




Final version of the Preservation Watch component

Authors

Luis Faria (KEEP Solutions), Kresimir Duretec, Artur Kulmukhametov (Vienna University of Technology), Per Moldrup-Dalum (State & University Library Denmark), Leila Medjkoune, Radu Pop, Stanislav Barton (Internet Memory), Alan Akbik (Technische Universität Berlin)

March 2014

This work was partially supported by the SCAPE Project. The SCAPE project is co-funded by the European Union under FP7 ICT-2009.4.1 (Grant Agreement number 270137).

This work is licensed under a CC-BY-SA International License 

Executive Summary

The implementation and testing of Scout, the preservation watch system, is described in this report. Scout is the final result of the development of a preservation watch component, included in the SCAPE digital preservation suite, which integrates preservation watch with planning and operations.

Scout is a system that allows the definition of *source adaptors* - extensible pieces of software that can be configured to gather information from aspects of the world - and consolidates all of this information in to a centralized knowledge base. In the web interface, users can browse and query the gathered information, cross-referencing it to find preservation threats. Scout further allows the creation of triggers, i.e. questions (technically queries) that are periodically monitored, sending a notification to the user when non-conformities are found.

Several source adaptors were developed that allow monitoring of content and events from repositories and web archives, and that allow gathering information from the environment, i.e. file format registries and raw information from the Web. Furthermore, several experiments were carried out to test the limits of the system.

A survey to the digital preservation community was executed to find the community's opinion of what are the most important preservation threats, the preferred methods to detect and monitor them, and the current practice in preservation watch. The results show that the prototype fulfils the most important requirements from the community and delineate a roadmap for future developments.

Table of Contents

1	Introduction	1
1.1	The preservation lifecycle	1
1.2	Preservation Watch.....	2
1.3	Goals.....	2
1.4	Document structure	3
2	Preservation Watch Component.....	3
2.1	Scout: a preservation watch system	3
2.2	C3PO: Clever, Crafty Content Profiling of Objects	7
2.3	Integration with planning and operations	8
3	Source adaptors	9
3.1	Digital Object Repository	9
3.1.1	Digital content	10
3.1.2	Events	10
3.1.3	Experiment of full preservation lifecycle.....	11
3.2	Web archive.....	11
3.2.1	Digital content	11
3.2.2	Browser renderability.....	21
3.3	Environment.....	24
3.3.1	File format registries	24
3.3.2	Information from the Web.....	24
4	Beyond the prototype	24
4.1	Survey: “Digital preservation: what to monitor and how?”	25
4.1.1	Invitation	25
4.1.2	Profile	25
4.1.3	Digital preservation threats.....	27
4.1.4	Detecting and monitoring preservation threats	30
4.1.5	Follow-up.....	38
4.2	Analysis of the prototype against survey results	38
4.3	Roadmap for future developments.....	39
4.3.1	Research data.....	39

4.3.2	Human input.....	39
4.3.3	Simulator	40
5	Conclusion	40
6	References	41

1 Introduction

Digital assets are continuously endangered by preservation threats that can hinder user access or even cause irreparable loss to the correctness or authenticity of valuable content. These threats belong to many domains, from technological to organizational, economic and political, and can relate to the holder of the content, the producers or to the target communities to which the content is primarily destined for.

In digital preservation, watch (also called monitoring) is a key capability that enables the early detection of these threats (Antunes et al., 2011). However, as the volume and heterogeneity of assets increase, it becomes unfeasible to manually monitor all aspects of the world that may hinder their preservation. Furthermore, monitoring should not only detect symptoms of preservation risks but also opportunities (e.g. cost reduction) and ensure that preservation actions, as defined by decision-making processes, continue to meet goals and fulfil expectations. Considering the problem scale, the automation of the monitoring process becomes a necessary step to ensure proper digital preservation (Faria et al., 2012).

The key goal of this deliverable is to provide a proof-of-concept system, i.e. a prototype, which enables automated mechanisms to support the monitoring and evolution of preservation plans over the digital object's lifecycle. This report presents the outcome of the development efforts to create a preservation watch component as described in the D12.1 – Identification of triggers¹ and preservation Watch component architecture, subcomponents and data model (Duretec, Faria, Petrov, & Becker, 2012). As this is a prototype deliverable, this focuses on the information that is not available with the software or papers produced on the project, and will point to relevant external sources of information as much as possible.

1.1 The preservation lifecycle

This section presents an encompassing vision for the preservation planning, watch and operations processes which allow for long-term preservation of digital information by continuously monitoring internal and external influences and adapting repository content accordingly. Preservation watch is the process that monitors the world (of interest) and notifies the relevant parties when possible risks or opportunities occur. The common relevant party would be a Planner, i.e. a person who as the role to manage the digital preservation of the repository content, e.g. a content manager. Preservation planning is the process that analyses the current situation and helps the Planner to decide what action to take (if any action is needed). Preservation operations are the processes in which the content or the repository is changed according to defined plans, in response to risks or opportunities.

¹ In the context of preservation watch, and in this deliverable, a trigger means a condition, on the properties measured from the world, which if true runs a pre-determined action, usually a notification to the user.

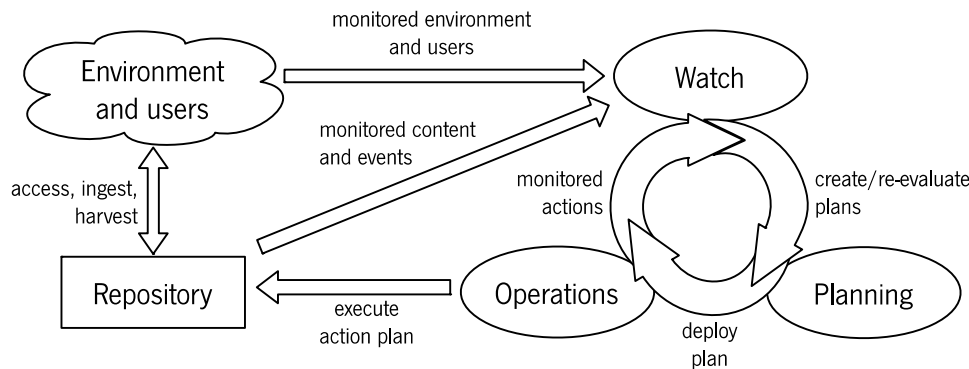


Figure 1 - Digital preservation lifecycle

Figure 1 illustrates the whole preservation lifecycle, details are described in (Becker, Faria, & Duretec, 2014). One of the main components is the watch component. Its role is to detect risks or opportunities which are related to digital content. It monitors a repository which stores digital content and the environment around it. The repository is monitored in two ways: 1) repository digital content and 2) repository events.

The characterization process extracts the key characteristics from the digital content held in the repository. Those characteristics are aggregated and fed to the watch component. Repository events are the second aspect which is monitored by the watch component. This enables prompt reaction on any event which happens in a repository. This can include events such as ingest or access but also such as execution of preservation plans.

Watch furthermore validates collected information about collections with institutional policies which are in a machine readable form and notifies preservation planning process which can then address detected violations.

Planning goes on into creating a preservation action plan, which is carried out in the repository by Operations. The action plan can include quality assurance tools that measure the performance of the plan. Watch is able to gather information on quality assurance via the repository events and notifies if the quality of the results is below expected, requiring plans to be re-evaluated. Also, if environment changes make plans not valid anymore, watch can also notify that plan re-evaluation is needed. This creates a continuous cycle that ensures content remains aligned with the requirements for digital preservation.

1.2 Preservation Watch

This deliverable focuses on the development of a software component that supports the preservation watch process. This process relates to the detection and monitoring of preservation threats by getting information of entities and properties of interest from the world, i.e. internal or external influences, and cross referencing this information. More information on the design of the preservation watch process and the software component is available on (Becker et al., 2012; Duretec et al., 2012; Faria et al., 2012).

1.3 Goals

The preservation watch component should be capable of monitoring content characteristics and repository events, allowing these to be cross-related with information from the outside world, to

identify preservation risks or opportunities. Also, it should be capable of working together with the planning tool to identify when a plan becomes “invalid” and should be re-evaluated.

From deliverable D12.1 – Identification of triggers and preservation Watch component architecture, subcomponents and data model (Duretec et al., 2012), the defined high-level goals are to:

1. Enable automatic monitoring entities and properties of interest;
2. Enable human users and software components to pose questions about entities and properties of interest;
3. Collect information from different sources through adaptors;
4. Act as a central place for collecting relevant knowledge that could be used to preserve an object or a collection;
5. Enable human users to add specific knowledge;
6. Notify interested agents when an important event occurs;
7. Act as an extensible component.

The rest of the document will describe how these goals are achieved by the development of a Preservation Watch Component and a set of Source adaptors. Several experiments further validate the viability of the approaches taken.

1.4 Document structure

Section 2 describes the Preservation Watch Component and how it integrates with the rest of the SCAPE preservation suite (i.e. planning and operations). Section 3 lists developed and planned source adaptors, that will get information from outside world into the system, and experiments made to test the system. Section 4 goes beyond the prototype, presents the result of a survey to analyse the community opinion on the most important preservation threats, the preferred methods to detect and monitor them, and the current practice, using this to define a roadmap for future developments. Finally, section 5 draws the final conclusions and a summary of how the goals are achieved.

2 Preservation Watch Component

2.1 Scout: a preservation watch system

The primary output from this deliverable is the Scout tool, version 0.3.0². Scout³ is an automated preservation watch service which supports the scalable preservation planning process by collecting and analysing information on the preservation environment (internal and external), pulling together information from heterogeneous sources and providing coherent unified access to it. It addresses the need to combine an awareness of the internal state of an organization and its systems (internal monitoring) with an awareness of the environment in the widest sense (external monitoring) to enable a continued assessment of the alignment between the two (Faria et al., 2012). Scout has a typical three layer enterprise application architecture: infrastructure layer, domain layer and presentation layer.

The infrastructure layer provides a linked-data based storage system. Once information is collected, it is saved in a controlled data model to the knowledge base (Faria et al., 2012). Built upon linked data principles, the knowledge base supports basic reasoning and analysis of the collected data using

² <https://github.com/openplanets/scout/releases/tag/v0.3.0>

³ <http://openplanets.github.io/scout/>

standard mechanisms such as SPARQL⁴. Such queries provide the mechanisms for cross-referencing information and automatic change detection.

The domain layer enables to gather information from different sources and process collected information before storing it to the knowledge base. It also enables managing different triggers which contain questions (with accompanying watch conditions) about the knowledge base status. By adding a watch trigger (also called request), which includes a SPARQL query, the results will be monitored periodically. When the condition is met, a notification is sent to the user. Triggers can cover arbitrary circumstances of relevance in the known domain, ranging from checks on content validity and profile conformance to certain constraints to the question whether any new tools are available to measure a certain property in electronic documents, or whether a Quality Assurance tool that is in use for validating authenticity of converted images is still considered reliable by the community.



Figure 2 - Scout information flow

The information is collected by implementing different source adaptors, as illustrated in Figure 2. Source adaptors are pieces of software, with accompanying metadata about the data source, which can fetch data from the outside and map it to the internal knowledge base. Scout has no restrictions on the types of data that can be collected. It is built to collect a variety of data from different sources such as format and tool registries, repositories, and policies. It already implements source adaptors for the PRONOM registry, content profiles⁵, repository events (ingest, access, and migration), policies and other specific adaptors. See section 3 for information about the several implemented source adaptors.

⁴ <http://www.w3.org/TR/rdf-sparql-query/>

⁵ Content profile is a summary of content technical characteristics such as file format distribution or compression scheme, image size, etc.

Administration

Private page to manage the configurations of scout.

Source Adaptors

All configured source adaptors that fetch information from external sources using plug-ins.

Instance	Plug-in name	Plug-in version	Source	Active	Actions
im-webarchive	c3po	0.0.9	IM	✓	⚙️
im-browsershots	c3po	0.0.9	IM	✓	⚙️
sb-webarchive	c3po	0.0.9	SB	✓	⚙️
keeps-roda-demo-content	c3po	0.0.9	keeps-roda-demo	✓	⚙️
keeps-roda-demo-events	Report API Adaptor	0.0.2	keeps-roda-demo	✓	⚙️
pronom	Pronom Adaptor	0.0.6	pronom	✓	⚙️

New source adaptor

Installed plug-ins

All installed plug-ins. To install a new plugin just drop the plug-in jar into the correct folder under `/usr/local/scout/plugins`.

Name	Version	Description	Type
Pronom Adaptor	0.0.6	A Scout adaptor for the PRONOM registry.	ADAPTOR
Report API Adaptor	0.0.2	A Scout adaptor for a repository Report API.	ADAPTOR
c3po	0.0.9	A scout adaptor for the c3po content profiler source	ADAPTOR
HtmlEmail	0.0.3	Send notification via email	NOTIFICATION

Figure 3 - Scout web interface showing the administration page

Scout has a simple web interface, Figure 3, which allows operations like management, adding new adaptors and triggers, and browsing the collected data.

