optionally provide a framework tool to orchestrate execution order:

1) The user or organisation elects a threshold.

The threshold is a percentage that identifies at what point of community trust they are comfortable with calling a file identified. Technically, this value need not be higher than 50%, since, remembering that there is a basic assumption that file format identifiers will either work, or not work, this percentage does not necessarily reflect the quality of the tool, merely that more people believe it to work than not.

This being said, a user is not restricted to a single value. They may elect a more 'democratic' 2/3rds value (75%). Alternatively, a paranoid value (100% - though this would probably use the majority of identification tools!).

2) For each file in the user's manifest (here we assume the user has already generated a manifest that points to each file they wish to identify), run the tools in the presented order of execution, adhering to the following clauses (various methodologies are presented):

(first variation)

2.i) If it so happens that the current tool identifies the file format, and the community trust in that tool is greater than the user's threshold, declare the file as identified, and do not execute any more tools.

2.ii) If it so happens that the current tool identifies the file format, but the community trust in that tool is less than the threshold, then look at the other tools that claim to identify the format 'suggested' by the current tool. If any of those tools have a community trust level that is greater than the user's current threshold, that tool is executed. If that tool provides a positive identification (agrees that the file is of the format suggested by the previous tool), then the file's format is identified, and no more tools are called. Alternatively, if that tool provides a negative result (definitely not the format suggested by the previous tool), then the next tool in the execution order is called. This process can be optimised by recording which tools have already been used, and their proposed format, so as not to call tools twice. (See figure 1 below)

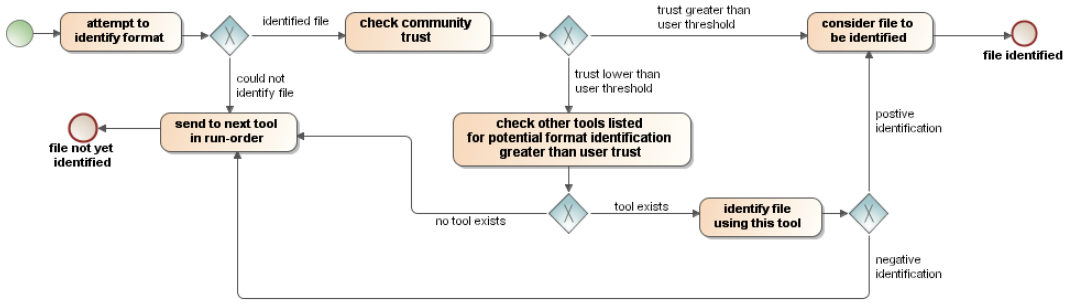


Figure 1 - first variation

(second variation)

2.i) If it so happens that the current tool identifies the file format, and the community trust in that tool is greater than the user's threshold, declare the file as identified, and do not execute any more tools.

2.ii) If it so happens that the current tool identifies the file format, but the community trust for this tool's capacity to identify that particular format is below the user's threshold, move onto the next tool in the chain. (See figure 2 below)

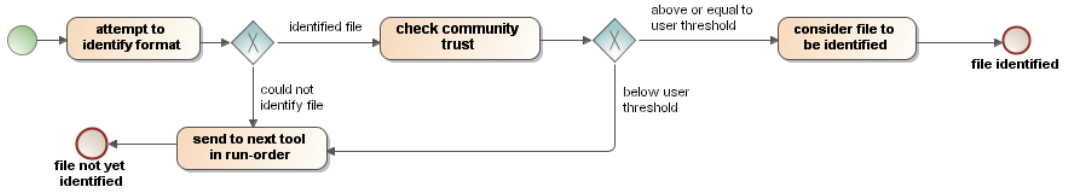


Figure 2 - second variation

(third variation)

this methodology requires an additional value. Instead of a single threshold, the user

identifies an *upper threshold*, and a *lower threshold*. The upper threshold represents the level of community trust at which the user feels confident that the file will have been identified. The lower threshold represents the level of trust at which the user is still not confident that the item has been identified.

For example, the upper threshold might be set at 80%, which may be quite high, and the lower threshold set at 70%, which may still be quite high.

(Remembering that tools are assumed to either work, or not work, a more useful set of values may be to set the upper threshold to 70%, and the lower threshold to 50%)

In this methodology, file identification that is above the threshold is assumed to be correct, and file identification that is inside the threshold values is assumed to be potentially correct.

2.i) If it so happens that the current tool identifies the file format, and the community trust in that tool is greater than the user's threshold, declare the file as identified, and do not execute any more tools.