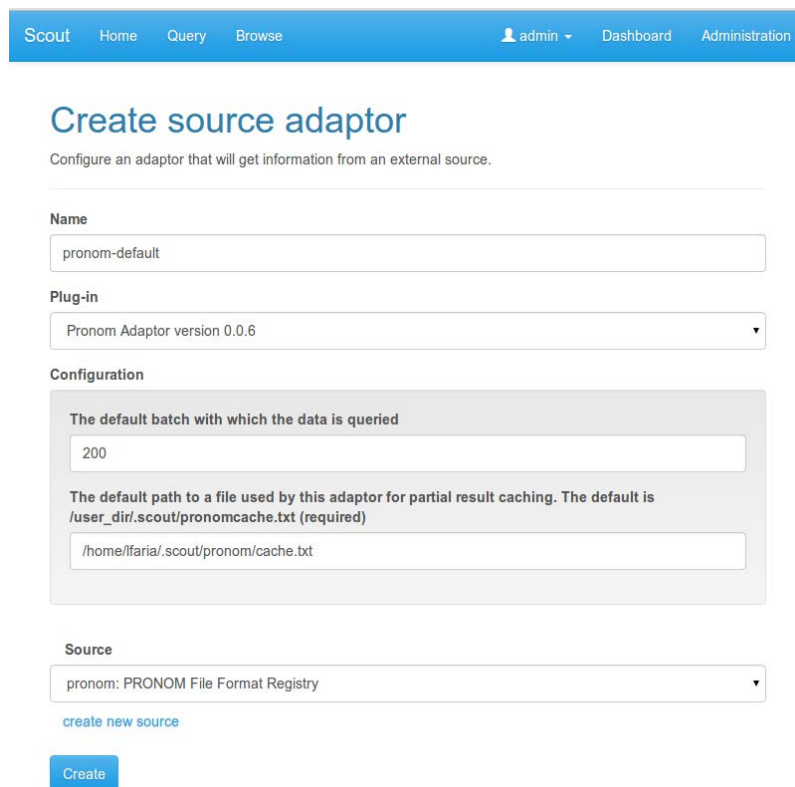
After installing Scout⁶, you can login as a user with administration permissions to access the Administration page (default username and password is “admin”). On this page there is a listing of available plugins and defined source adaptors.

An adaptor plugin is a piece of software, developed in Java, which can be configured to fetch information from a specific platform. To develop new plugins you just need to implement a defined interface and package the software on a specific way⁷. Currently, available adaptor plugins are for PRONOM, C3PO and Report API, section 3 will have more information on how this specific plugins are used. Every adaptor plugin can define a set of parameter it needs to be set up. There are also notification plugins, which allow new ways of sending notifications to users. The only current notification plugin is the HTML email, which sends notification as an e-mail.

⁶ Install instructions at <http://openplanets.github.io/scout/#how-to-install-and-use>

⁷ Details on developing plugins at <https://github.com/openplanets/scout/wiki/Adaptor-Development>

To fetch information from a defined source, you need to create a source adaptor. A source adaptor defines which plugin will be used, what are the parameters of the plugin, and metadata about the source of information and that source adaptor in particular. A plugin can be used in several source adaptors, spawning several instances of the same plugin but with different configurations.



Scout Home Query Browse admin Dashboard Administration

Create source adaptor

Configure an adaptor that will get information from an external source.

Name

Plug-in

Configuration

The default batch with which the data is queried

The default path to a file used by this adaptor for partial result caching. The default is /user_dir/.scout/pronomcache.txt (required)

Source

[create new source](#)

Figure 4 - Scout web interface showing the creation of a new source adaptor

For example, in Figure 4, you can create a PRONOM source adaptor, which will fetch information from the PRONOM file format registry. The source will be defined as the PRONOM registry, and the adaptor can be named, for example, as “pronom-default”, because it makes no sense in this case to have more than one adaptor for this platform. Also, you can define several adaptors for your repository, using the C3PO plugin and the Report API plugin, but the same Source. To add a new adaptor you just need to click the “New source adaptor” button on the Administration.

Adding these and other adaptors will populate Scout knowledge base with information, which can be browsed on the Browse menu option. On the Dashboard, a user can upload their machine readable policies using the SCAPE Policy Model (Sierman, Jones, Bechhofer, & Elstrom, 2013).

On the Query menu option, Figure 5, you can query the knowledge base, cross-referencing information and using the reasoning features of the linked data platform. You can use SPARQL on the advanced query, or select one of the query templates in the simple query. These queries can reason on all data in the knowledge base, including the policies. For example, you can check if a defined collection, processed by C3PO and integrated into the system via the adaptor, conforms to the defined control policies in terms of compression scheme. By clicking “Search” button, the system will search for non-conformities and present them. By clicking “Create trigger” the defined query will be

used to create a watch trigger (also called watch request). Here you can define the monitoring period and the email where notification should be sent if non-conformities are found. At any time you can manage the trigger in the Dashboard menu.

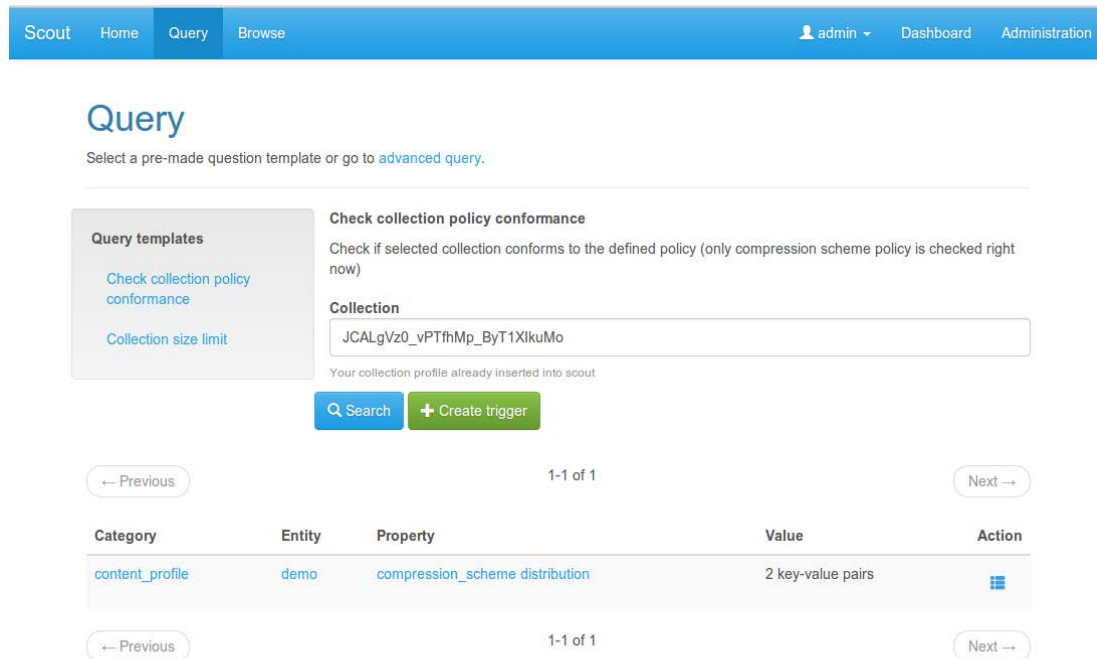


Figure 5 - Scout web interface showing the simple query

The notification email describes which query triggered the notification. The email could also list the non-conformities found and the related control policies that are being violated, but these are enhancements listed for future work.

2.2 C3PO: Clever, Crafty Content Profiling of Objects

C3PO – or ‘Clever, Crafty, Content Profiling of Objects’ is a software tool⁸, which enables a detailed content analysis of large-scale collections (Petrov & Becker, 2012). It uses metadata extracted from files of a digital collection as input to generate a profile of the content set. The tool transforms the data for faster and scalable analysis and stores it, then post-processing solves issues like conflict resolution and provides a machine-readable overview, and a web application enables the user to filter and explore any part of the data further.

C3PO provides end-users with in-depth knowledge of their content. It uses extendable filtering techniques, which allow dividing complex heterogeneous collections into homogeneous sets based on the characteristics. Content analysis may be done through the web-based application which brings effective visualization of content. Current state-of-the-art technology (MongoDB⁹) used in the backend successfully tackles scalability issues when processing big amount of data.

⁸ <http://ifs.tuwien.ac.at/imp/c3po>

⁹ <https://www.mongodb.org>

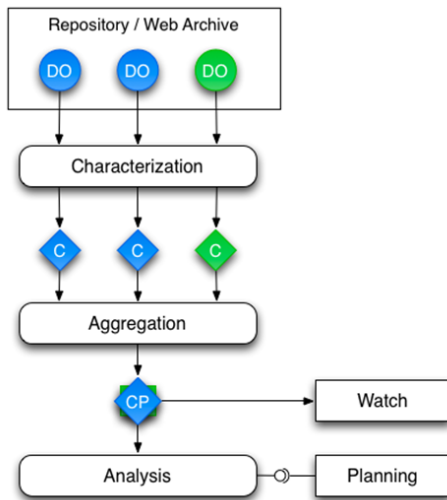


Figure 6 - The steps in content profiling workflow

Figure 6 provides a high level overview of processes occurring in C3PO. The tool uses characterisation results of a digital collection from a repository as input, aggregates them and generates a profile of the content set in an automated manner. It outputs a detailed content profile describing the key characteristics of a collection.

This content profile may be used by watch component Scout through C3PO API for immediate collection monitoring. Also, C3PO provides facilities for data export and external analysis of the content. Further details are available in (Becker et al., 2014).

2.3 Integration with planning and operations

The vision presented in section 1.1, the preservation lifecycle, has been implemented by a publicly available set of components and API specifications that can be freely integrated with any repository system. The suite of components aims to provide the tool support necessary to enable organizations to advance from largely isolated, ad-hoc decisions and actions reacting to preservation threats to well-supported and well-documented, yet scalable and efficient preservation management. The following section describes the main interface points between each of the key building blocks of this tool suite and points to references for further in-depth information.

Figure 7 depicts the SCAPE software components supporting the preservation lifecycle and how the software tools described above integrate with the overarching preservation lifecycle system presented in section 1.1.

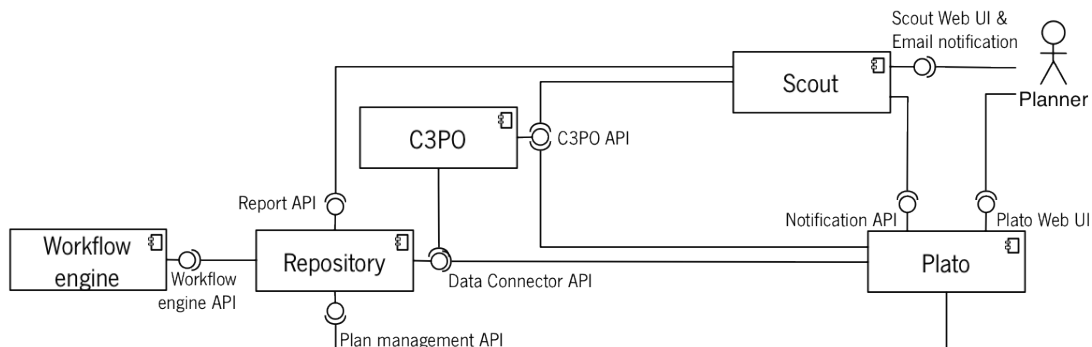


Figure 7 - SCAPE software components supporting the preservation lifecycle

There are APIs, presented in Figure 7, which integrate the planning component Plato, the watch component Scout and the content profiling component C3PO. These tools and APIs altogether constitute The SCAPE Planning and Watch suite. The following APIs are in the suite:

- **Data Connector API:** It allows basic operations on content in the repository e.g. reading, writing, updating and searching.
- **Report API:** This API is responsible for distributing events happening in the repository. The use of this API is further detailed in section 3.1.2.
- **Notification API:** It allows notifications such as emails sent to users in case of specified events triggered.
- **C3PO API:** Having several content profiles of one collection, Scout is able to generate a historic profile for the collection through C3PO API. A historic profile shows growth and changes of chosen features in the data collection such as formats, collection size etc. The integration with Plato provides Plato with a content profile of a collection, including statistics and sample objects of the collection.
- **Plan management API:** Plan Management API is needed for deployment and management of preservation plans in the repository.

Detailed description of the APIs and information flow is available in (Becker et al., 2014).

3 Source adaptors

This section describes the several source adaptors developed to gather different types of information from the World.

3.1 Digital Object Repository

Digital Object Repositories are systems that control and manage digital objects through all their lifecycle and are common in memory institutions like archives and libraries. Due to the long-term public interest on content from these domains, a Digital Object Repository usually has digital preservation concerns and strives to follow architectures for long-term preservation such like OAIS.

Next it is described how digital content kept in the repository is analysed and monitored, how the repository events as ingest, access and execution of preservation actions are gathered, and how the repository and the preservation watch component integration with planning and operations.

3.1.1 Digital content

Content characteristics are monitored using C3PO and the C3PO adaptor on Scout. To monitor content which is already in a repository, some integration with the repository system is needed in order to run a characterization tool which output is supported by C3PO, like FITS¹⁰ or Apache Tika¹¹, on all objects of the repository, and export the result to a directory in the file system.

An ad-hoc implementation of this step was implemented for the RODA repository¹² using a plugin that would periodically go throughout all objects, run FITS on every file, and save the output result to a specific directory in the system. This directory can then be periodically monitored by C3PO and Scout can monitor this C3PO endpoint using the C3PO adaptor. This architecture was deployed on a test instance in KEEP SOLUTIONS¹³ and used in the full preservation lifecycle experiment presented in section 3.1.3.

C3PO could use directly the Data Connector API¹⁴, which would allow a general implementation for the access of repository content. This would allow C3PO, and transitively Scout, to be integrated with repositories without the need for the implementation of specific components. Such enhancements are defined as future work.

3.1.2 Events

Events that occur in repositories can be of interest for the digital preservation of content. These events can relate, for example, to ingest and access of content and they can reveal trends and problems with producers and consumers. Also, the result of preservation actions executed on the repository, especially if they include the execution of quality assurance tools, can be very important to monitor if actions are having the results expected in preservation plans.

To support the monitoring of all this information, a Report API was developed which defines a specification of a service that allows harvesting of repository events¹⁵. The Report API is, in a nutshell, an OAI-PMH provider of repository events encoded in the form of PREMIS events. A reference implementation of the Report API is available for the RODA repository¹⁶ and a set of reusable components to develop the report API for other repository implementations is also available¹⁷.

In Scout, a Report API adaptor was developed to allow harvesting of the repository events available via Report API implementations. The adaptor harvests the available events and aggregates them into summary values. For example, for ingest it calculates the average time, and for quality assurance outputs from preservation actions, it calculates the maximum, minimum and average of numeric outputs and the frequency distribution of non-numeric outputs. The adaptor can be changed to calculate other aggregated summary of values to fit a specific purpose.

¹⁰ <http://projects.iq.harvard.edu/fits>

¹¹ <http://tika.apache.org>

¹² <http://www.roda-community.org>

¹³ <http://www.keep.pt>

¹⁴ Data Connector API specification available at <https://github.com/openplanets/scape-apis>

¹⁵ Report API specification available at <https://github.com/openplanets/scape-apis>

¹⁶ Report API reference implementation in RODA at https://github.com/openplanets/roda/tree/master/roda-core/roda-core-services/src/main/java/eu/scape_project/roda/core/report

¹⁷ Reusable components for Report API implementation at <https://github.com/openplanets/report-api>

3.1.3 Experiment of full preservation lifecycle

To demonstrate how the presented tools support the preservation lifecycle, we consider a simple scenario. Let's assume that an institution has its content in place (stored in a digital repository, which was RODA on this experiment) and preservation policies defined. As preservation policies are not the topic of this deliverable details on creating them and making them machine readable can be found in the SCAPE deliverables D13.1 and D13.2 (Bechhofer et al., 2013; Sieman, Jones, & Elstrom, 2014).

The most important thing for the content manager now is to see if the content conforms to the specified policies. Let's assume that the content manager already has the characterization data extracted from his content with tools such as FITS (see section 3.1.1 on how this was done for RODA). Now the content manager is able to use C3PO to aggregate those data and provide meaningful statistics about the state of the content. Even though this kind of analysis can already reveal potential policy violations it is not scalable in terms of the fact that the content can change over time and that there could be a number of different collections. To enable automatic monitoring of its content the content manager will therefore use the watch component (Scout) and install an adaptor which automatically monitors the content from C3PO. Having preservation policies deployed in the watch component watch component, the content manager is able to automatically check policy conformance for a specific content.

On the experiment, the selected scenario has a policy defining a condition that all the content must be in a lossless compression and that there is a content which has a lossy compression. In this case the watch component detected the violation and sent a notification to the content manager. After receiving the notification, the content manager was able to create a preservation plan to address the detected risk. As preservation planning is not the topic of this chapter all the details can be found in the SCAPE deliverable D14.1 (Hamm & Becker, 2011) and upcoming deliverable D14.2.

Once the preservation plan was created, it was deployed to the digital repository (RODA) and executed from there. As the plan was being executed it provided a number of different measures, coming from the quality assurance components included in the plan. Those measures were aggregated by the repository and provided to the watch component (via the Report API, see section 3.1.2). The watch component monitored the provided plan performance and, when it was below the expected values, it raised a new notification. This experiment demonstrates the complete the full preservation lifecycle presented in section 1.1.

3.2 Web archive

Web archives preserve a very important part of the cultural heritage of the world and are, at the same time, a representative sample of the technological landscape, enabling the inference of trends in technology. Also, web archives are a good example of the large-scale issues common in many institutions.

Next, several experiments on analysis and monitoring of the digital content in web repositories are presented, allowing each to reveal the limitations and potentialities of the preservation watch tool.

3.2.1 Digital content

Two experiments were made using deep characterization of web archive data: one was set to find limitations of the used tools (FITS, C3PO and Scout) for processing a large scale amount of data in a real world situation, while the other was set to find procedural and bias problems when focusing on a well curated set of data relative to the archiving of prominent domains over the time span of 10

years. Both experiments portray real world situation that occur in large institutions with the mandate to archive big sets of the web, such as the Danish National Library and the Internet Memory Foundation.

3.2.1.1 Experiment with large scale data

Extracting metadata from large scale data collections is a daunting task. Analysing it for anomalies, structure, patterns, format profiles, etc. is no lesser task. We employed the File Information Tool Set (FITS)¹⁸ on a large data set, ingested it into C3PO for profile extraction and analysis and then imported it into Scout to make it available for automatic Planning and Watch.

Deep characterization of content with FITS

To provide data for the Planning and Watch sub project we have run the File Information Tool Set on a 12TB sample from the Danish web archive.

This web archive consists of web resources that have been harvested by crawling the Danish part of the Internet since 2005, i.e. from every publicly available URL on the Danish top level domain “.dk”. During the crawl process the web resources are collected in uncompressed ARC containers storing approximately 100MB each that on average corresponds to 3600 web resources. These ARC files are then stored in the Danish web archive Netarkivet¹⁹. For the SCAPE project the State and University Library has selected a representative sample covering all the years for which data has been harvested. Table 1 lists the number of ARC files per year. Due to Danish legislation this sample cannot be made available for the partners in the SCAPE project, but the State and University Library can perform experiments on it for the SCAPE project.

Table 1 - Size of data sample

Year of harvest	Number of ARC files
2005	4,024
2006	20,497
2007	17,139
2008	30,685
2009	23,019
2010	14,090
2011	13,386
2012	17,897
Sum	140,747

As the acquisition of characterisation data was the main issue, and because we initiated this characterisation in the early days of the SCAPE project, we implemented a simple Bash based approach²⁰. The characterisation process was executed on a group of machines described in Table 2. The ARC files are stored on a SAN and accessed using sash/scp. The processing load was handled manually by giving each machine a manually defined subset of ARC files to work on.

Table 2 - Hardware specification

CPU	Intel® Xeon® processor X5670 (12MB cache, 2.93GHz, 6.40 GT/s Intel® QPI)
CPUs/Cores	2 CPU per server, each CPU with 6 hyper threaded cores

¹⁸ <http://fitstool.org>

¹⁹ <http://netarkivet.dk>

²⁰ <https://github.com/statsbiblioteket/SB-Fits-webarhive>

RAM	288 GB in total
Network	2Gbit
Operating System	CentOS

The job was initiated in November 2011 and ran more or less non-stop until April 2013; resulting in 106GB XML data distributed over 140,000 files each with an average number of 3641 records per ARC file (437,000,000 web resources total). A quantitative analysis of this data is presented in the blog post *A Year of FITS*²¹.

The acquired data was during the characterisation process made available for ingest into C3PO through a simple HTTP interface for SCAPE internal use.

The study “Evaluation of characterisation tools - Part 1: Identification” (Kniff & Wilson, 2011) presents an analysis of how FITS performs in general.

Processing of FITS output with C3PO

The result of the previous step is a large number of FITS output files in XML. To enable the analysis of the deep characterization output, the C3PO tool was used to process the FITS output and enable real time analytics on the global result and transitive integration with Scout. But the C3PO tool had not been tested with such a large volume of data before. To evaluate how this tool would perform on a scale larger than a typical desktop computer, we tried to ingest the characterisation data acquired from the above-presented process. At the time of this C3PO evaluation only data was present for the years 2005 until 2011.

We tested the three main functionalities of C3PO:

- 1) Ingest
- 2) Profiling
- 3) Exploratory user interface

We used version 0.20 of C3PO running against version 2.4 of MongoDB²² in Tomcat²³ version 7. All performance tests were performed on a server with the specification described in Table 2.

Ingest

The first test was to ingest the complete data set of circa 440,000,000 FITS output files, which were the result of processing the FITS tool on a 12 TB sample explained in the previous section. During the characterisation process described above, the FITS files were organised in TGZ files (i.e. compressed TAR archives²⁴), one for each ARC file. The files were organised in circa 123,000 TGZ files with an average of 3,600 FITS data per TGZ. The ingest process first extracted the data from the TGZ before running the Java based C3PO ingest process. The ingest times for these extracted TGZ files are visualised in Figure 8 where each line represents a TGZ sample which again represents an ARC file.

²¹ <http://www.openplanetsfoundation.org/blogs/2013-01-09-year-fits>

²² <http://www.mongodb.org>

²³ <http://tomcat.apache.org>

²⁴ <http://www.gnu.org/software/tar/>

Figure 8 shows that the ingest time is linear with the data size with only a few outliers. For more about outliers in this data sample see *A Year of FITS*²⁵. The data implies that C3PO is able to ingest large data sets, probably at least in tens of TB in size.

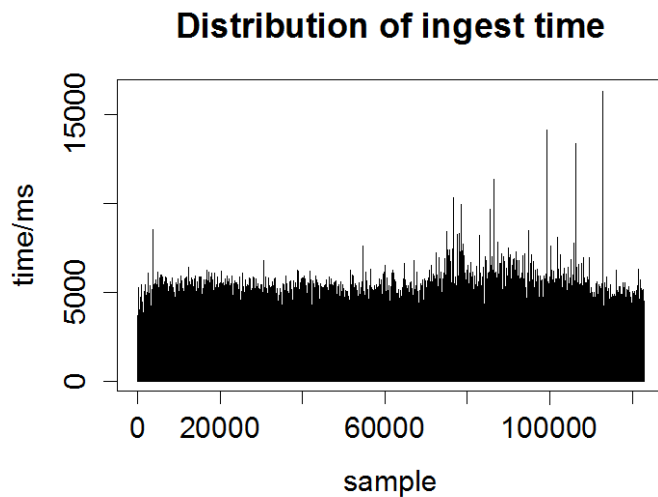


Figure 8 - Ingest times for the complete set of TGZ files

Profiling

C3PO has a command line interface for extracting profiles of the ingested data. This functionality can be used to create data for e.g. Scout. We tested this functionality on two samples: the 2005 data and the complete set. As can be seen in Table 1 we have circa 4,000 TGZ files for the year 2005. These TGZ files contain 11,905,931 FITS output files and a content profile was calculated in 15 hours and 18 minutes. The complete set contains 441,923,560 FITS output files and we let the ingest process run for 60 hours. This showed that the C3PO ingest process could run for at least that long without crashing. If the implementation of the algorithm behaves linearly as expected, we would have had to wait for another about 22 days for completion. This would not have provided additional value so the process was terminated. This experiment shows that the tested version of C3PO is not fit for handling complete collections.

Exploratory user interface

C3PO has a very nice graphical user interface for exploring extracted properties and correlations. We knew beforehand that this user interface would not be able to handle the ARC data set because previous experiments showed it could not provide the real time analytics available on the user interface on data of this volume scale. In this test we wanted to explore how much data could actually be handled by the application while maintaining usable performance.

This was done by importing a data set of a given size and then measuring the time needed to show the first screen with the overview plots in Figure 9. Next step was to go into the first bar on the first plot by clicking it and measuring the time it took to show the second plot.

²⁵ <http://www.openplanetsfoundation.org/blogs/2013-01-09-year-fits>

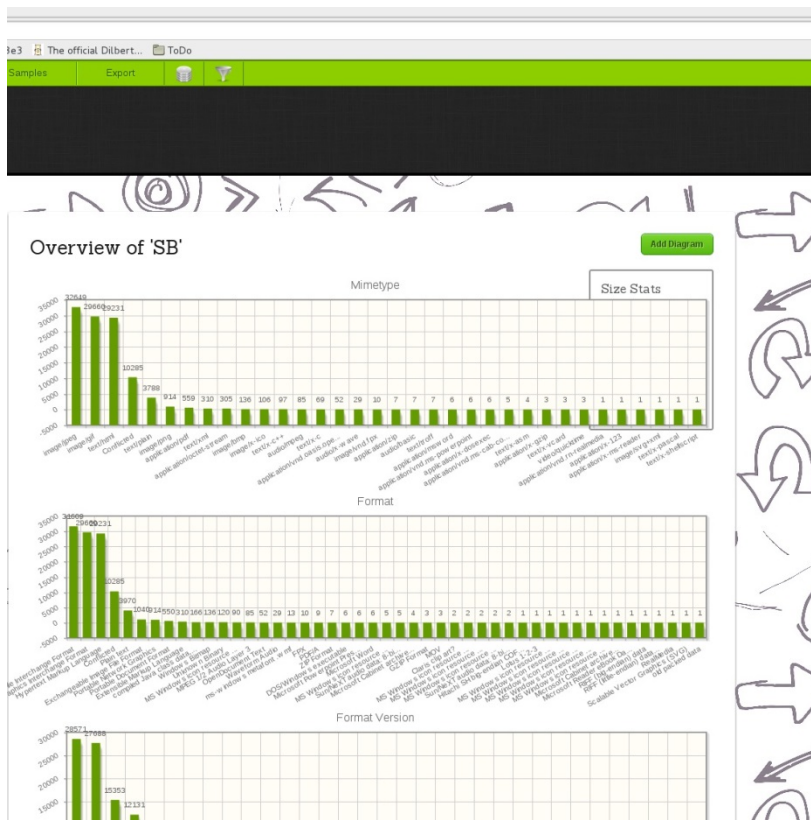


Figure 9 - Collection overview page in C3PO

The results are shown in Table 3 and as can be seen C3PO stopped being responsive somewhere close after 2.5 million FITS result files. In the table “Elements” and “Properties” are both MongoDB specific collections and they both grow through the ingest process.