2.ii) If it so happens that the current tool identifies the file format, and the community trust in that tool is in between threshold values, the file is identified as 'potentially' correct, and we look at the other tools that claim to identify that particular format. If any tool claims to identify that format at a higher level of trust than the user's upper threshold, then that tool is polled, to check whether the initial identification is correct. Additionally, tools that do not claim to identify that format at a higher level of trust than the user's upper threshold, but which are still at a greater level of trust than the previous tool, may also be polled.

2.iii) If the file is identified, but community trust in that tool is below the lower threshold, the identification is assumed to be unreliable, and the next tool in the chain is polled. (see figure
below)

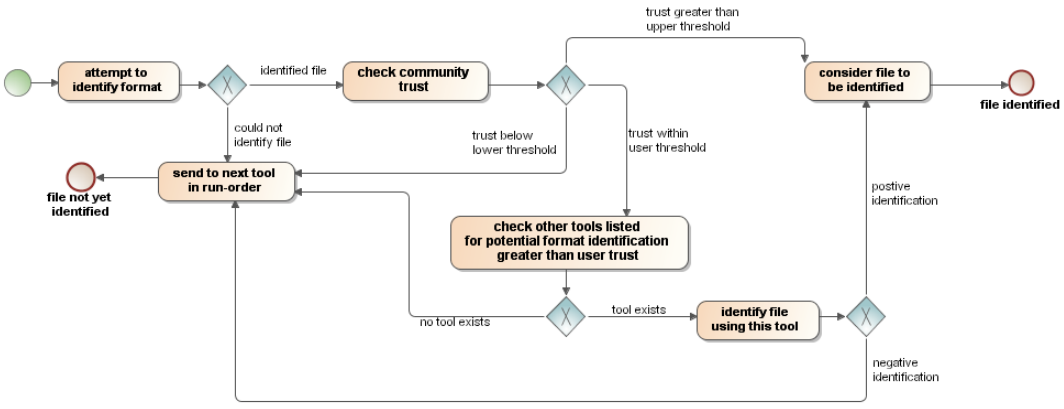


Figure 3 - third variation

For all methodology, the following additional step may be added.

3) At the end of the process, if no certain match is made, then the file is tagged as 'uncertain', and the potential file formats are listed, together with the community trust in those results. In these cases, it may be useful to aggregate the negative scores, as well as presenting the unaltered community trust scores.

This may be done as follows:

1) For each potential result (e.g., JHOVE and DROID may both believe it is Word 97, and XENA believes it is Word 2000 - this would describe two separate 'results'), sum the TP value (the total positive votes), to arrive at value RP*n*.

2) Sum all RPn, to arrive at the total number of participating users (TPU)

3) derive a relative percentage T for each RPn as:

$T = ((RPn \div TPU) \times 100)$

e.g., a file may not be entirely identified as:
90% - Word 97 (40% trust)
10% - Word 2000 (30% trust)

which implies that of all the users that believe it would be a Word 97 document, that makes up 90 percent the total participating users for all results. It also identifies that the level of trust for that result is 40%.

It is presumed that though this does not provide a definite result, it provides enough information for each institution to implement case handling based on their preservation and digitisation policies (in this case, the difference may be irrelevant, as either way the document could potentially be opened in Word 2000).

Getting Information From the Service:

As a web based service, it would be sufficient for most institutions to simply poll the website, and retrieve the most current data as XML. This information could either be recalculated for every request, or alternatively, if this would present a performance issue, most of the information could be calculated hourly (such as overall trust levels across tools and formats), with only the information specific to each user request calculated at the time of request (such as call order for specific file format ranges).

Additional Possibilities:

The basic framework for this system could be improved, or expanded upon in various ways:

- Users could submit information at the time of ingest about what files are being identified, which could help dictate what the service sees as the most commonly encountered files (this could help define some set variables such as "MOST COMMONLY ENCOUNTERED" in the case of the initial subset of preferred files.

- The ranking and voting system could potentially be expanded to include other digital preservation related services. For example, to decide which tool is best suited to normalising a specific type of file format (together with detailed information about which information is kept/lost during the transition).

- The voting system could be applied in an obsolescence capacity, which may in turn contribute to the matrix of information that dictates the operation of programs such as AONS II.

- If a uniform way of identifying files was established (e.g., a 'fingerprint', or 'magic number'), then results could be checked against the fingerprint, rather than the result derived from any one tool.

- e.g., the 40% 'finger print' for a Word 97 document might also give a 90% result for an MP3 file.

- users could provide a list of the identification tools they currently operate with to the service, and assuming those tools are registered, the service could provide the user with an execution path that is specific to those tools (i.e., the system will not ask the user to execute a tool they do not own).

- The system could be fine tuned to deal with results duplicates, or tools that represent a subset of the capacity of other tools. E.g., if DROID identifies all the formats that JHOVE identifies, and more, at equal or greater level of trust, only DROID need ever be called.