Based on this performance test it is evident that C3PO at the time needed much more work on optimisation. Some of these optimizations are detailed in the next section.

Table 3 - Responsiveness of user interface

Run #	Number of FITS files	Size of elements (GB)	Total elements index size (GB)	Number of properties	Size of properties (kB)	Properties index size (kB)	Processing time for 1 st plot	Processing time for 2 nd plot
1	13,962	0.03	0.01	80	5	16	Fast	Fast
2	108,348	0.26	0.04	96	6	16	18s	11s
3	363,991	1.00	0.15	106	7	16	30s	34s
4	1,020,514	2.46	0.41	113	8	16	2m 25s	1m 42s
5	1,639,842	3.95	0.67	119	8	16	3m 52s	2m 50s
6	2,683,596	6.44	1.10	119	8	16	6m 28s	4m 25s
7	11,905,931	28.63	4.83	211	15	16	>3h	N/A
8	441,923,560	1183.50	192.45	5,122	492.88	687	N/A	N/A

Enhancing C3PO for this experiment

During SCAPE Y3, it was a challenging task for C3PO to produce a content profile for 0.5 billion FITS characterisation results obtained from SB Web Archive. The rough estimations based on the data from the previous section show that it would take several months of processing time on a single server without sharding to ingest all the data in MongoDB and run aggregation procedure to

generate a content profile. Further investigations pointed out at disk IO performance bottleneck, which occurs due to a small size of a characterization metadata file and a big amount of such files.

To solve this problem, C3PO was extended with an adaptor that generates a content profile of characterisation metadata directly read from storage. This improvement became possible due to a modular architecture of C3PO, where functionality of components may be easily extended through APIs. A gathering interface was implemented in a way that all needed data statistics aggregation and analysis is done on the fly, just after file read. There is no intermediate step of storing data in a database with following extraction of data. Also the adaptor was modified for parallel execution which scaled-up computations.

In order to evaluate the adaptor, a dataset was generated from a subset of FITS characterization results of SB Web Archive Data. A desktop PC with Intel 4 cores, 8GB RAM, 250GB Hard Drive and Ubuntu 13.04 OS was used for the tests. For experimentation, 10 first TGZ packages from each year of the collection (i.e. from the years 2005-2012) were selected.

The runtime of C3PO with and without the adaptor was evaluated in order to measure improvement in computing performance. Figure 10 shows how much time is needed to feed a collection of FITS characterization results into C3PO and generate a profile. To produce a profile, usually, we need to ingest characterization results into MongoDB²⁶ and then run a content profile generation procedure.

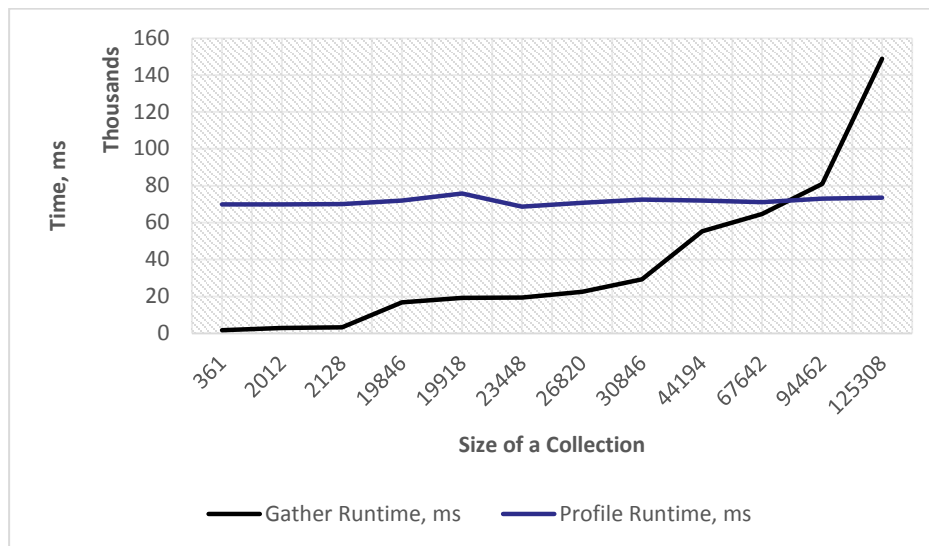


Figure 10 - Execution Time vs. Size of a Collection

As it may be noticed from the Figure 10, the profile generation runtime is almost constant, because of small collection size fully stored in RAM for effective calculations. However, inner mechanisms of MongoDB take some time to initialize map-reduce jobs which take almost 60 seconds to start. The gathering runtime is dependent on a collection size and is limited by Disk IO.

The developed adaptor skips the step of ingesting data into database, so there is only one process to run. Figure 11 contains comparison results of 2 cases:

²⁶ <http://www.mongodb.org/>

1. C3PO without the adaptor (gather and profile process execution),
2. C3PO with the adaptor

Overall time to create a profile in the first case is equal to a sum of two process runtimes.

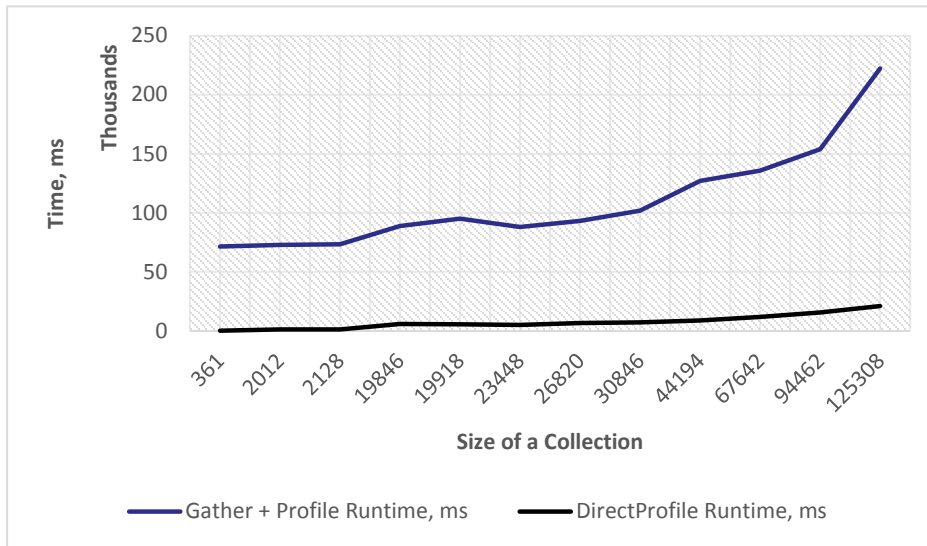


Figure 11 - Execution Time vs. Size of a Collection

The curve for the DirectProfile (a main function within the developed adaptor) execution does not increase as fast as Gather + Profile execution. With this adaptor, a content profile for the SB Web Archive dataset was generated in 49.6 hours of processing time on the desktop computer described previously.

Importing into Scout

Using the content profiles published in XML on an ad-hoc web server, Scout was able to import the data from the SB into the system. This presented its own difficulties due to the sheer amount of content. The last of harvests provides a collection size of about 13 TB and 550 million files. The format distribution table had more than 4300 different file formats. These values are just for the last harvest, and there were 8 harvests ingested into the system, though the previous ones were smaller. This amount makes the historic visualization graphs very slow and difficult to read. This could be improved in future work.

Figure 12 show the page overview of the last value of the collection profile (only last harvest). Though the system collects information from all harvests and plots them throughout time, like in Figure 13, the overview shows the last known state of the indicator, which corresponds to the last harvest. The overview shows that there are 62 different values for compression schemes used, and 4316 for file formats (though some of these are errors, conflicts, or name mismatches). Also, the overview shows that the average file size is 25.8 KB.

Figure 13 shows an example of the view throughout time, in which we can see that the content size increased from about 400 GB in 2006 to 13 TB in 2013. The increase of the content size can be due to changes in harvest policies but also to the internet natural growth. This kind of analysis can be done

to any property harvested by Scout. For example, we could analyse the evolution of the image format market shares by comparing the number of each of the image formats in the format distribution with the collection size.

Scout Home Query Browse admin Dashboard Administration

Categories / content_profile / sbweb

Entity

Name sbweb

Properties

Name	Value	Action
Collection size	13.14 TB	
compression_scheme distribution	62 key-value pairs	
Format distribution	4316 key-value pairs	
Objects avg size	25.8 kB	
Objects min size	0 bytes	
Objects max size	4.93 GB	
Objects count	546924526	

Figure 12 - Content profile overview of a large scale collection from the Danish Web Archive

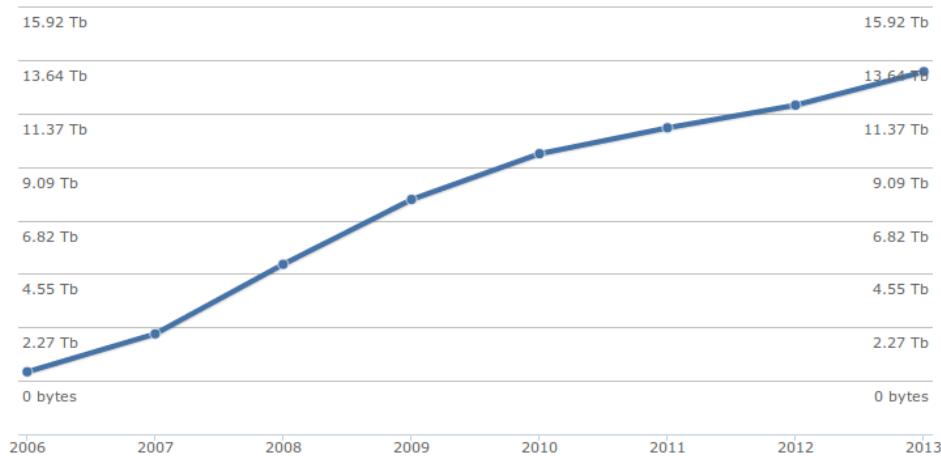


Figure 13 - Visualization of the collection size property throughout time, each point is a different harvest

Table 4 - Top 10 of the most common formats of the last harvest of the Danish web archive sample

Format name	Number of files	Percentage
JPEG File Interchange Format	157,939,162	28.88%
Hypertext Markup Language	147,345,550	26.94%
Graphics Interchange Format	87,649,358	16.03%
Conflicted	75,995,185	14.00%
Plain text	41,754,511	7.63%
Portable Network Graphics	17,682,954	3.23%
Extensible Markup Language	7,484,183	1.37%
Exchangeable Image File Format	4,078,432	0.75%
Portable Document Format	3,490,794	0.64%
MS Windows icon resource	604,771	0.11%

Table 4 shows the top 10 of the detected formats in the collection. There is a significant fraction of conflicts (14%), which are due to incoherence on the format names between different format identification tools available in FITS. Further analysis of the list of more than 4300 file formats, too big to list on this deliverable, shows that there are cases of program errors that are spilled into the output as file formats (e.g. “cannot open ...”), and content characteristics that are appended to file format names, which skew the file format distribution (e.g. “MS Windows icon resource - 2 icons, 16x16, 256-colors”). Using mime types, appended with format version information, might be a better alternative to the file format name in getting file format distribution information. This is listed as an enhancement for future work.

3.2.1.2 Experiment with data selected over 10 years

The Internet Memory Foundation (IMF) web archive consists of web sites crawled on behalf of institutions and crawls made by the foundation’s initiative. Content crawled on behalf of institutions are either made freely made available online through IMF public access, either served on-site when the institution is not allowed to offer a public access (e.g.: legal deposit not in place).

Although content hosted within the IMF web archive can be used within the project, neither was it always possible to provide content crawled to other project partner nor to list crawled websites used as part of an experiment such as the FITS characterisation.

IMF used a complementary approach on the deep characterization of Web content and selected a sample of 10 representative websites over its web archive. The targeted timeframe was initially set up to 10 years, with the characterization of two snapshots taken for each website during each year of the period. The process started in early 2012, with the characterization of 2012 snapshots and backwards up to year 2008. In the last period, we extended the timeframe over the year 2013 and also on the beginning of 2014.

The characterization processes have run on 3 dedicated servers, for several snapshots in parallel. Each server is built on a Jetway NF81-T56N-LF motherboard, with the configuration on Table 5.

Table 5 – Experiment with data selected over 10 years test environment

CPU	AMD G-T56N 1.6 GHz
RAM	8 GB
OS	Debian Squeeze

Disks	5 x 3TB
-------	---------

Prior to launching the characterization processes on the selected websites, IMF performed a short evaluation of the FITS performances on the internal cluster, trying to fine-tune the best configuration for the given infrastructure. Among others, we switched on and off the JHOVE analyser for HTML files²⁷ and we increased the memory dedicated to the JVM to 512MB, for each FITS process.

In the first phase, IMF performed a strict selection of the web content specific to each website in the list. This basically consisted in building the lists of ARC files resulting from different snapshots taken as a focused crawl of each website. By clustering thus the content of the archive, the characterization results are also more specific to each website and their evolution can be analysed over the timeframe. The outputs from the deep characterization were made available on an ftp server.

In the second phase, IMF also deployed several C3PO instances on the cluster, in order to import the characterization results and to build the C3PO profiles for each website and each snapshot. These profiles were made available on an http server as well, so that an automatic import in Scout is possible.

The C3PO profiles are grouped together in a global collection, organized in sub-collections specific to each snapshot of each website:

```
<collections>
    <collection name="website1_2010-05-01"/>
    <collection name="website2_2011-12-01"/>
    <collection name="website1_2012-12-01"/>
    ...
</collections>
```

Scout C3PO adaptor allows to import and monitor the provided C3PO profiles. Figure 14 shows the result of the import of one of the domains as the evolution of file formats throughout a 4 years' time.

²⁷ JHove handler for HTML files does not give any useful content and has many performance problems.

Property history

This property has changed in time as represented in the chart below. Click on the chart dots for more information.

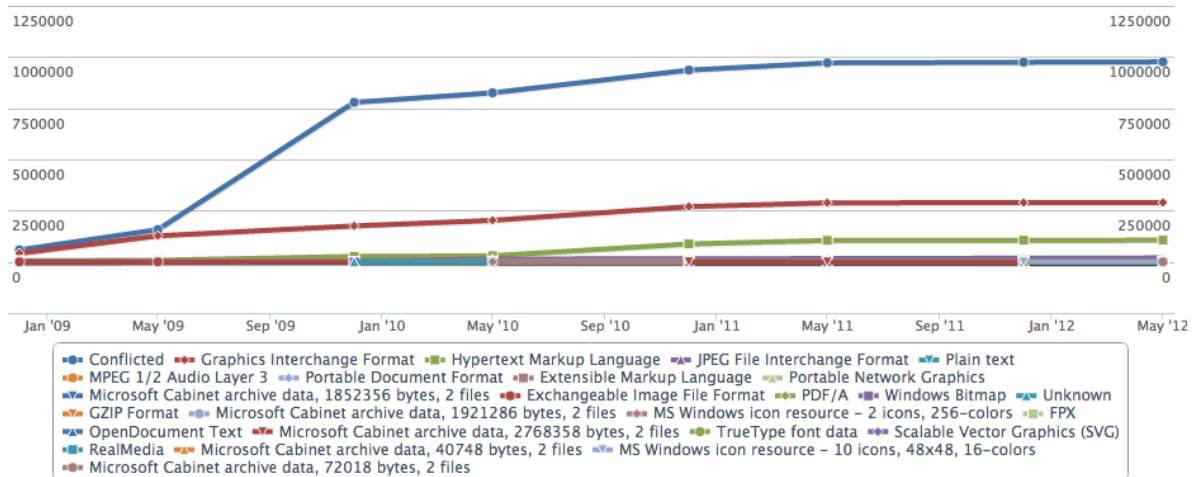


Figure 14 - A format distribution of 1.4 million files from the Internet Memory Foundation web archive

3.2.2 Browser renderability

The web archive content is very dependent on the web browsers that open that content. But web browsers change the way that content is rendered through time, and also change the way they support older file formats, especially the ones that need plugins, such as Adobe Flash.

To tackle this problem, a tool named **browser-shots** was created with the goal to perform automatic visual comparisons, in order to detect rendering issues in the archived Web pages.

The browser-shots tool started by using MarcAlizer tool²⁸, developed by the University Pierre et Marie Curie within the SCAPE project, which performs the visual comparison between two web pages. In a second phase, the renderability analysis also included the structural comparison of the pages, which is implemented by the new Pagelyzer tool²⁹.

Since the core analysis for the renderability is thus performed by an external tool, the overall performance of the browser-shot tool will be dependent on this external dependency. The overall performance can be greatly improved by new releases issued from the MarcAlizer development, as well as the updates on the tool issued from a more specific training, and future work includes continued integration of these with the browser-shots tool.

The detection of the rendering issues is done in the following three steps:

1. Web pages screenshots automatically taken using Selenium framework, for different browser versions.
2. Visual comparison between pairs of screenshots using MarcAlizer tool (recently replaced by Pagelyzer tool, to include also the structural comparison).

²⁸ <https://github.com/openplanets/MarcAlizer>

²⁹ <https://github.com/openplanets/pagelyzer>

3. Automatically detect the rendering issues in the Web pages, based on the comparison results.

3.2.2.1 Initial Implementation

The browser-shots tool is developed as a wrapper application, to orchestrate the main building blocks (Selenium instances and MarcAlizer comparators) and to perform large scale experiments on archived Web content.

The browser versions currently used and tested are: Firefox (for all the available releases), Chrome (only for the last version) and Opera (for the official 11th and 12th versions).

The initial, sequential implementation of the tool is represented by several Python scripts, running on a Debian Squeeze (64 bits) platform. This version of the tool was released on GitHub³⁰ and we received some valuable feedback from the sub-project partners.

For the preliminary rounds of tests, we deployed the browser-shots tool on three nodes of IMF's cluster and we performed automated comparisons for around 440 pairs of URLs. The processing time registered in average was about 16 seconds per pair of Web pages. These results showed that the existing solution is suitable for small-scale analysis only. Most of the time in the process is actually represented by IO operations and disk access to the binary files for the snapshots. Taking the screenshots proven to be very time consuming and therefore if this solution is to be deployed on a large scale, the solution needed to be further optimized and parallelized.

These results showed also that a serious bottleneck for the performance of the tool is represented by the passage of intermediary parameters in between the modules. More precisely, the materialization of the screenshots in binary files on the disk is a very time consuming operation, especially when considering large scale experiments on a large number of Web pages. Due to this bottleneck a new implementation of the tool was done, using an optimized version of MarcAlizer. The Web pages screenshots taken with Selenium are directly passed over to MarcAlizer comparator using streams and the new implementation of the browser-shots tool is represented by a MapReduce job, running on a Hadoop cluster. Based on this framework, the current rounds of tests could be extended up to much higher number of pairs of URLs.

In this second round, the browser shot comparison tool is implemented as a MapReduce job to parallelize the processing of the input. The input in this later case is a list of URLs that together with a list of browser versions, that are used to render the screen shot - note the difference in comparison to the former version where the input were pairs of URLs that were rendered using one common browser version and these were compared.

3.2.2.2 Optimizations

In order to achieve acceptable running times of the tool newer version of the MarcAlizer comparison tool was integrated into this tool. The major improvement brings the possibility of feeding to tool with in-memory objects instead of pointers to files on disk. This improvement and the elimination of the unnecessary IO operations led into following average times measured for the steps in the shot comparison:

1. Browser shot acquirement: 2s

³⁰ <https://github.com/crawler-IM/browser-shots-tool>

2. MarcAlizer comparison: 2s

The time taken to render the screenshot using a browser may vary on the rendered page size. For example, capturing The Wall Street Journal web page³¹ takes about 15s on the IMF machine where the resulting PNG image has several MBs (in direct proportion to the page size); this is well above the experiment medium time for capturing a web page. Therefore, the experiment results are highly dependent on the selected web pages size and complexity.

3.2.2.3 MapReduce

As can be seen, the operations on the operations on the screenshots are very expensive (the list of the tested browsers can be very long and for each we need to spend one browser screen shot operation). Therefore we needed to parallelize the tool to several machines working on the input list of URLs. To facilitate this, we employed Hadoop MapReduce which is part of the SCAPEs platform.

The result of the comparisons is then materialized in a set of XML files where each file represents one pair of browser shots comparisons. In order to alleviate the problem of having big numbers of small files, these files are automatically bundled together into one ZIP file. A C3PO adapter has been implemented so the result can be processed and passed further to Scout.

3.2.2.4 Tests

We ran preliminary tests on the currently supported browser versions - Firefox and Opera. The list of URLs to test is about 13 000 entries long. We are using the IMF central instance for these tests, currently having two worker nodes (thus we can cut the processing time to half in parallel execution).

A large scale test is to be made as part of the final Testbed Work Package test and benchmarking.

In addition to the web pages rendering comparison, this large scale final test will provide statistics about format and format versions contained within a sample of the IMF web archive with the aim of extracting information about the web technological landscape and typical rendering errors due to format not being rendered any more by some browser versions.

Indeed, the Web pages generally represent complex containers that bring together different types of information with various representations: from simple text content and images to complex multimedia content (typically, audio and video files) and interactive objects. In the Web archiving terminology, the Web contents that are stored in external files, but included and shown inside a Web page are generally called "embeds". The rendering issues for embeds in the Web pages are essentially generated by a wrong interpretation of their file format or by the lack of support for a given format on different browser versions.

The detection of file formats can be done by an internal analysis of the file (using a deep characterization tool such as FITS or TIKa, for instance), but it can also be done by analysing the rendering context. The context is therefore the Web page and the format is represented by an embedded element.

Our proof of concept for format detection is based on the principle of joining the information retrieved from the visual comparison of page rendering with the information extracted from the text analysis of the Web page. On one hand, the visual comparison can spot the eventual rendering issues

³¹ <http://wsj.com>

on the Web page and identify the correspondent element in the DOM structure of the page. On the other hand, the text analysis of the page can reveal the existence of embedded material, based on the key-word description of its format.

When the two references match, the given format will be thus associated with the rendering issues. Applying the method at large scale will produce a statistical data to describe the distribution of formats on the Web.

3.3 Environment

Monitoring information from external influences, inferring trends from the technological landscape and getting information from external sources is a hard and broad task. Here, some examples of adaptors that fetch information from outside source are presented.

3.3.1 File format registries

PRONOM is one of the most known file format registries specialized for the digital preservation community. The PRONOM adaptor is enabling the watch component to gather information about different file formats. It connects to the PRONOM SPARQL end point³² and queries it for available formats. It extracts information such as name, version, mime, PUID (PRONOM Unique Identifier) and release date and adapts it to the watch knowledge base model.

This information can be used to cross-reference with other data, for example file format distribution from content monitoring, to enrich the information about formats in the repository.

3.3.2 Information from the Web

We investigated the use of Information Extraction technologies to automatically mine relevant information on preservation risks from large collections of Web text and integrate this information into Scout. As a primary use case we considered an information requirement from the National Library of the Netherlands, which is interested in knowing which publishers are responsible for which digital and analogue content in their electronic journal collection. As such information continuously changes (journals may appear or be discontinued, organizations may cease to exist), the challenge we addressed was automatically detect such information from Web content. We employed our information extraction system to find thousands of relevant pieces of information (Faria et al., 2013).

Other use cases we have since investigated were to automatically mine information on file types and their required software from statements in the Web. Preliminary experiments are encouraging and point to strong potential of integrating information from large quantities of unstructured data into Scout for the purposes of more informed preservation watch.

4 Beyond the prototype

This section presents an analysis of the community requirements for a preservation watch component, defining a prioritized list of the most important preservation threats to detect and monitor and what are the preferred methods to the detection and monitoring. This analysis is made via a survey of the digital preservation community, asking their opinion and current practice. The result of the analysis is a roadmap for the watch preservation component to go beyond the prototype into a product that is ready for community use.

³² <http://test.linkeddatapronom.nationalarchives.gov.uk/sparql/endpoint.php>

4.1 Survey: “Digital preservation: what to monitor and how?”

This survey aimed to identify the most relevant digital preservation threats, the preferable methods to detect them and what is the current practice in terms of digital preservation watch. The following sections show the details of the survey audience and results; Appendix A lists the original set of questions.

4.1.1 Invitation

The survey ran from 31st January to 28th February 2014. There was an initial invitation at the start date and a reminder at 24th February. The invitations and reminders were sent through many channels, such as the SCAPE page and newsletter, Open Planets Foundation blog, mailing list and LinkedIn, Digital Preservation Coalition news page, JISC mailing lists, and DIGLIB mailing lists. Also, requests were made to forward the invitation in the Nestor mailing list, DigCurV and DCC organization and projects, and the DLM forum. The invitation was further disseminated on social networks, such as Twitter and Facebook.

4.1.2 Profile

There were a total of 342 responses to the survey, although not all responses answered all questions. The country of each response was not requested, but anonymous web analytics show that there were a total of 588 visits³³ from many countries, mainly from Europe and the United States, see Table 6 for the top 10.

Table 6 - Top 10 countries that visited the survey site

Country	Visits	% of total visits
United Kingdom	196	33.33 %
United States	92	15.65 %
Germany	41	6.97 %
Netherlands	25	4.25 %
Italy	22	3.74 %
Portugal	18	3.06 %
Belgium	17	2.89 %
Switzerland	17	2.89 %
Canada	13	2.21 %
France	13	2.21 %
Totals	454 of 588	77.2 % of 100%

From the 259 respondents (76% of the total) that answered to the questions in the Profile section of the survey we can infer that responses mainly came from people in Universities and Memory institutions or content holders, followed by Government institutions, Small or medium enterprises, and Publishers or content producers (see Figure 15). On the Profile section we can also see that the respondents mostly have the role of Digital preservation manager or Archivist, followed by Information technology, Researcher, Organizational manager and Technical support (see Figure 16).

³³ How visits are calculated in Google Analytics: <https://support.google.com/analytics/answer/2731565?hl=en>

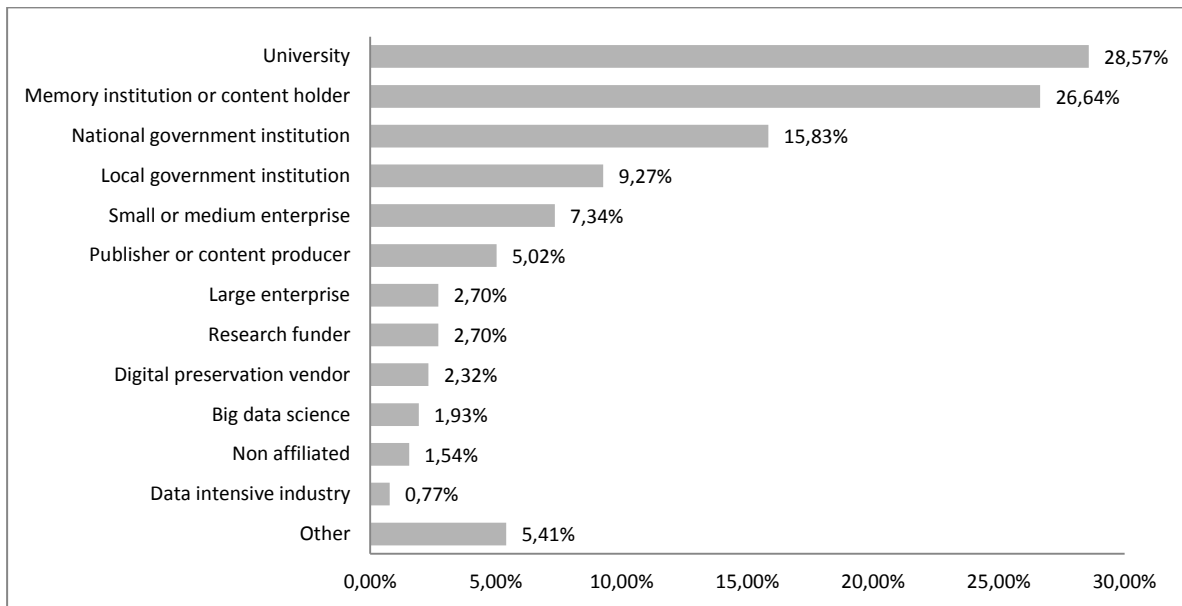


Figure 15 - What descriptions fit your organization?

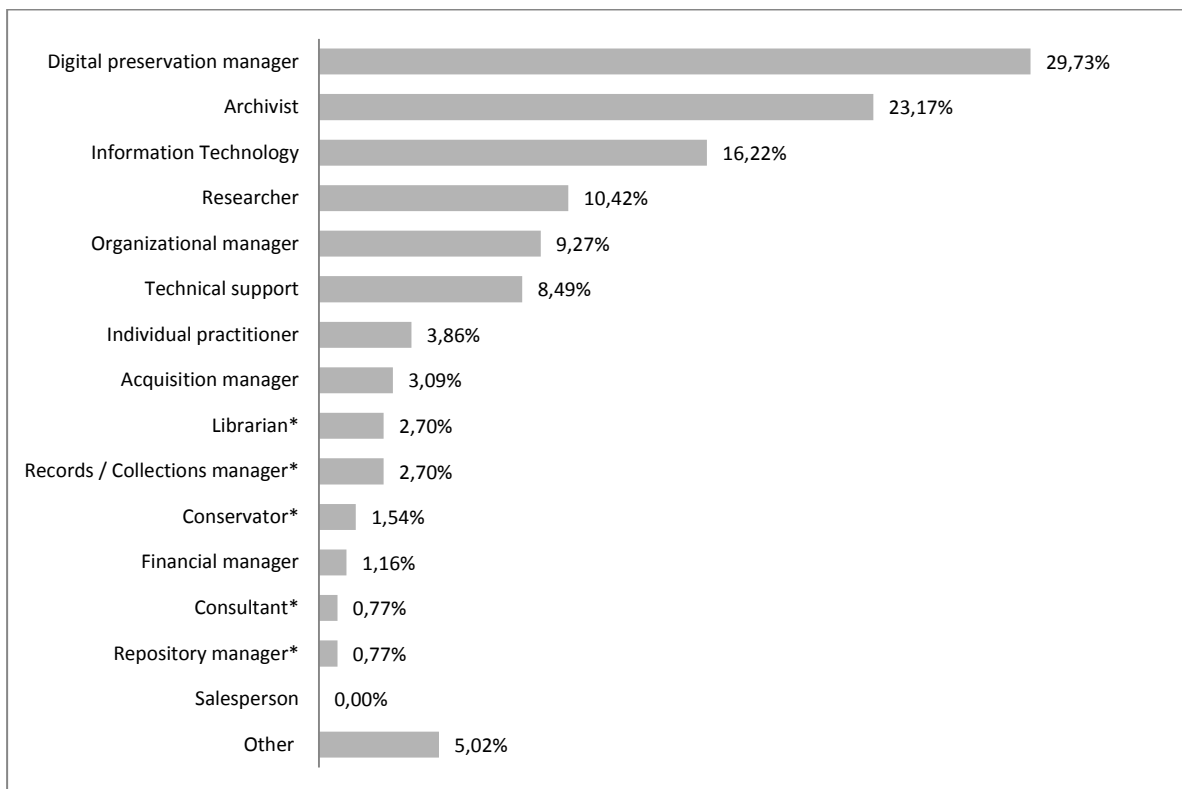


Figure 16 - What description fit your role in the organization?

4.1.3 Digital preservation threats

Preservation watch mainly focuses on the detection of preservation threats (or opportunities), that is the possibility that an incident³⁴ with negative impact (or positive if it is an opportunity) on the preservation of a digital object might occur. Knowing which preservation threats are more important for the community and which are already monitored gives an important insight into what value the preservation watch component can bring to the community and what are the gaps future work should focus on.

To set the context for users and allow some normalization of response and prioritization, a suggested list of preservation threats was created by a focus group within SCAPE partners and the user group, which was tested against a small audience at the SCAPE training event in Aarhus (“Effective, Evidence-Based Preservation Planning”³⁵). In Table 7 it is presented the suggested list of preservation threats, with short names for convenience in identifying them on diagrams.

Table 7 - Suggested preservation threats

Threat	Short name
File corruption	File corruption
Backup failure	Backup failure
Hardware no longer supported or degraded	Hardware platform obsolescence
Software platform no longer supported or degraded	Software platform obsolescence
A relevant percentage of the producers cannot comply with the established ingest policies	Producers misalignment
A relevant percentage of the consumers cannot read the disseminated file format	Consumers misalignment
There is not enough context information to understand the file content	Lack of context information
Content does not conform with defined institutional policies	Content not aligned with policies
Organization staff is not enough or adequately trained to maintain content	Staff not enough or adequate
Actions executed on files (e.g. file format migration) are not having expected results	Incorrect action results
Defined preservation plans (e.g. defined preservation format) became outdated	Outdated preservation plans

The respondents were asked for their opinion on the importance of each of the preservation threat on a 1 to 5 scale, where:

- **1:** Not at all important
- **2:** Slightly important (Informational)
- **3:** Important (Requires action but with low priority)
- **4:** Fairly important (Requires action with average priority)
- **5:** Very important (Requires urgent and immediate action)

³⁴ A preservation incident, in this context, means a discrete occurrence of an event with positive or negative impact on the preservation of a digital object, e.g. file X was corrupted at date Y, or better disk drives were installed at server Z at date Y.

³⁵ <http://wiki.opf-labs.org/display/SP/SCAPE+Training+Event+-+Effective,+Evidence-Based+Preservation+Planning>

Figure 17 shows the result of the question which was answered by 181 respondents. All threat importance levels have an arithmetic mean between 3.1 and 4.6, which show that all suggested preservation threats are viewed as important. The black line on the limit of every bar illustrates the standard deviation, which was between 0.77 (for file corruption) and 1.1 (for incorrect action results), which mean that more agreement was found for file corruption than for the incorrect action results, but in average there is a fair agreement on the importance of all preservation threats. Nevertheless, we can verify that file corruption and backup failure stand out in importance, which could be explained by the fact that these are the only two threats that relate to the physical preservation of content. Also, we can verify that outdated preservation plans, producer misalignment, and content not aligned with policies stand out by their relative low importance.

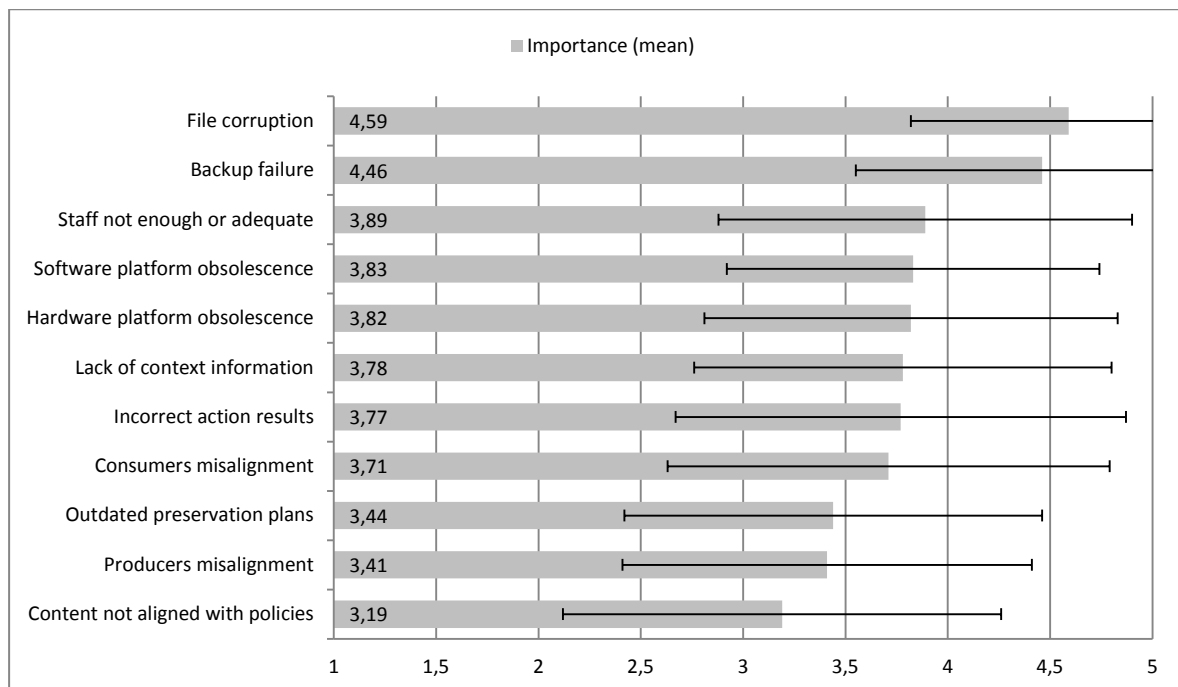


Figure 17 – Importance of digital preservation threats

Figure 18 shows the current practice on monitoring these preservation threats. The 181 respondents that got it this point were asked which of the preservation threats they monitor, they don't monitor, or if there are uncertain. Backup failure stands out as the most monitored threat, followed by hardware and software platform obsolescence, file corruption and staff not enough or adequate. Consumer misalignment stands out as the least monitored threat, followed by lack of context information, producer misalignment, incorrect action results and outdated preservation plans.

In order to relate the importance to the current practice, so a useful prioritization of the most important threats that are less monitored could be drawn, the following score formula was devised. The formula normalizes the importance mean to a 0-1 scale and multiplies by the ratio of not monitoring respondents relative to the respondents which knew if they were monitoring or not (this ignores responses with Uncertain or no answer). This score gives equal value to the importance and to the current practice; this is an assumption that could be studied further.

$$ThreatScore = \frac{ImportanceMean - 1}{4} \times \frac{NotMonitoring}{Monitoring + NotMonitoring}$$

Table 8 shows the score of all suggested preservation threats and highlights the ones above the mean score. As expected, the threats with low ratio of monitoring practice have the higher score, such as consumer and producer misalignment, lack of context information, incorrect action results and outdated preservation plans. Surprisingly, file corruption is also above mean because, even though it has a relatively high ratio of monitoring practice, it is a very important threat to monitor and the 18% of responses state that they do not monitor it.

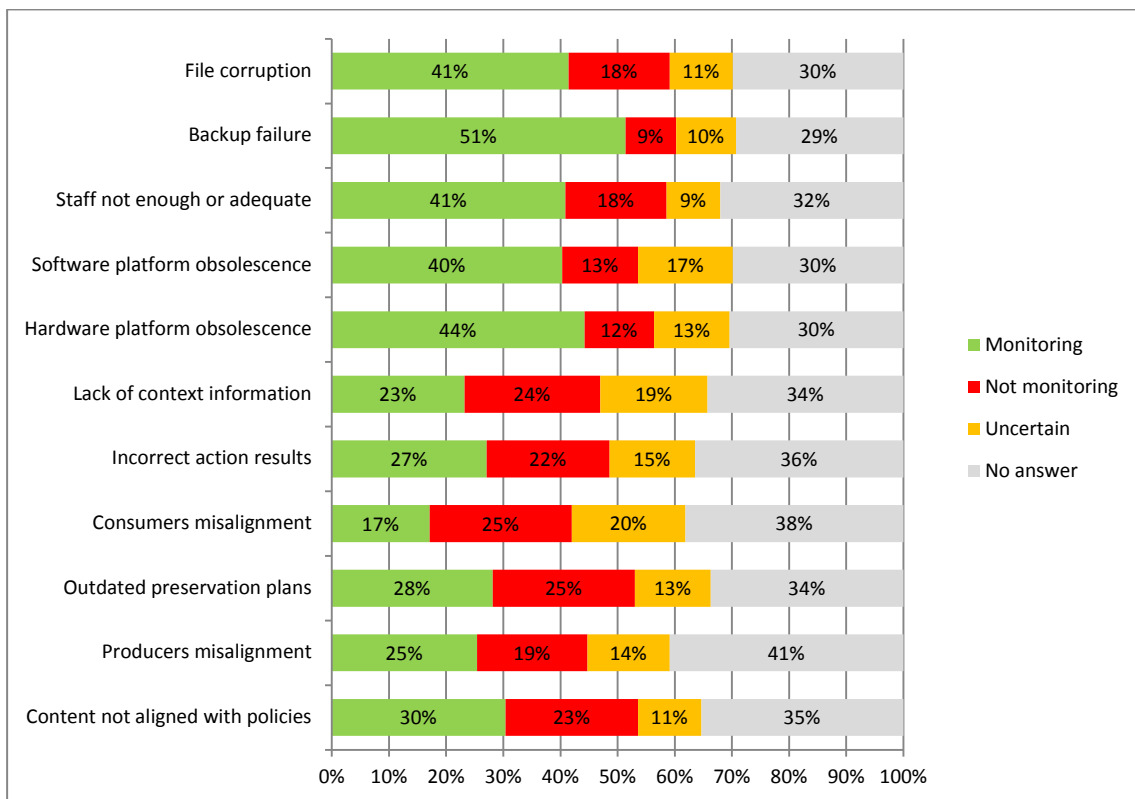


Figure 18 - Current practice on monitoring digital preservation threats

Table 8 - Preservation threat score

Threat short name	Score	Above mean
File corruption	0,268	Yes
Backup failure	0,127	No
Staff not enough or adequate	0,218	No
Software platform obsolescence	0,175	No
Hardware platform obsolescence	0,152	No
Lack of context information	0,352	Yes
Incorrect action results	0,307	Yes
Consumers misalignment	0,401	Yes
Outdated preservation plans	0,286	Yes
Producers misalignment	0,260	Yes
Content not aligned with policies	0,237	No

An open question for respondents to point out other preservation threats that they perceived as important provided the following results. The responded threats below are rephrased and categorized to allow a better reading.

Lack of content or metadata completeness:

- Undefined preservation metadata

Issues on content capture or production:

- Degradation of analogue media and capture hardware, lack of skilled personnel
- Not enough digital object representation capability or production environment information to match the content creator expectations
- Capacity to recover from analogue media and execute quality assurance

Technical capabilities or shortcomings:

- Capability to identify file formats
- Capability to validate files (e.g. check if PDF files are valid and well-formed)
- No adequate preservation action available (e.g. file format migration or upgrade has unacceptable quality and emulation is not possible/feasible/allowed)
- Long-term access to propriety software
- Tools to validate and/or migrate are not robust

Archival storage and service:

- Security breach, malicious/accidental tampering
- Service availability
- Inadequate network services (especially for audio-visual assets)

Human or organizational environment:

- Digital preservation awareness
- Organizational or political change
- Loss of contact information for maintenance purposes (e.g. loss of tacit knowledge, know-how and ways to recover it)
- Lack of budget, management support or human resources
- Lack of clear standards for specific asset types (e.g. audio-visual or geographical datasets)

4.1.4 Detecting and monitoring preservation threats

A list of methods to detect or monitor each one of the previously suggested preservation threats was defined, using the same focus group and trial survey as explained in the previous section. Table 9 describes all suggested methods and their short name, for convenience reading on the following diagrams.

Table 9 - Suggested methods to detect or monitor preservation threats and their short names

Threat	Threat short name	Detect method	Detect method short name
File corruption	File corruption	Check files manually	Check files manually

		Automatic file fixity checks	Automatic file fixity checks
Backup failure	Backup failure	Manual verification of backup success	Manual verification
		Alerted by backup program when failure occurs	Alerted by backup program on failure
		Notified by backup program on every execution	Notified by backup program every time
		Third-party program monitors correct functioning of backups	Monitored by 3rd party program
Hardware no longer supported or degraded	Hardware platform obsolescence	Manual analysis of the hardware inventory by IT staff	Manual by IT staff
		Manual analysis of hardware inventory by preservation experts	Manual by preservation experts
		Have a close relationship with hardware vendors	Relationship with vendors
		Automatic cross-reference of hardware inventory with a known hardware issues registry	Crossing inventory with issues registry
Software no longer supported or degraded	Software platform obsolescence	Manual analysis of software inventory by IT staff	Manual by IT staff
		Manual analysis of software inventory by preservation experts	Manual by preservation experts
		Subscribe relevant mailing lists or other information channels	Subscribing mailing lists and others
		Automatic cross-reference of software inventory with known software issues registry	Crossing inventory with issues registry
A relevant percentage of the producers cannot comply with the defined ingest policies	Producers misalignment	Feedback by direct engagement with producers	Direct engagement with producers
		Monitoring producer trends and problems by analysis of content from the web	Trends from web analysis
		Monitor ingest process (e.g. SIP rejection statistics)	Monitor ingest process
A relevant percentage of the consumers cannot read the disseminated file format	Consumers misalignment	Feedback by direct engagement with consumers	Direct engagement with consumers
		Consumer feedback by email or analogous channels	Feedback by email or other channel
		Repository integrated consumer feedback	Repository integrated feedback
		Manual analysis of file formats in your collections by IT staff	Manual check collections by IT staff
		Manual analysis of file formats in your collections by preservation experts	Manual check collections by preservation experts

		Automatic cross-reference of file formats in your collections with tools that can render formats	Crossing formats with rendering tools
		Automatic cross-reference of file formats in your collections with information available on format registries	Crossing formats with format registries
		Automatic cross-reference of file formats in your collections with known file format issues	Crossing formats with format issues
		Automatic analysis of consumer trends by inspection of consumer used software (e.g. browsers and operative systems)	Trends from web analysis
There is not enough context information to understand the file content	Lack of context information	Manual inspection of content on ingest	Manual check content on ingest
		User feedback	User feedback
		Automatic verification that external references still exist (e.g. web sites)	Check external references
Content does not conform with defined institutional policies	Content not aligned with policies	Manually verify content on ingest and whenever applicable policies change	Manual check content on policy change
		Define control policies in a machine readable format and have tools that automatically check the conformance	Automatic check with control policies
Organization staff is not enough or adequately trained to maintain content	Staff not enough or adequate	Manually evaluate staff performance and quality	Manual staff evaluation
		Have automatic indicators of staff performance (e.g. objects ingested, described, words written)	Automatic indicators of staff performance
		Have consumer feedback on the quality of content description and cataloguing	Consumer feedback on staff performance
Actions executed on files (e.g. file format migration) are not having expected results	Incorrect action results	Manually verify all (or a sample) of the action results	Manually verify the action results
		Run quality assurance tools on action results and automatically check against expected results	Automatic check with Q&A tools
Defined preservation plans (e.g. defined preservation format) became outdated	Outdated preservation plans	Update preservation plans periodically (e.g. yearly)	Update plans periodically
		Use tools to create preservation plans (e.g. Plato) and be automatically notified when assumptions taken may have become invalid	Use plan creation tools that notify automatically

The 111 respondents who got to this part of the survey were asked for their opinion on the preference of each of the preservation threats on a 1 to 7 scale, where:

- **1:** Completely disagree
- **2:** Disagree
- **3:** Somewhat disagree
- **4:** Neither agree nor disagree
- **5:** Somewhat agree
- **6:** Agree
- **7:** Completely agree

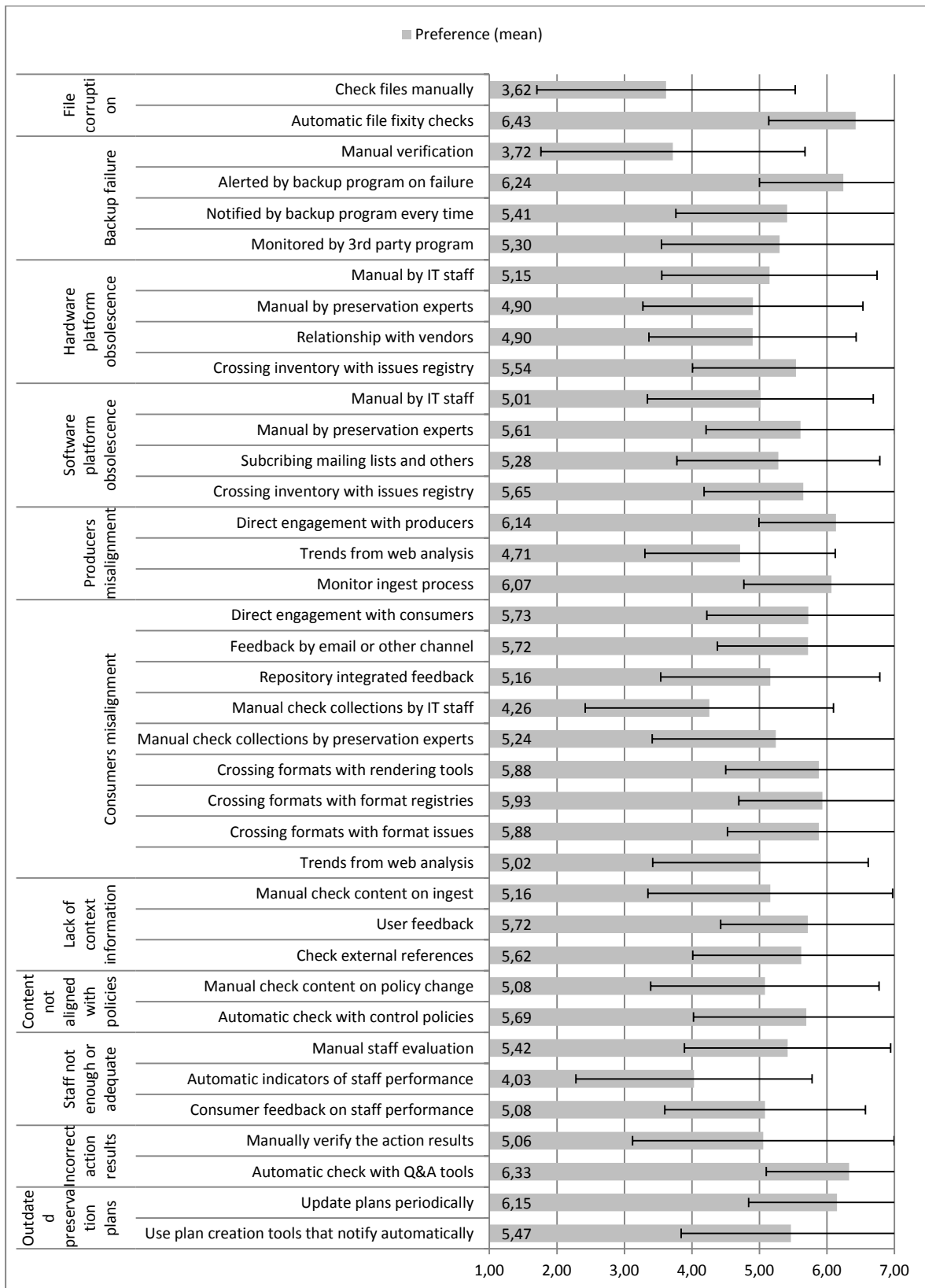


Figure 19 - Preference on methods to detect or monitor preservation threats



Figure 20 - Current practice on methods to detect or monitor preservation threats

Figure 19 shows the preference on each of the suggested methods while Figure 20 shows the current practice on the use of each of the methods. To relate the two figures, the following formula was devised to create a score for the preservation monitoring method. As preference can have a depreciative value, it was normalized in a -1 to 1 scale on the *NormalizedPreference* variable, and the current practice of not using the method, the *DontUseRatio* variable, was made relative to the percentage of responses that answered “Use” or “Don’t use”, ignoring “Uncertain” and “No answer” responses. The *MethodScore* is then the direct multiplication of the normalized preference with the ratio of people not using the method, if the preference is bigger than 0. For methods with negative normalized preference, the score is multiplied by the ratio of people that **do use** the method. This score tries to reveal the methods that need more attention, which are the most preferred methods with less current practice. Inversely, the score should be minimum when methods which the community disagrees that should be used and actually have a high current practice.

$$NormalizedPreference = \frac{PreferenceMean - 4}{3}$$

$$DontUseRatio = \frac{DontUse}{Use + DontUse}$$

$$MethodScore \begin{cases} NormalizedPreference \times DontUseRatio, & NormalizedPreference \geq 0 \\ NormalizedPreference \times (1 - DontUseRatio), & NormalizedPreference < 0 \end{cases}$$

Table 10 - Preservation threat monitoring method score

Threat short name	Detect method short name	Score	Above mean
File corruption	Check files manually	-0,082	NO
	Automatic file fixity checks	0,289	YES
Backup failure	Manual verification	-0,055	NO
	Alerted by backup program on failure	0,158	NO
	Notified by backup program every time	0,225	NO
	Monitored by 3rd party program	0,313	YES
Hardware platform obsolescence	Manual by IT staff	0,125	NO
	Manual by preservation experts	0,179	NO
	Relationship with vendors	0,147	NO
	Crossing inventory with issues registry	0,503	YES
Software platform obsolescence	Manual by IT staff	0,125	NO
	Manual by preservation experts	0,273	YES
	Subscribing mailing lists and others	0,132	NO
	Crossing inventory with issues registry	0,481	YES
Producers misalignment	Direct engagement with producers	0,181	NO
	Trends from web analysis	0,198	NO
	Monitor ingest process	0,273	YES
Consumers misalignment	Direct engagement with consumers	0,166	NO
	Feedback by email or other channel	0,156	NO
	Repository integrated feedback	0,320	YES
	Manual check collections by IT staff	0,052	NO
	Manual check collections by preservation experts	0,182	NO
	Crossing formats with rendering tools	0,502	YES
	Crossing formats with format registries	0,434	YES
	Crossing formats with format issues	0,500	YES
	Trends from web analysis	0,293	YES
Lack of context information	Manual check content on ingest	0,106	NO
	User feedback	0,209	NO
	Check external references	0,475	YES

Content not aligned with policies	Manual check content on policy change	0,134	NO
	Automatic check with control policies	0,414	YES
Staff not enough or adequate	Manual staff evaluation	0,173	NO
	Automatic indicators of staff performance	0,010	NO
	Consumer feedback on staff performance	0,221	NO
Incorrect action results	Manually verify the action results	0,125	NO
	Automatic check with Q&A tools	0,534	YES
Outdated preservation plans	Update plans periodically	0,387	YES
	Use plan creation tools that notify automatically	0,450	YES

For each preservation threat, the respondents were also asked about other ways to detect that threat. Next, we analyse the score of the suggested methods for each threat and give a summarized list of the other recommended monitoring or detection methods.

File corruption

The manual verification of file corruption was, understandably, one of the few methods with negative score. The automatic file fixity checks were preferable and the score was above mean. Other methods recommended by the respondents were:

- User feedback
- Use of file validation (and characterization) tools like JHOVE, Exiftool, Droid, etc.
- Manual check of sampled files
- Comparison of similarity of visual content
- Automatic error detection on storage level
- Manual readability check on sampled files

Backup failure

On detecting backup failure, the manual verification was one of the few methods with negative score. The method with higher score was monitoring with a 3rd party program. Other methods recommended by the respondents were:

- Periodic fixity checking of backups
- Disaster recovery testing to verify backup procedures

There were also some comments regarding this item as not helpful in inactive archives, as content does not change, or viewing backups as an outside matter, i.e. not related with the archive, managed by central IT or storage service provider.

Hardware platform obsolescence

All suggested methods had positive score, but crossing inventory with issues registries stands out with a score above mean. Other methods recommended by respondents were:

- Review of published hardware lifecycle milestones and roadmaps
- Awareness of the state of industry and common practices
- Monitoring of usage in reading room
- Maintain multiple generations of hardware

There were also comments that this subject was not a subject of digital preservation but of IT management.

Software platform obsolescence

All suggested methods have positive score, but crossing inventory with issues registry stands out, followed by manual analysis by preservation experts, which is also above mean. Other methods recommended by respondents were:

- Relationship with software vendor
- Review of published software lifecycle milestones and roadmaps
- User feedback
- Creation of knowledge base of what software versions are compatible with files formats
- Monitoring of usage in reading room
- Information from the Web
- Engagement with other users of the same preservation software

Producer misalignment

All suggested methods have positive score, but monitoring ingest process stands out with a score above mean. Other methods recommended by respondents were:

- Auto-detect compliance with machine readable transfer agreement
- Feedback from users and exchange information with other institutions using the same content

Other comments referred to the current inexistence of defined ingest policies, the general difficulty of controlling producers, and a recommendation to use open file formats.

Consumer misalignment

The methods with score above mean were cross-referencing formats with rendering tools, known format issues and format registries, have repository integrated feedback, and trends from web analysis. Other methods recommended by respondents were:

- Implement feedback channels

Lack of context information

All suggested methods have positive score, but check external references stands out with a score above mean. Other methods recommended by respondents were:

- Engage with the producer
- Verify (or enforce) the minimum required descriptive metadata was provided
- Define and verify formal metadata policies
- User feedback

Content not aligned with policies

The automatic check with control policies stands out with a score above mean. There were no other recommended methods, but comments referred to the need of working with producers and intended consumers to set up the rules before content is sent, and also that checking if content is aligned with policies should not be done only on ingest alone.

Staff not enough or adequate

Although all suggested methods have positive score, no method score was above mean. Other methods recommended by respondents were:

- Monitor amount of backlog due to lack of staff availability, or content not acquired due to lack of staff expertise, or expected content not available

There was also a comment referring to the need of organizational commitment to training and to include continued education and training as part of the digital preservation strategy.

Incorrect action results

Automatic check with quality assurance tools stands out with a score above mean. There were no other recommended methods but comments referred that testing with quality assurance tools with be the best solution but it would depend on the existence and accuracy of such tools. Also, another comment referred that quality assurance should not be completely trusted and that the original data should always be kept as a failsafe.

Outdated preservation plans

All suggested methods, update plans periodically and use plan creation tools and notify automatically had a score above mean. Other methods recommended by respondents were:

- Maintain preservation plans in a Technical Registry

4.1.5 Follow-up

The last part of the survey asked the respondent if they wanted to know more about SCAPE tools for preservation watch. Of the 91 respondents that got to this part of the survey:

- **53 (58%)** shared their contact information to receive further information on SCAPE preservation monitoring tools;
- **38 (42%)** stated that they would like to run our tools to know their file format distribution, along with other characteristics, and compare it with others;
- **49 (54%)** would like to participate in workshops, virtual or face-to-face, to know how to use the digital preservation tools created in the SCAPE project.

4.2 Analysis of the prototype against survey results

The prototype for the preservation watch component was able to implement most of the methods with score above mean for the producer and consumer misalignment, content not aligned with policies, incorrect action results and outdated preservation plans.

For file corruption, the automatic file fixity check was the preferable method. Although automatic file fixity check has a score above mean, and therefore could be candidate to include as an adaptor for the preservation watch component, it can be argued that this feature is better fitted to storage and repository system to implement. On another side, the responses of the open question for other methods to detect file corruption pointed to the execution of file characterization and validation tools, which is exactly what is accomplished by monitoring of content with FITS, C3PO and Scout (see sections 3.1.1 and 3.2.1). FITS actually wraps most of the tools recommended (e.g. Droid, JHOVE, Exiftool).

For producer misalignment, the method with greater score was monitoring of the ingest process, which is accomplished by the prototype with the monitoring of repository events (see section 3.1.2). One of the defined events is ingest, and this event is accompanied with extra metadata which allows analysis and monitoring of the ingest process.

For consumer misalignment, it is already possible to cross-reference file formats with file format registries (name PRONOM, see section 3.3.1). Cross-referencing with format issues and rendering tools would be possible by adding source adaptors that provide information on format issues and

rendering tools. On the repository integrated feedback, some information could already be introduced via the repository events, if the repository would support gathering user feedback and provides that information via the Report API. Finally, the method of getting consumer trends via web analysis (i.e. inspection of consumer used software, e.g. browsers and operative systems) is also available via the repository events adaptor, see section 3.1.2.

For content not aligned with policies, the automatic check with control policies is already supported by the prototype using control policies defined in the SCAPE control policy model (refer to sections 2.1 and 3.1.3).

For incorrect action results, the method with higher score, automatic check with quality assurance tools, is already supported in the prototype via the repository events. An action plan created by Plato can include the execution of quality assurance tools, and their output can be provided to Scout via the Report API. Also, Plato defines Scout triggers which monitor if the action results are the expected, raising a notification if otherwise. See section 3.1.3 for details on monitoring incorrect action results.

For outdated preservation plans, the prototype already allows the method of using plan monitoring tools that notify automatically, i.e. use tools to create preservation plans (e.g. Plato) and be automatically notified when assumptions taken may have become invalid. This is exactly what the prototype supports, as it integrates with Plato and allows it to automatically create triggers on which monitor the assumptions taken on the decision-making process.

Of the preservation threats with score above mean only the lack of context information is not monitored currently monitored by the prototype, but plans are already in place to develop adaptors that focus on that preservation threat, specifically on the research data domain, see section 4.3.1.

4.3 Roadmap for future developments

This section identifies the most important features for future developments and also developments that are still planned to occur within the SCAPE project, namely the Simulator.

4.3.1 Research data

Research data is one of the three SCAPE testbed themes, together with Large Scale Digital Repositories and Web Archives, but it was deliberately omitted from the initial objectives of this work package as it typically a very complex subject. Nevertheless, there are plans to use Scout to validate research object by checking the completeness and whereas external links are still valid. More information about the possibilities to use Scout with research data is available at (Jones & Matthews, 2013).

4.3.2 Human input

To allow generic information to be manually inputted into the system, a web form could be created that would allow an authenticated user to add or improve information on the knowledge base. The web forms could be optimized for specific types of data, to improve the data quality and usability.

Some possible types of manually inserted information are:

- Generic risks or problems associated with file formats or tools
- Legal context or restrictions that are relative to some countries or types of assets

- Improvement of information available about file format (e.g. open source status, standardization, acceptance wideness, complexity) or tools (e.g. ease of use, price, usage)
- Information of the cost of hardware and software needed for digital preservation (e.g. cost of storage)

4.3.3 Simulator

One of the major challenges when dealing with huge amount of heterogeneous data is to anticipate the future state of a repository. Simulation methods are useful for addressing this kind of a problem because they offer a cheap way of testing repository responses in different scenarios. Thus, this will enable an interested user to get a better overview of possible future resource requirements. To enable this two parts of a simulation environment are build: simulation engine and simulation language. The simulation engine is based on discrete event simulation principles and is considered to be completely hidden from a user. The simulation language offers to a user a modelling environment which is much closely related to the digital preservation community that the simulation engine. The goal for the simulation engine is to be as simple as possible but also as expressive as possible. The simulation environment will be described in details in the D12.3.

The goal of the simulation tool is to provide an environment which can support simulating different assumptions. This can include distributions such as ingest or format evolution. The watch component contains data (mainly collection profiles) which can be analysed to derive such distributions. To perform such analysis bigger amounts of data is needed. At the moment web archive collection profiles are the only place where such data can be found. This analysis will be performed as a part of the simulation deliverable. Automatic connections between the watch component and the simulation environment are not planned at the moment.

5 Conclusion

The implementation of the preservation watch component was accomplished by the development of a prototype product named Scout.

Scout uses source adaptors to gather and monitor properties from the world, such as digital object repository content and events, web archives, and the technical and organizational environment just as file format registries and any information from the Web. The source adaptors are developed based on plugins, which can be easily developed and added into the system, extending Scout and adding more information about the world.

All information goes into a centralized knowledge base which is based on the Linked Data principles and allows representation of any kind of information, cross-referencing and basic reasoning to allow detection of preservation threats.

Scout allows human users, via a web interface, and software components, via a programmatic interface, to browse the gather information and pose questions about entities and properties of interest by using SPARQL or pre-defined question templates.

Furthermore, Scout enables the creation of triggers, which periodically monitor a watch question and notify users when non-conformities are found, for example by email.

This system achieves all defined goals and its functionality was put into test with several localized experiments that looked into several aspects of the system. Large-scale experiments showed the existing bottlenecks and limitations of the system or some of its components, and allowed to enhance the system to work with multi-terabyte collections. Other experiments looked into possible problems on gathering specific information, like renderability analysis of web content, or how to use information extraction tools to extract information encoded in natural language from the Web. These are only examples of information that would have interest to monitor, stemmed from the needs of content holder partners.

The development of more adaptors that fit to the purpose of the several stakeholder types in the digital preservation community will be a continuous effort, but the easiness of development of new adaptors and openness of the system will allow for the community to develop their own adaptors and contribute to the enrichment of the available information in Scout. The presented survey shows that the prototype already achieves the most important preservation threats and preferred monitoring methods, nevertheless the success of Scout will always depend on the community engagement on providing information and contributing to the system. Digital preservation watch is a continuous and community effort and Scout can become the platform that allows the community coming together for the greater good.

6 References

- Antunes, G., Becker, C., Borbinha, J., Proença, D., Vieira, R., & Barateiro, J. (2011). *SHAMAN Reference Architecture (version 3.0)*. SHAMAN project report.
- Bechhofer, S., Sierman, B., Jones, C., Elstrom, G., Kulovits, H., & Becker, C. (2013). *Final version of Policy specification model*. SCAPE project deliverable (D13.1).
- Becker, C., Duretec, K., Petrov, P., Faria, L., Ferreira, M., & Ramalho, J. C. (2012). Preservation Watch: What to monitor and how. In *iPres'12*. Toronto, Canada.
- Becker, C., Faria, L., & Duretec, K. (2014). Scalable Preservation Intelligence for Information Longevity. *OCLC Systems & Services*.
- Duretec, K., Faria, L., Petrov, P., & Becker, C. (2012). Identification of triggers and preservation Watch component architecture, subcomponents and data model. SCAPE D12.1.
- Faria, L., Akbik, A., Sierman, B., Ras, M., Ferreira, M., & Ramalho, J. C. (2013). Automatic Preservation Watch using Information Extraction on the Web. In *iPres'13*. Lisbon, Portugal.
- Faria, L., Petrov, P., Duretec, K., Becker, C., Ferreira, M., & Ramalho, J. C. (2012). Design and architecture of a novel preservation watch system. In *ICADL'12* (pp. 168–178). Taipei, Taiwan: Springer. doi:10.1007/978-3-642-34752-8_23
- Hamm, M., & Becker, C. (2011). *Report on decision factors and their influence on planning*. SCAPE Project Deliverable (D14.1).

Jones, C., & Matthews, B. (2013). *White Paper: Context and linking for Research Data*.

Kniff, J. van der, & Wilson, C. (2011). *Evaluation of characterization tools - Part 1: Identification*.

Petrov, P., & Becker, C. (2012). Large-scale content profiling for preservation analysis. In *iPres'12*. Toronto, Canada.

Sierman, B., Jones, C., Bechhofer, S., & Elstrom, G. (2013). Preservation Policy Levels in SCAPE. In *International Conference on Preservation of Digital Objects*. Biblioteca Nacional de Portugal.

Sierman, B., Jones, C., & Elstrom, G. (2014). *Catalogue of preservation policy elements*. SCAPE Project deliverable (D13.2).